

# VCF DEVREADY WITH TANZU

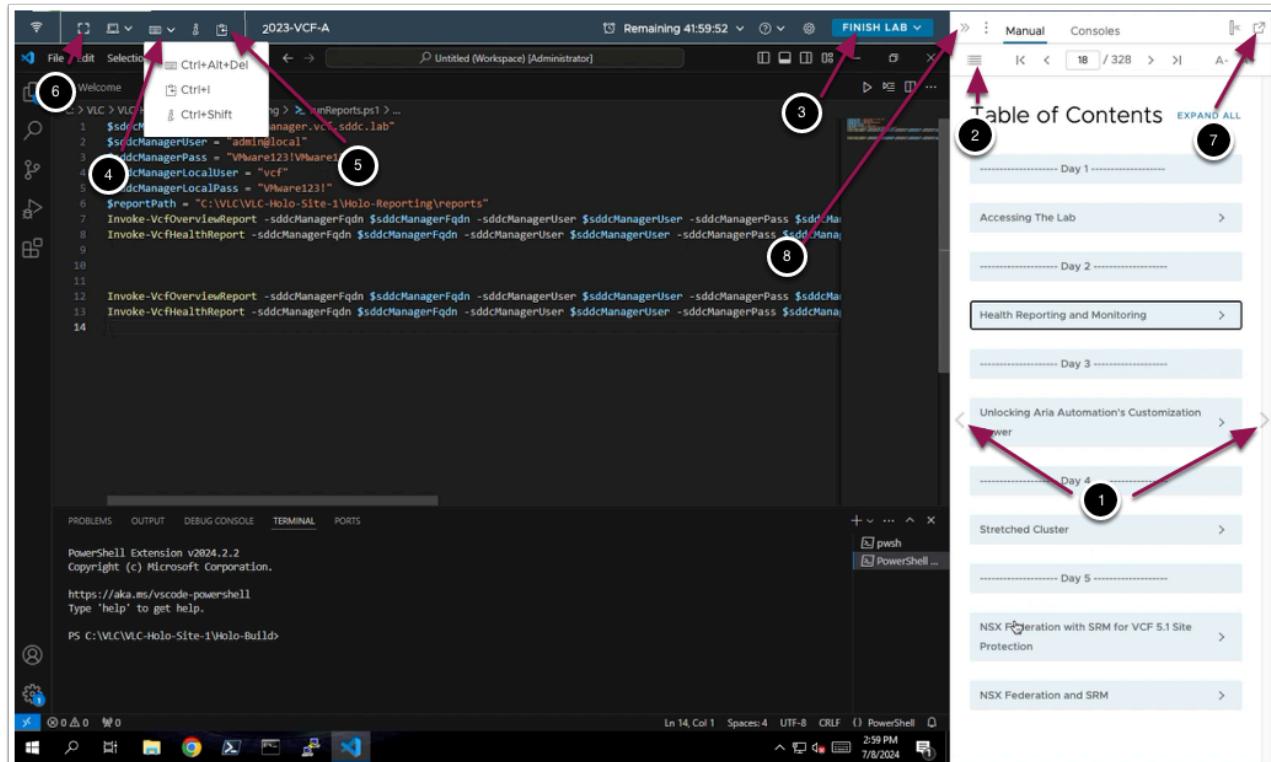
# Table of Contents

DevReady with Tanzu - TDD - Lab Manual .....	3
Lab Guidance .....	4
Pre-Deployment considerations: Sites, Network, Storage .....	8
Pre-Deployment considerations: Sites, Network, Storage, Permissions .....	9
Storage: Create a Storage Policy .....	14
Deployment: Developer Ready Infrastructure .....	29
Deploy a Supervisor .....	30
Install Supervisor Services.....	43
Create and Configure a Namespace.....	105
Deploy a TKG Workload Cluster .....	120
K8s Tools for IaaS Control Pane Management - CLI .....	135
Log in to Supervisor via CLI .....	152
Deploy a TKG Workload Cluster via CLI .....	158
Manage a TKG Workload Cluster.....	172
Packaging in a TKG Workload Cluster.....	181
Set the Image Registry .....	191
Configure Access to Developers and Deploy a Workload .....	203
Post- Deployment: Troubleshooting.....	215
Post Deployment Basic Troubleshooting.....	216

# **DevReady with Tanzu - TDD - Lab Manual**

# Lab Guidance

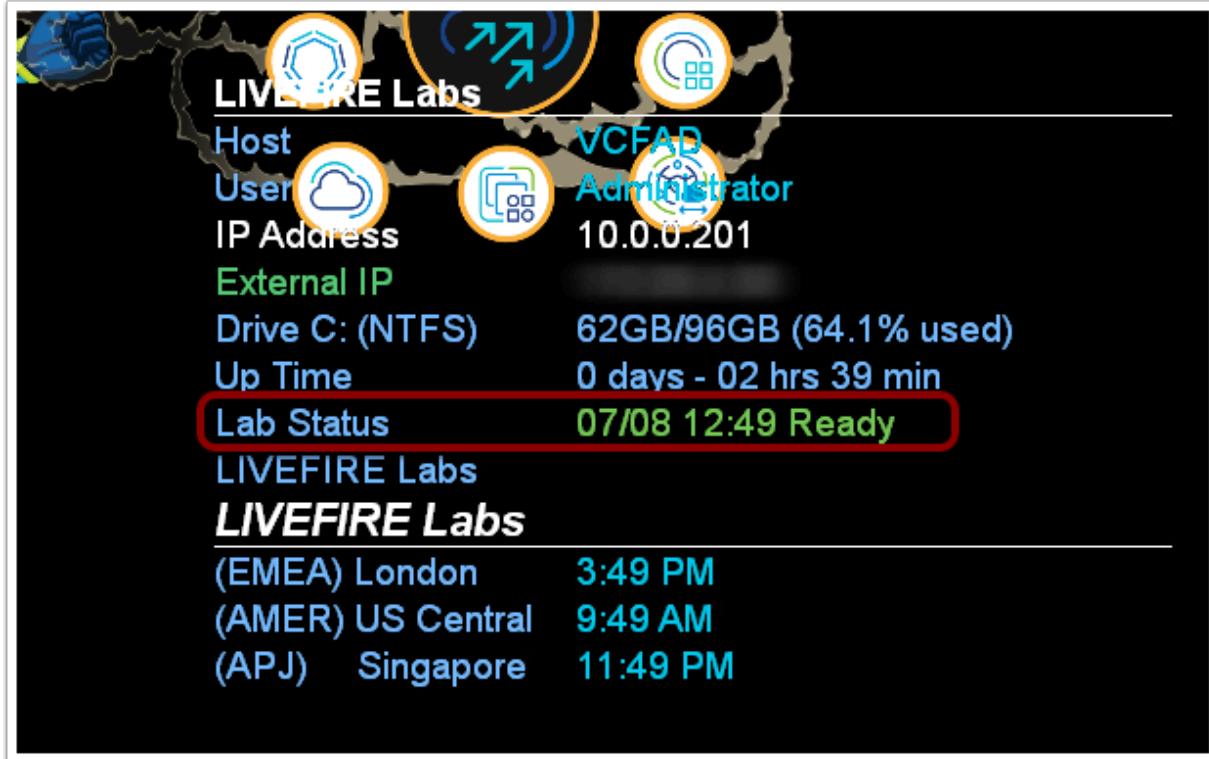
Welcome! To start somewhere other than the beginning, use the Table of Contents in the upper right-hand corner of the Lab Manual or click on one of the modules below.



From here you can:

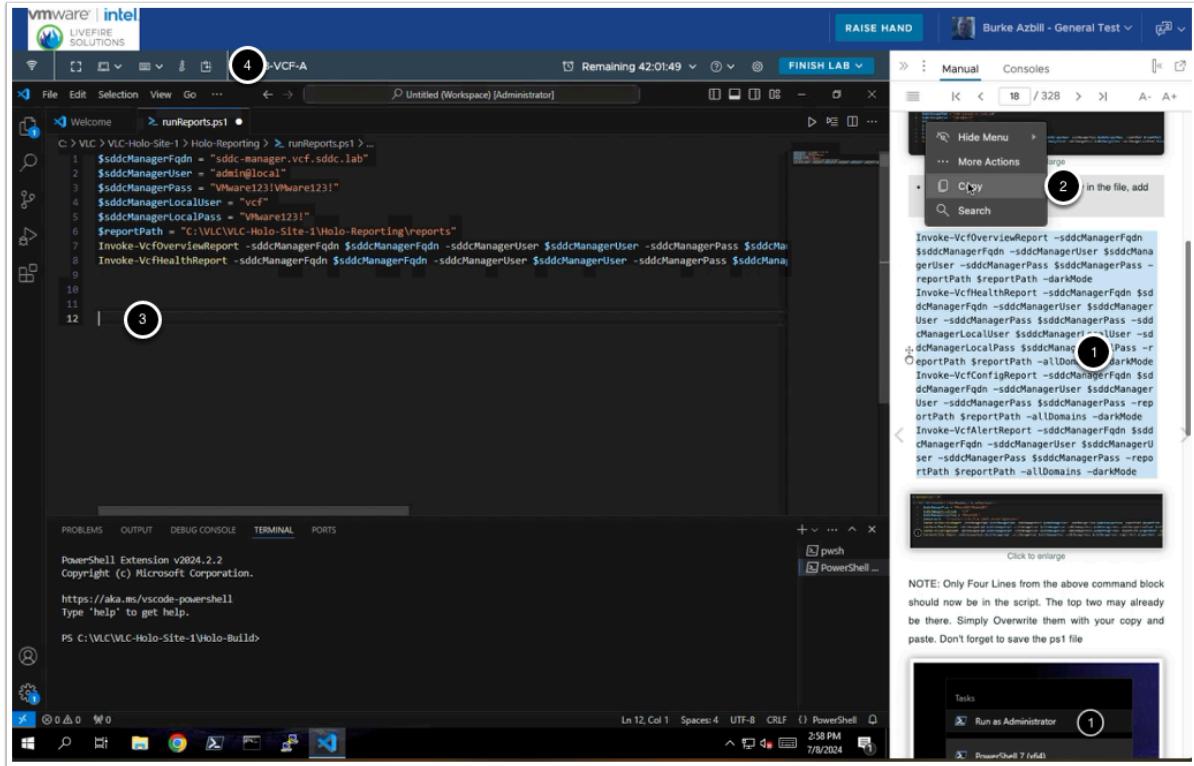
1. Click to advance to the next page and continue with the next lab module
2. Open the **TABLE OF CONTENTS** to jump to any module or lesson in this lab manual
3. Exit your lab and come back and resume it again in the future
4. Send Ctrl+Alt+Del to the console
5. Send Text into (Paste) the console
6. Enter Full-Screen
7. Detach the Lab Manual
8. Collapse the Lab Manual

## You are ready....is your lab?

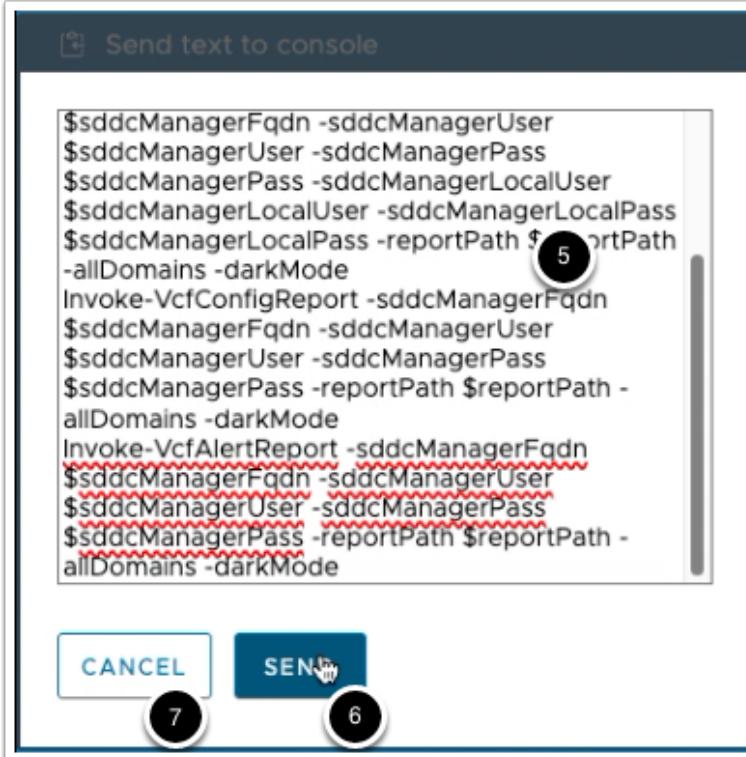


The lab console will indicate when your lab has finished all the startup routines and is ready for you to start. If you see anything other than "Ready", please wait for the status to update. If after 5 minutes your lab has not changed to "Ready", please ask for assistance.

# How to Copy from manual and Paste into Console



1. Use your mouse to select the text you wish to copy
2. When you release the mouse button to complete your selection, a context menu shows up - Click Copy  
NOTE: Right click on selected text DOES NOT bring up a copy option and CTRL+C/CMD+C does not work
3. Place your cursor/prompt at the location you wish to paste inside the console
4. Click the Clipboard icon to bring up the "Send text to console" window



5. Paste the contents of your clipboard into the window using Right-Click -> Paste, or your OS Keyboard combination of either CTRL+V or CMD+V
6. Click the "SEND" button to send the text to the console
7. Click the "CANCEL" button when you are done sending text to the console. This will close the popup window

```

C:\ > VLC > VLC-Holo-Site-1 > Holo-Reporting > runReports.ps1 > ...
1
2
3
4
5
6  'reports"
7  '$fqdn -ssdcManagerUser $ssdcManagerUser -ssdcManagerPass $ssdcManagerPass -reportPath $reportPath -darkMode
8  '$dn -ssdcManagerUser $ssdcManagerUser -ssdcManagerPass $ssdcManagerPass -ssdcManagerLocalUser $ssdcManagerLocalUser -s
9
10
11
12  '$fqdn -ssdcManagerUser $ssdcManagerUser -ssdcManagerPass $ssdcManagerPass -reportPath $reportPath -darkMode
13  '$dn -ssdcManagerUser $ssdcManagerUser -ssdcManagerPass $ssdcManagerPass -ssdcManagerLocalUser $ssdcManagerLocalUser -s
14  '$dn -ssdcManagerUser $ssdcManagerUser -ssdcManagerPass $ssdcManagerPass -reportPath $reportPath -allDomains -darkMode
15  'In -ssdcManagerUser $ssdcManagerUser -ssdcManagerPass $ssdcManagerPass -reportPath $reportPath -allDomains -darkMode

```

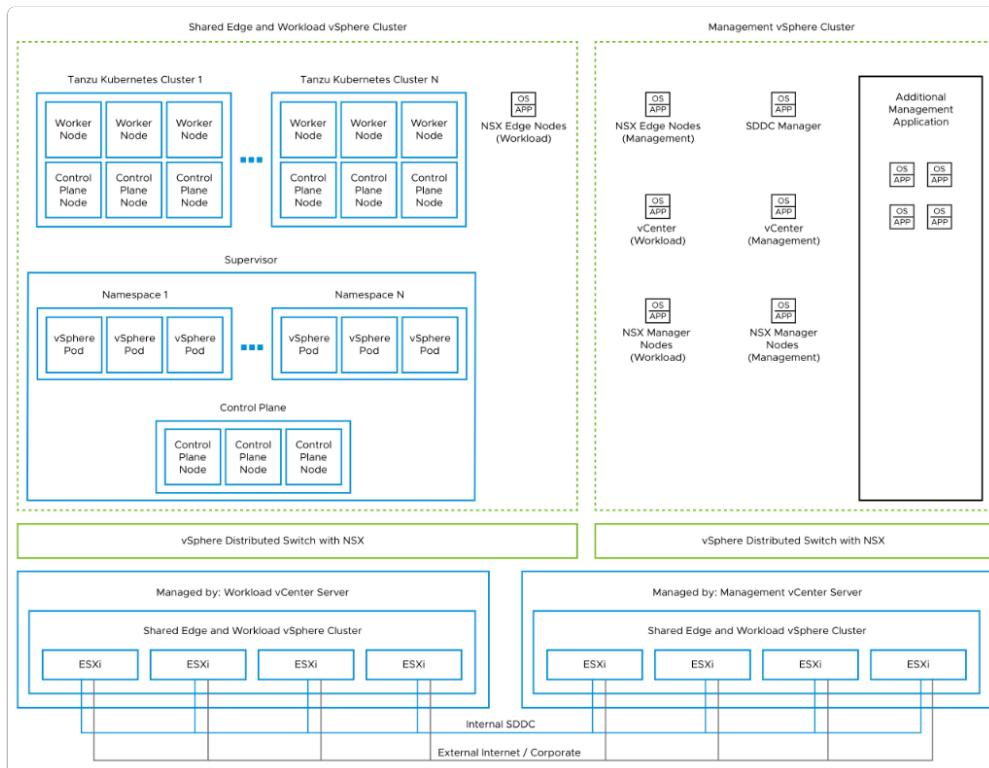
8. Confirm that your pasted text is at the desired location

# **Pre-Deployment considerations: Sites, Network, Storage**

# Pre-Deployment considerations: Sites, Network, Storage, Permissions

**i** Taken Directly from the **VMware Validated Solution: Network Design for Developer Ready Infrastructure for VMware Cloud Foundation** found at

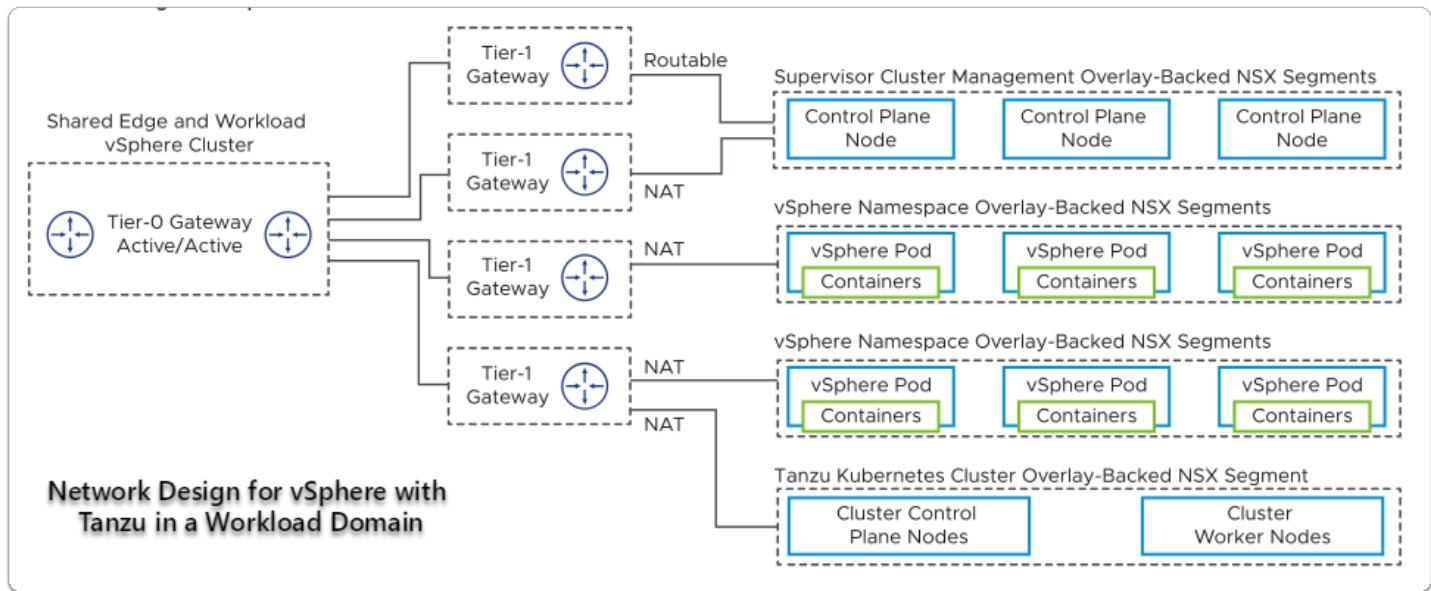
<https://docs.vmware.com/en/VMware-Cloud-Foundation/services/vcf-developer-ready-infrastructure-v1/GUID-AF178A31-D09A-4265-89FD-5987D1B36757.html>



## NETWORK Considerations:

vSphere with Tanzu requires multiple networks. This section discusses networking design not covered in the NSX detailed design.

You deploy all vSphere with Tanzu workloads to overlay-backed NSX segments. NSX Edge nodes in the shared edge and workload vSphere cluster are deployed to VLAN-backed portgroups.



## Networks Used by vSphere with Tanzu

Network	Routable/NAT	Usage
Supervisor Cluster Management Network	Routable	Used by the Supervisor control plane nodes.
Namespace Networks	NAT	When you create a namespace, a /28 overlay-backed NSX segment and corresponding IP pool is instantiated to service pods in that namespace. If that IP space runs out, an additional /28 overlay-backed NSX segment and IP pool are instantiated.
Service IP Pool Network	NAT	Used by Kubernetes applications that need a service IP address.
Ingress IP Pool Network	Routable	Used by NSX to create an IP pool for load balancing.
Egress IP Pool Network	Routable	Used by NSX to create an IP pool for NAT endpoint use.
Pod Networks	NAT	Used by Kubernetes pods that run in the cluster. Any

Network	Routable/NAT	Usage
Tanzu Kubernetes Cluster Service Pool Network		Tanzu Kubernetes Clusters instantiated in the Supervisor also use this pool.
Tanzu Kubernetes Cluster Service Pool Network	NAT	When you create a Tanzu Kubernetes cluster, an NSX Tier-1 Gateway is instantiated in NSX. On that NSX Tier-1 Gateway, a /28 overlay-backed NSX segment and IP pool is also instantiated.

 [Design Decisions on Networking for Developer Ready Infrastructure for VMware Cloud Foundation](#)

Design Decision	Design Justification	Design Implication
Add a /24 overlay-backed NSX segment for use by the Supervisor control plane nodes.	Supports the Supervisor control plane nodes.	You must create the overlay-backed NSX segment.
Use a dedicated /20 subnet for pod networking.	A single /20 subnet is sufficient to meet the design requirement of 2000 pods.	Private IP space behind a NAT that you can use in multiple Supervisors.
Use a dedicated /22 subnet for services.	A single /22 subnet is sufficient to meet the design requirement of 2000 pods	Private IP space behind a NAT that you can use in multiple Supervisors.
Use a dedicated /24 or larger subnet on your corporate network for ingress endpoints.	A /24 subnet is sufficient to meet the design requirement of 2000 pods in most cases.	This subnet must be routable to the rest of the corporate network.  A /24 subnet will suffice for most use cases, but you should evaluate your ingress needs prior to deployment
Use a dedicated /24 or larger subnet on your corporate network for egress endpoints.	A /24 subnet is sufficient to meet the design requirement of 2000 pods in most cases.	This subnet must be routable to the rest of the corporate network.

Design Decision	Design Justification	Design Implication
		A /24 subnet will suffice for most use cases, but you should evaluate your egress needs prior to deployment

## STORAGE Considerations

You must configure a datastore with the activation requirements before activating a Supervisor. The Supervisor configuration requires the use of vSphere Storage Policy Based Management (SPBM) policies for control plane nodes, ephemeral disks, and image cache. These policies correlate to Kubernetes storage policies that can be assigned to vSphere Namespaces. These policies are consumed in a Supervisor or a Tanzu Kubernetes cluster, deployed by using Tanzu Kubernetes Grid Service in the Supervisor.

Design Decision	Design Justification	Design Implication
Create a vSphere tag and tag category, and apply the vSphere tag to the vSAN datastore in the shared edge and workload vSphere cluster in the VI workload domain.	Supervisor activation requires the use of vSphere Storage Based Policy Management (SPBM). To assign the vSAN datastore to the Supervisor, you need to create a vSphere tag and tag category to create an SPBM rule.	This must be done manually or via PowerCLI
Create a vSphere Storage Policy Based Management (SPBM) policy that specifies the vSphere tag you created for the Supervisor	When you create the SPBM policy and define the vSphere tag for the Supervisor, you can then assign that SPBM policy during Supervisor activation.	This must be done manually or via PowerCLI

## PERSONAS and PERMISSION Considerations:

Personas describe types of system users, aligned with real people and their functions within the organization. You build a persona set based on your organization requirements for role-based access control.

The following is an example of personas defined by the Developer Ready Infrastructure for VMware Cloud Foundation validated solution and their equivalent access. To delegate roles and define access based on roles and responsibilities within your organizational structure, you use these personas as a baseline for defining and building a set of personas.



## Example Personas for Developer Ready Infrastructure

Persona	Responsibility	Solution Component	Component Role or Group
vSphere Administrator	Full administrative access to vSphere with Tanzu infrastructure - configuring and activating Supervisors and vSphere namespaces	vCenter Server	Administrator
DevOps Engineer	Deploying vSphere Pods, VMs, and Tanzu Kubernetes clusters on existing vSphere namespaces within a Supervisor	vSphere namespace	Can edit
Auditor	Read-only access for security and compliance review	vSphere namespace	Can view

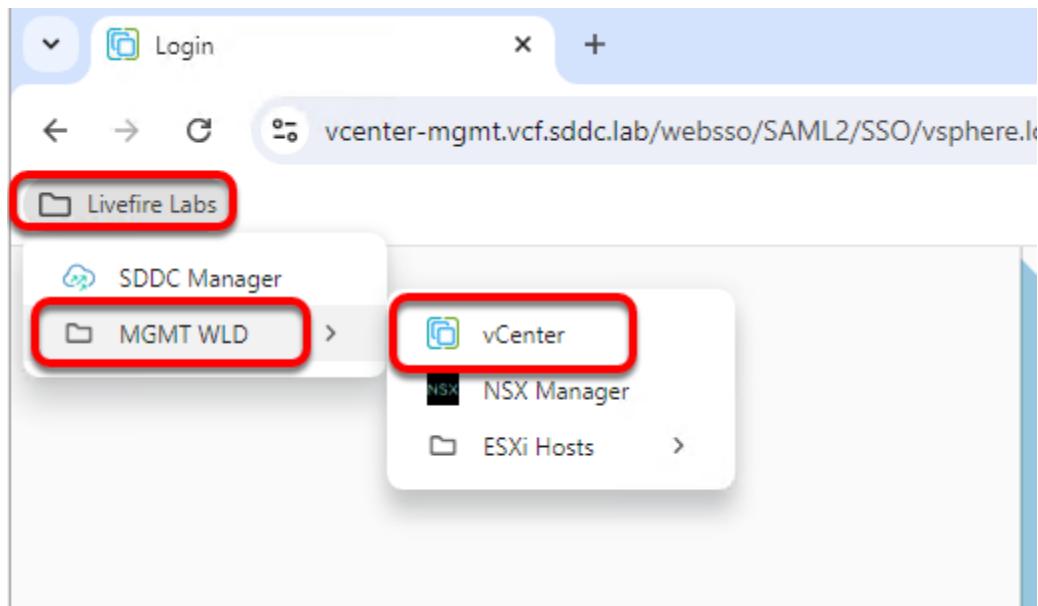
# Storage: Create a Storage Policy

## TASK DESCRIPTION AND OBJECTIVES

You need to create a storage policy before the Supervisor deployment. It is going to be used for Supervisor hosted Applications and Services.

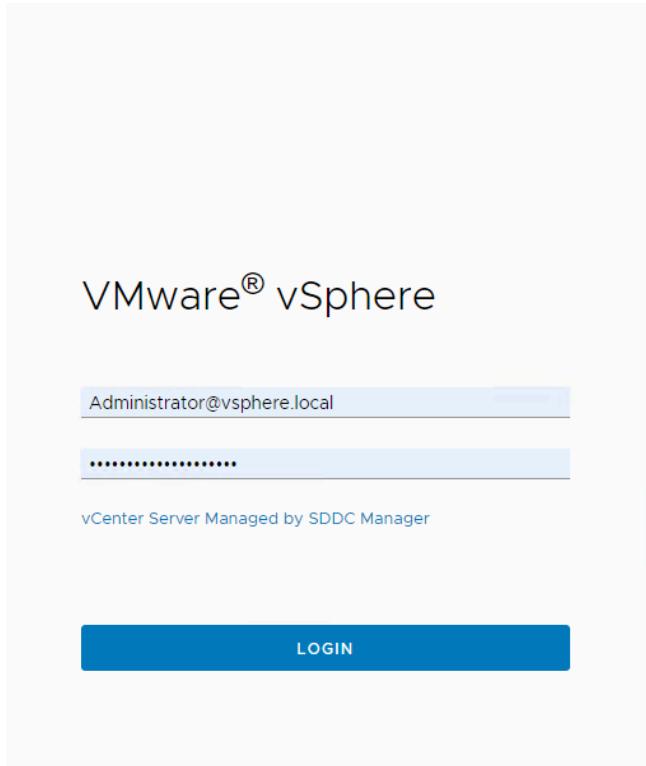
-  Before creating the **storage policy**, you need to create a custom **tag** and **category**, then tag your Datastore with the new tag. The policy then uses these tags as part of its usage rule.

## Open Chrome



- Open the Livefire Labs Bookmarks Folder
- Choose --> **MGMT WLD** --> **vCenter**

## User and password



Login Using  
User:

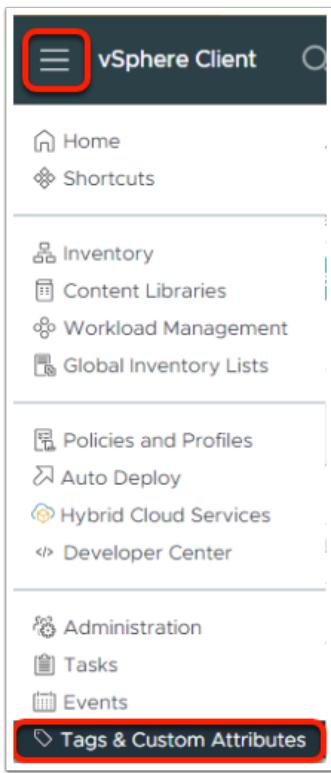
Administrator@vsphere.local

Password:

VMware123!VMware123!

## Open Tags and Custom Attriburtes

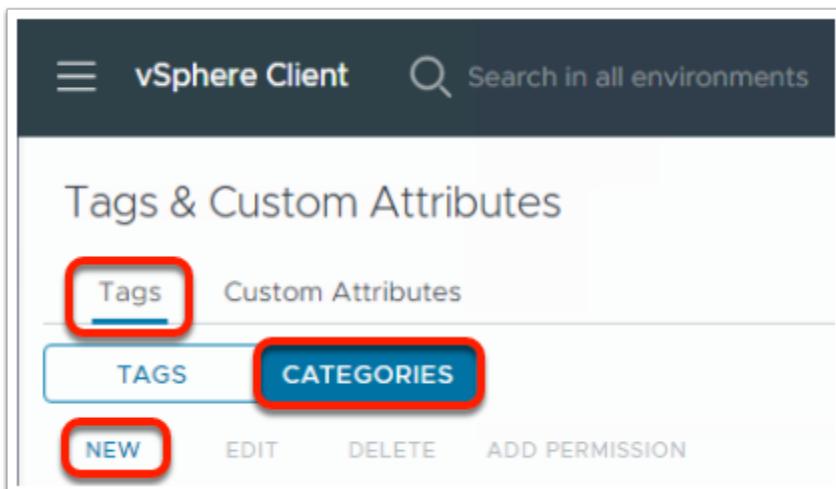
 Once logged into the vSphere client UI:



Click on the hamburger (ellipsis) menu in the top left corner

Select **Tags & Custom Attributes**

## Create a new Category:



Under **Tags & Custom Attributes**,

Select **Tags**, then **CATEGORIES** --> **NEW**

# Create Category

Create Category X

Category Name:  vsphere-with-tanzu-category

Description:

Tags Per Object:  One tag  Many tags

Associable Object Types:

<input type="checkbox"/> All objects	<input type="checkbox"/> Cluster
<input type="checkbox"/> Folder	<input checked="" type="checkbox"/> Datastore
<input type="checkbox"/> Datacenter	<input type="checkbox"/> Distributed Port Group
<input type="checkbox"/> Datastore Cluster	<input type="checkbox"/> Host
<input type="checkbox"/> Distributed Switch	<input type="checkbox"/> Library Item
<input type="checkbox"/> Content Library	<input type="checkbox"/> Resource Pool
<input type="checkbox"/> Network	<input type="checkbox"/> Virtual Machine
<input type="checkbox"/> vApp	

CANCEL CREATE

In the **Create Category** wizard enter as following:

**Category Name:**

vsphere-with-tanzu-category

- **Tags Per Object:** One Tag
- **Associable Object Types:** Datastore

Confirm with **CREATE**

# Confirm New Category

The screenshot shows the vSphere Client interface with the 'Tags & Custom Attributes' section open. The 'CATEGORIES' tab is selected. A table lists categories with columns for Category Name, Description, Multiple Cardinality, and Associable Entities. One row, 'vsphere-with-tanzu-category', is highlighted with a red box.

	Category Name	Description	Multiple Cardinality	Associable Entities
○	Application-Name	vcs-uncategorized-tag-d escription	true	Distributed Switch, VMware Distributed Switch, Distributed Port Group, Virtual Machine
○	OC-Deployment	vcs-uncategorized-tag-d escription	true	Distributed Switch, VMware Distributed Switch, Distributed Port Group, Virtual Machine
○	Application-Tier	vcs-uncategorized-tag-d escription	true	Distributed Switch, VMware Distributed Switch, Distributed Port Group, Virtual Machine
○	vsphere-with-tanzu-category		false	Datastore

You will see your Category created.

Let's continue with creating a **Tag** that is bound under this **Category**.

## Create Tag

The screenshot shows the vSphere Client interface with the 'Tags & Custom Attributes' section open. The 'Tags' tab is selected. The 'TAGS' button is highlighted with a red box. The 'NEW' button is also highlighted with a red box.

Under **Tags & Custom Attributes**,

Select **Tags**, then again **TAGS** --> **NEW**

## Create Tag

X

Name:

vsphere-with-tanzu-tag

Description:

Category:

vsphere-with-tanzu-category

[Create New Category](#)

CANCEL

CREATE

In the **Create Tag** wizard enter as following:

**Name:**

vsphere-with-tanzu-tag

Category: vsphere-with-tanzu-category (from the drop-down)

Confirm with **CREATE**

## Assign Tag

Tags & Custom Attributes

Tags      Custom Attributes

TAGS      CATEGORIES

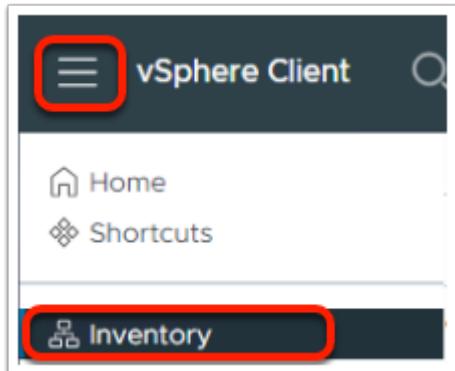
NEW      EDIT      DELETE      ADD PERMISSION

Tag Name	Category	Description
oc-lab-test	Application-Name	oc-lab-test
Test-Deploy-AC	Application-Name	Test-Deploy-AC
vsphere-with-tanzu-tag	vsphere-with-tanzu-category	
Apache	Application-Tier	Apache
5d6944aa-f68f-4207-a2dd-dd9f369b0a76	OC Deployment	5d6944aa-f68f-4207-a2dd-dd9f369b0a76
756a8239-9f10-4791-bb03-307652281ba3	OC Deployment	756a8239-9f10-4791-bb03-307652281ba3

You will see your Tag created and related to the specified Category you created earlier.

Let's continue with applying this Tag on the vSAN Datastore

## Assign Tag



Click on the hamburger (ellipsis) menu in the top left corner

Select Inventory

A screenshot of the vSphere Client showing the properties of the "vcf-vsang" Datastore. The left sidebar shows the navigation path: "vcf-vsang" under "vcf-vsan" which is under "vcf-vsan" in "vcf-vsan" under "vcf-vsan" under "vcf-vsan". The "Actions" tab is selected. In the center, the "Properties" section is visible. At the bottom right of the properties section, there is a button labeled "Assign Tag..." with a blue heart icon. This button is highlighted with a red box. Other buttons in the same row include "Remove Tag" and "Edit Custom Attributes...".

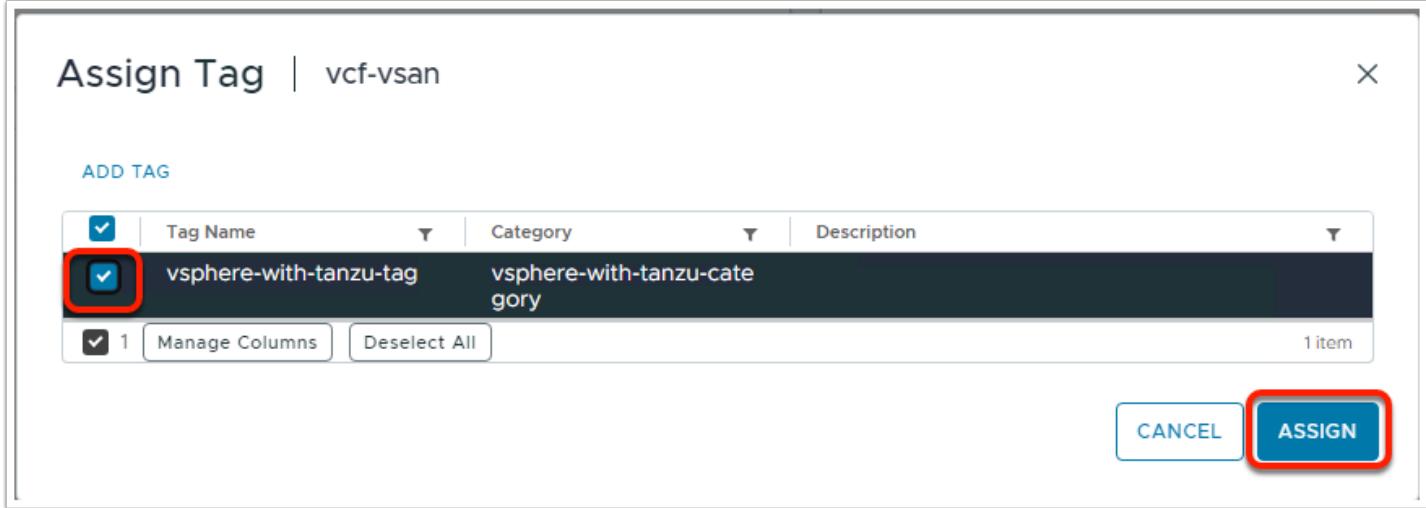
In the Inventory menu:

Select the **Storage** section, Expand the vCenter, Expand the Datacenter

Select the **vcf-vsang** Datastore, right click on it and select **Tags & Custom Attributes**

Expand it and select **Assign Tag**

# Assign Tag



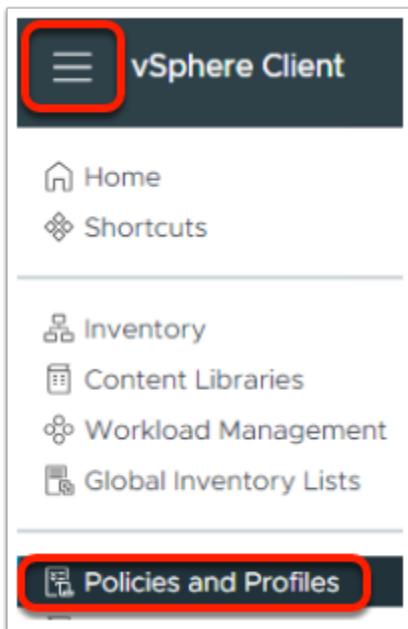
In the **Assign Tag wizard**

Select the **vsphere-with-tanzu-tag** you created earlier

Confirm with **ASSIGN**

 Now you can configure a relevant **storage policy**, working with **tag based placement** of the objects, over the **vcf-vsang** Datastore.

# Create Storage Policy



Click on the hamburger menu in the top left corner

Select **Policies and Profiles**

# Create Storage Policy

A screenshot of the "VM Storage Policies" screen in the vSphere Client. On the left, there is a sidebar with a red box around the "VM Storage Policies" link. The main area shows a table with several policy components listed. At the top right of the main area, there is a "CREATE" button which is also highlighted with a red box. Other buttons visible include CHECK, REAPPLY, EDIT, CLONE, and DELETE. A "Quick Filter" input field is present. The table rows are as follows:

<input type="checkbox"/>	Name
<input type="checkbox"/>	VM Encryption Policy
<input type="checkbox"/>	vSAN Default Storage Policy
<input type="checkbox"/>	VVol No Requirements Policy
<input type="checkbox"/>	Management Storage Policy - Regular

Under **VM Storage Policies**

Select **VM Storage Policies** --> **CREATE**

# Create Storage Policy

The screenshot shows the 'Create VM Storage Policy' wizard. The left sidebar lists steps: 1. Name and description (highlighted with a red box), 2. Policy structure, 3. Storage compatibility, and 4. Review and finish. The main panel is titled 'Name and description'. It shows 'vCenter Server: VCENTER-MGMT.VCF.SDDC.LAB' and a 'Name:' field containing 'vsan-tanzu-storage' (also highlighted with a red box). A 'Description:' field is empty. At the bottom right are 'CANCEL' and 'NEXT' buttons, with 'NEXT' also highlighted with a red box.

In the **Create VM Storage Policy** wizard,

For **Name and description** enter as following:

```
vsan-tanzu-storage
```

Confirm with **NEXT**

# Create Storage Policy

Create VM Storage Policy

Policy structure

Host based services

Create rules for data services provided by hosts. Available data services could include encryption, I/O control, caching, etc. Host based services will be applied in addition to any datastore specific rules.

Enable host based rules

Datastore specific rules

Create rules for a specific storage type to configure data services provided by the datastores. The rules will be applied when VMs are placed on the specific storage type.

Enable rules for "vSAN" storage

Enable rules for "vSANDirect" storage

Enable rules for "VMFS" storage

Enable tag based placement rules

Tanzu on vSphere Storage topology

Create a Tanzu rule for storage topology that will be applied to all other datastores. Specific rules in this step will be applied to all other datastores. Specific rules in this step will be applied to all other datastores.

CANCEL BACK NEXT

For **Policy structure**, under **Datastore specific rules**,

Select **Enable tag based placement rules**

Confirm with **NEXT**

# Create Storage Policy

Create VM Storage Policy

Tag based placement

Add tag rules to filter datastores to be used for placement of VMs.

Rule 1 REMOVE

Tag category: vsphere-with-tanzu-category

Usage option: Use storage tagged with

Tags: BROWSE TAGS

ADD TAG RULE

X

1 Name and description

2 Policy structure

**3 Tag based placement**

4 Storage compatibility

5 Review and finish

CANCEL BACK NEXT

For **Tag Based Placement**, under **Rule 1**,

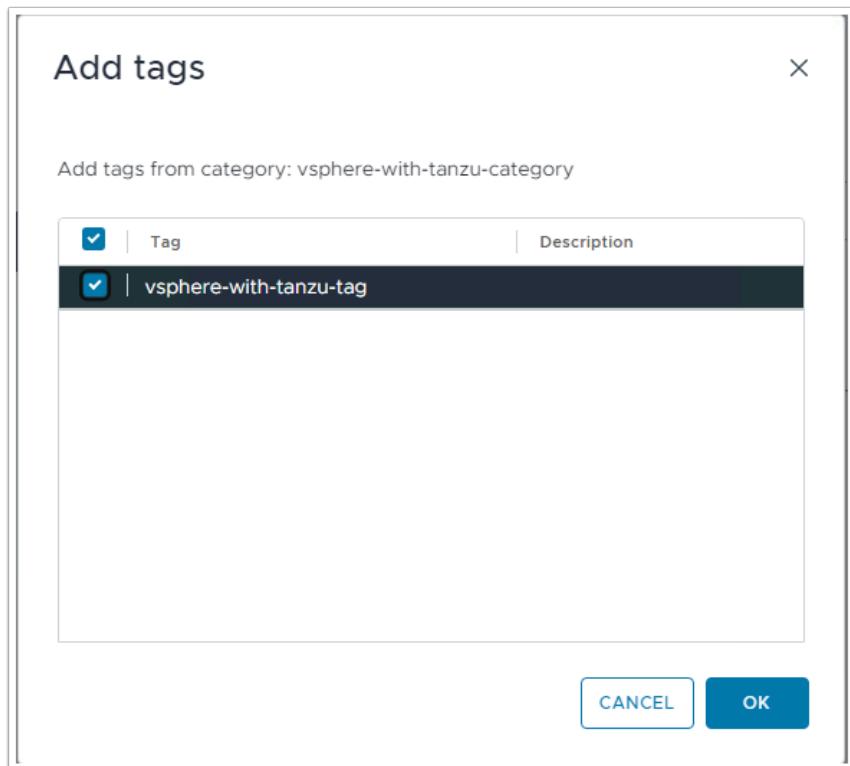
Enter the following:

Tag category: vsphere-with-tanzu-category

Usage option: Use storage tagged with

Tags: vsphere-with-tanzu-tag

You can select the **tanzu-tag** clicking on **BROWSE TAGS**



In the **Add tags** wizzard,

Select your **vsphere-with-tanzu-tag** and confirm with **OK**

## Create Storage Policy

The screenshot shows the "Create VM Storage Policy" wizard, specifically step 3: "Tag based placement". On the left, a sidebar lists steps 1 through 5. Step 3, "Tag based placement", is selected and highlighted with a dark blue background. The main area shows a "Rule 1" configuration. Under "Tag category", the value "vsphere-with-tanzu-category" is entered. Under "Usage option", it says "Use storage tagged with". Under "Tags", the value "vsphere-with-tanzu-tag" is listed with a delete icon. Below these fields is a "BROWSE TAGS" button. At the bottom of the form is an "ADD TAG RULE" button. At the very bottom right are three buttons: "CANCEL", "BACK", and "NEXT", with "NEXT" being highlighted with a red box.

You can see and the needed values for Tag Category and Tags are in place.

Confirm with **NEXT**

## Create Storage Policy

The screenshot shows the 'Create VM Storage Policy' wizard at step 4: 'Storage compatibility'. The 'COMPATIBLE' tab is selected. A table lists a single datastore: 'vcf-vsan' (mgmt-datacenter-01, vSAN, 2.15 TB free, 3.52 TB capacity). The 'vcf-vsan' entry is highlighted with a red box. At the bottom right, the 'NEXT' button is highlighted with a red box.

Name	Datacenter	Type	Free Space	Capacity	Warnings
vcf-vsan	mgmt-datacenter-01	vSAN	2.15 TB	3.52 TB	

**Storage compatibility**

**COMPATIBLE** **INCOMPATIBLE**

Expand datastore clusters

Compatible storage 3.52 TB (2.15 TB free)

Quick Filter  Enter value

Name Datacenter Type Free Space Capacity Warnings

vcf-vsan mgmt-datacenter-01 vSAN 2.15 TB 3.52 TB

Manage Columns 1 item

CANCEL BACK NEXT

For **Storage Compatibility**, under **COMPATIBLE**,

you can see the listed **vcf-vsan** Datastore.

Confirm with **NEXT**

# Create Storage Policy

The screenshot shows a software interface for creating a storage policy. On the left, a vertical navigation bar lists five steps: 1. Name and description, 2. Policy structure, 3. Tag based placement, 4. Storage compatibility, and 5. Review and finish. Step 5 is highlighted with a dark background and white text. The main panel is titled "Review and finish". It contains two sections: "General" and "Tag based placement". Under "General", the "Name" field is set to "vsan-tanzu-storage". Under "Tag based placement", it shows "Tagged with: vsphere-with-tanzu-tag" and "tanzu-category". At the bottom right of the main panel are three buttons: "CANCEL", "BACK", and "FINISH", with "FINISH" being the only one highlighted with a red border.

Create VM Storage Policy

Review and finish

General

Name	vsan-tanzu-storage
Description	vCenter Server
vCenter Server	vcenter-mgmt.vcf.sddc.lab

Tag based placement

Tagged with:	vsphere-with-tanzu-tag
tanzu-category	

X

1 Name and description

2 Policy structure

3 Tag based placement

4 Storage compatibility

5 Review and finish

CANCEL

BACK

**FINISH**

For Review and finish you can see the configured properties in your new storage policy.

Confirm with **FINISH**

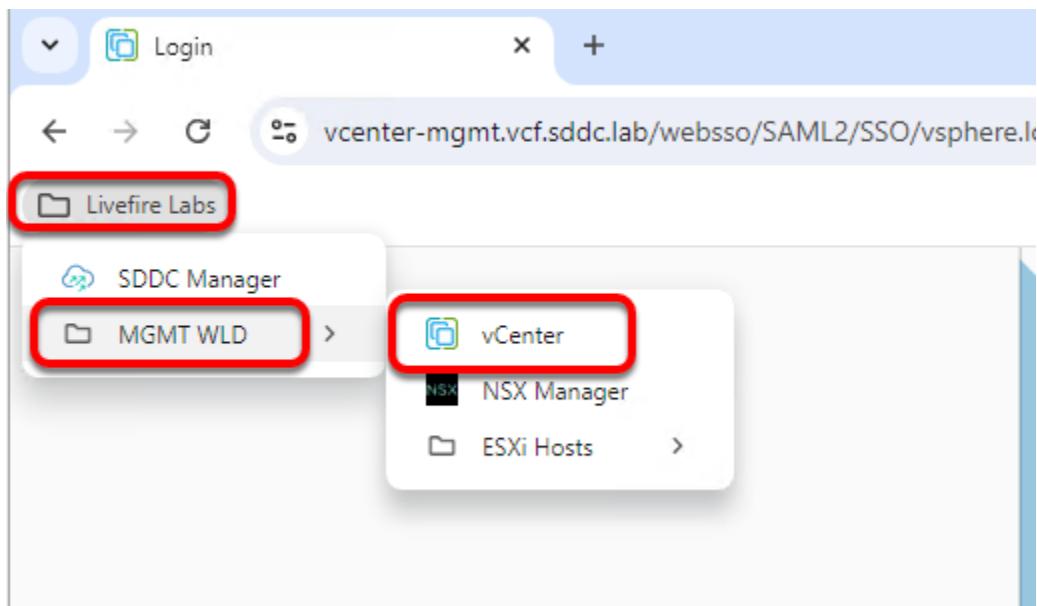
# **Deployment: Developer Ready Infrastructure**

# Deploy a Supervisor

## Task description and objectives

After you meet the prerequisites, you can begin the deployment of the vSphere with Tanzu environment to support the Developer Ready Infrastructure for VMware Cloud Foundation solution. The deployment of vSphere with Tanzu involves deploying and configuring an **IaaS Control Plane Supervisor Cluster**.

## Login to vSphere

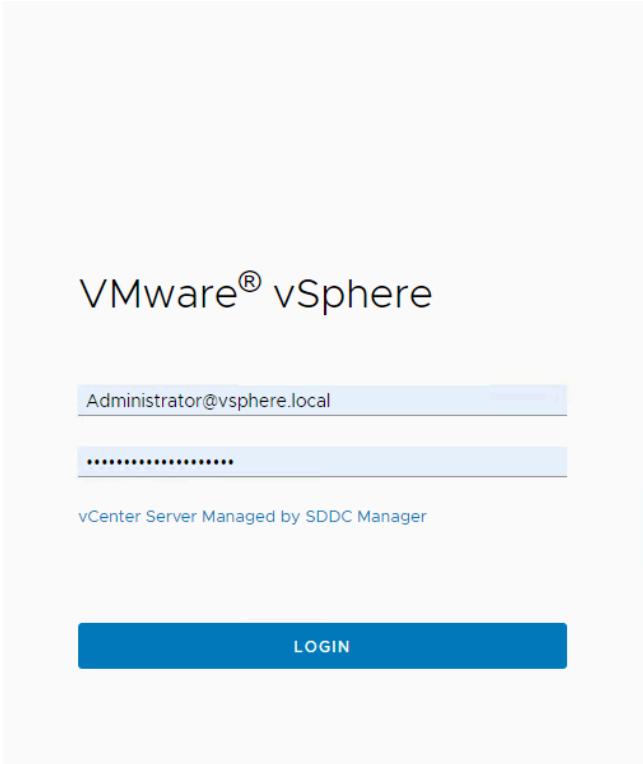


Open Chrome

Open the Livefire Labs Bookmark Folder

Choose --> **MGMT WLD** --> **vCenter**

# Login to vSphere



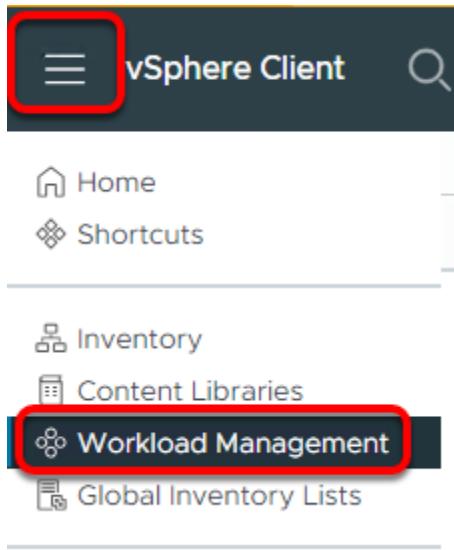
Login with User:

Administrator@vsphere.local

And Password:

VMware123!VMware123!

# Open Workload Management



Open the **Hamberger** Menu

Choose **Workload Management**

A screenshot of the "Workload Management" page within the vSphere Client. The title "Workload Management" is at the top. Below it is a descriptive text block: "Workload Management enables deploying and managing Kubernetes workloads in vSphere. By using Workload Management, you can leverage both Kubernetes and vSphere functionality. Once you configure a vSphere cluster for Workload Management and it becomes a Supervisor, you can create namespaces that provide compute, networking, and storage resources for running your Kubernetes applications. You can also configure Supervisors with policies for resource consumption." A blue "GET STARTED" button is located below this text, with a red rectangle highlighting it. To the right of the text is a small illustration of two people standing on a platform, looking at a large server or network diagram. The interface includes a sidebar on the left with a "Namespaces" section showing a "No items found" message and a "GET STARTED" button. The top right corner shows the user "Administrator@VSPHERE.LOCAL" and other standard interface icons.

Click **GET STARTED**

# vCenter Server and Network

1. vCenter Server and Network Select a vCenter Server system and a network to set up a Supervisor.

Select the vCenter Server system that will host this Supervisor.

Select a vCenter Server system  VCENTER-MGMT.VCF.SDDC.LAB (SUPPORTS NSX) 

Select the networking stack that will provide connectivity to this Supervisor.

Select a networking stack  NSX  vSphere Distributed Switch (VDS)

**NEXT**

2. Supervisor location	Deploy the Supervisor on a group of vSphere Zones or a vSphere cluster
3. Storage	Select the storage policy for the control plane VMs on this Supervisor.
4. Management Network	Configure networking for the control plane VMs on this Supervisor.
5. Review and Confirm	Review and confirm all details and default settings to start this Supervisor setup

Make Sure the **NSX** Radio button is chosen

Click **NEXT**

**!** No idea why the UI, on occasion, seems to post this step again after you click **NEXT**

# Supervisor Location

This vSphere cluster will be set up as a Supervisor. Select a vSphere cluster with enough space to support your Kubernetes workloads.

Cluster Name	VSphere Zone	Number of Hosts	Available CPU	Available Memory
mgmt-cluster-01	-	4	84.27 GHz	234.23 GB

A vSphere Zone will be automatically created and assigned to the vSphere cluster that you select. If you don't provide a vSphere Zone name, one will be automatically generated. You cannot change vSphere Zone name once it is set.

vSphere Zone name: lf-tdd-supervisor

NEXT

1. Click **CLUSTER DEPLOYMENT**

2. Give the Supervisor Cluster a name, i.e.:

lf-tdd-supervisor

3. Click the Radio button for the one VCF Cluster

4. Click **NEXT**

# Storage

## Workload Management

The screenshot shows the 'Workload Management' configuration interface. The top navigation bar includes 'BACK', 'IMPORT CONFIG', and 'VIEW PREREQUISITES'. The main area displays two completed steps: 'vCenter Server and Network' (vCenter Server Network: vcenter-mgmt.vcf.sddc.lab, NSX) and 'Supervisor location' (Supervisor Name Cluster: lf-tdd-supervisor, mgmt-cluster-01, vSphere Zone Name: --). Step 3, 'Storage', is currently selected. A sub-instruction at the top of this step says: 'Select the storage policy for the control plane VMs on this Supervisor.' Below this, a note states: 'Select a storage policy to be used for datastore placement of Supervisor control plane VMs. The policy is associated with one or more datastores on the vSphere environment.' Three dropdown menus are shown under 'Storage Policy': 'Control Plane Storage Policy' (vsan-tanzu-storage), 'Ephemeral Disks Storage Policy' (vsan-tanzu-storage), and 'Image Cache Storage Policy' (vsan-tanzu-storage). The 'NEXT' button is highlighted with a red box. The bottom section lists steps 4 ('Management Network') and 5 ('Workload Network') with their respective descriptions.

Choose the single Storage Policy *vsan-tanzu-storage* for each Storage Policy setting

Click **NEXT**

# Management Network

v 4. Management Network Configure networking for the control plane VMs on this Supervisor.

A Supervisor contains three control plane VMs. Each Supervisor sits on a management network that supports traffic to vCenter Server.

[VIEW NETWORK TOPOLOGY](#)

Network Mode <small>(i)</small>	Static
Network <small>(i)</small>	sddc-vds01-vmmgmt
Starting IP Address <small>(i)</small>	10.0.11.180
Subnet Mask <small>(i)</small>	255.255.255.0
Gateway <small>(i)</small>	10.0.11.253
DNS Server(s) <small>(i)</small>	10.0.0.253
DNS Search Domain(s) <small>(i)</small>	vcf.sddc.lab <small>Optional</small>
NTP Server(s) <small>(i)</small>	10.0.0.253

**NEXT**

**Network Model:** Static

**Network:** sddc-vds01-vmmgmt

**Starting IP Address:**

10.0.11.180

**Subnet Mask:**

255.255.255.0

**Gateway:**

10.0.11.253

**DNS:**

10.0.0.253

**DNS Search Domain(s):**

vcf.sddc.lab

**NTP:**

10.0.0.253

# Workload Network

5. Workload Network

Configure networking to support traffic to the Kubernetes API and to workloads and services.

The Workload Network supports traffic to the Kubernetes API and to the Workloads/Services that are deployed on the Supervisor. This network is supported by NSX.

[VIEW NETWORK TOPOLOGY](#)

vSphere Distributed Switch	mgmt-vds01	Edge Cluster	EC-01
DNS Server(s)	10.0.0.253	Tier-0 Gateway	VLC-Tier-0
NAT Mode	<input checked="" type="checkbox"/> Enabled	Subnet Prefix	/28
Namespace Network	10.244.0.0/20	Service CIDR	10.96.0.0/23 <small>This field cannot be edited later once saved. Make sure all CIDR values are unique.</small>
Ingress CIDRs	10.80.0.0/24	Egress CIDRs	10.70.0.0/24

**NEXT**

## vSphere Distributed Switch:

mgmt-vds01

## DNS Server(s):

10.0.0.253

## NAT Mode:

Enabled

## Edge Cluster:

EC-01

## Tier-0 Gateway:

VLC-Tier-0

## Ingress CIDRs:

10.80.0.0/24

## Egress CIDRs:

10.70.0.0/24

Workload Network	
vDS	mgmt-vds01
Edge Cluster	EC-01
DNS Server(s)	10.0.0.253
Namespace Network	10.244.0.0/20
Service CIDR	10.96.0.0/23
Ingress CIDRs	10.80.0.0/24
Egress CIDRs	10.70.0.0/24
Tier-0 Gateway	4693e4af-50c7-47a9-95a4-35e43015f05d
NAT Mode	Enabled
Subnet Prefix	28

Click **NEXT**

## Review and Confirm

6. Review and Confirm

Review and confirm all details and default settings to start this Supervisor setup

Advanced Settings

Supervisor Control Plane Size [\(i\)](#) Small (CPUs: 4, Memory: 16 GB, Storage: 32 GB) [\(i\)](#)  
You can edit this default setting

API Server DNS Name(s) [\(i\)](#) kubeapi.vcf.sddc.lab [\(i\)](#)  
Optional

Review and confirm the steps above. Click Finish to start setting up mgmt-cluster-01 as a Supervisor.  
You can view these configuration details in the Supervisor view under the Configure tab.

Export configuration [\(i\)](#)

**FINISH**

### Supervisor Control Plane Size:

Small

### API Server DNS Name(s):

kubeapi.vcf.sddc.lab

Click **FINISH**

## Track Status

As the Supervisor deploys there are various places to track progress.

Under **Workload Management --> Supervisors Tab --> Config Status (View)**

Workload Management

Namespaces Supervisors Services Updates

ADD SUPERVISOR DEACTIVATE CLONE CONFIG EXPORT CONFIG EXPORT LOGS RESTORE

Quick Filter Enter value

	Supervisor	↑	Namespaces	Hosts	Services	Config Status	Host Config Status	Control Plane Node Address	CPU for namespaces
○	If-tdd-supervisor	⚠	0	4	--	Configuring (view)	N/A		80.241 GHz

Manage Columns 1 item

### Configuring If-tdd-supervisor

Supervisor configuration follows a desired state in which it will keep retrying to reach a condition. Conditions can transition from a configured state to configuring based on dependencies. Monitor for any errors that require attention.

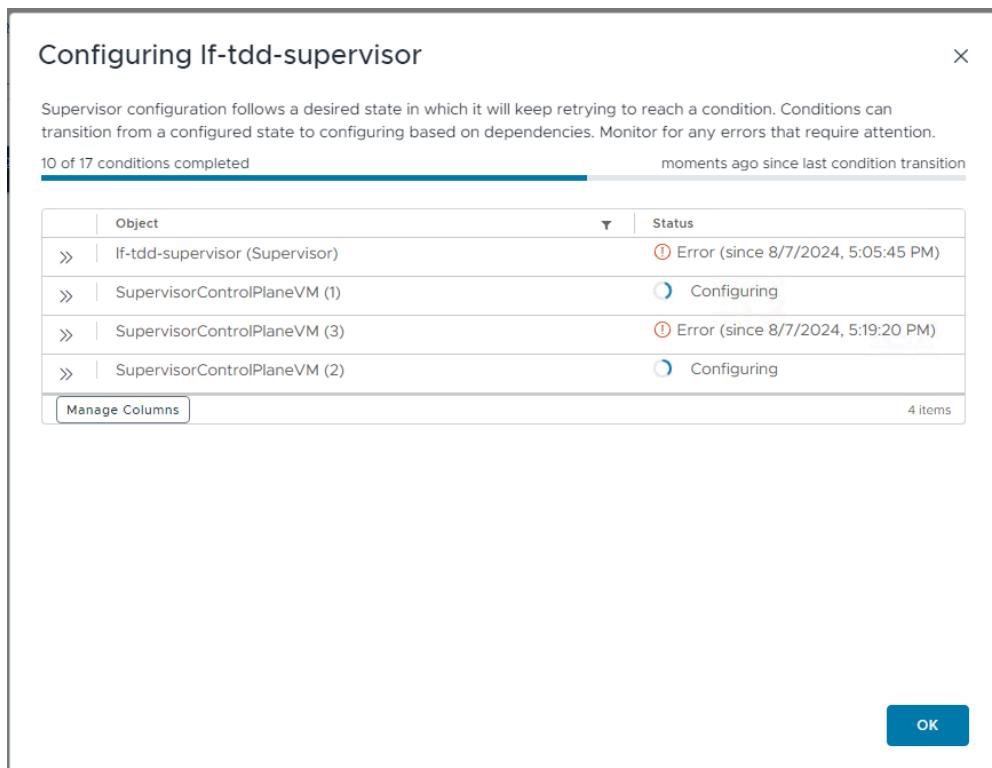
1 of 17 conditions completed 3 minutes since last condition change

Object	Status
» If-tdd-supervisor (Supervisor)	Configuring
» SupervisorControlPlaneVM (to be determined)	
» SupervisorControlPlaneVM (to be determined)	
» SupervisorControlPlaneVM (to be determined)	

Manage Columns 4 items

OK

This screen will progress as the Supervisor Nodes Deploy



Recent Tasks Alarms

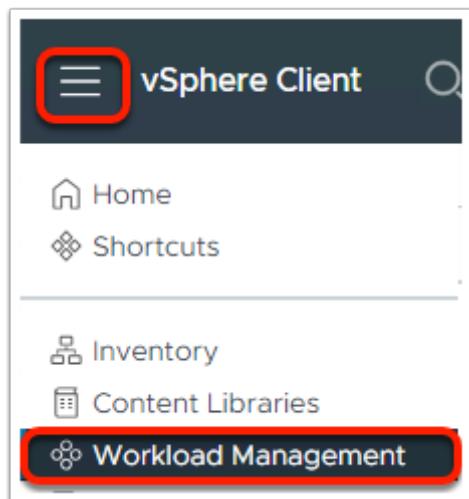
Task Name	Target	Status	Details
Deploy OVF template	Namespaces	0%	Copying Virtual Machine configuration
Deploy OVF template	Namespaces	0%	Copying Virtual Machine configuration
Deploy OVF template	Namespaces	0%	Copying Virtual Machine configuration
Download remote files			<div style="border: 2px solid red; padding: 5px;"> <p>HTTP communication could not be completed with status 404</p> <p>HTTP communication could not be completed with status 404</p> <p>HTTP communication could not be completed with status 404</p> </div>
Download remote files			
Download remote files			
Sync Library	Kubernetes Service Co	Completed	

Ignore these errors. They will clear.

The screenshot shows the 'Workload Management' interface. At the top, there are tabs for 'Namespaces', 'Supervisors' (which is selected), 'Services', and 'Updates'. Below the tabs are buttons for 'ADD SUPERVISOR', 'DEACTIVATE', 'CLONE CONFIG', 'EXPORT CONFIG', 'EXPORT LOGS', and 'RESTORE'. A 'Quick Filter' input field with the placeholder 'Enter value' is present. The main area is a table with columns: Supervisor, Namespaces, Hosts, Services, Status, Host Config Status, Control Plane Node Address, and CPU for namespaces. One row is visible, showing 'if-tdd-supervisor' with 2 namespaces, 4 hosts, and a status of 'Running' (indicated by a green checkmark). A red box highlights the 'Running' status in the 'Status' column.

## Check the Supervisor Status

Once deployment is complete, please examine and confirm its status



in vSphere client, click on the elipsis in the upper left corner and select Workload Management

The screenshot shows the vSphere with Tanzu interface. In the left navigation pane, 'Supervisors' is selected, and 'If-tdd-supervisor' is chosen. The main area shows the 'Summary' tab for the supervisor. The 'Status' section displays various metrics: Configuration Status (Running), Version (v1.28.3+vmware.2-fips.1-vsc0.1.9-23905380), vCenter Version (8.0.3), Kubernetes Status (Ready), Control Plane Node Address (10.80.0.2), and Node Health (Healthy). At the bottom right of the summary card, there is a 'VIEW DETAILS' button.

In the navigation pane on the left:

1. Go for **Supervisors** -> **If-tdd-supervisor**
2. This will navigate you to the **Summary** tab: Please note the **Running status, Kubernetes status** should be in a **Ready** state. You can also mark the **CP address: 10.80.0.2**
3. Click on **VIEW DETAILS**

The screenshot shows the 'Monitor' tab for the supervisor. It displays the 'Overview' section with various status items: Events, Configuration Status (Running), Version (v1.28.3+vmware.2-fips.1-vsc0.1.9-23905380), vCenter Version (8.0.3), Kubernetes Status (Ready), Control Plane Node Address (10.80.0.2), and Node Health (Healthy). Below this, a table lists the 'Control Plane VMs' with columns for Node Name, Status, and Object Name. The table shows seven items, with the last item being '4200b7c56b369021b669a23 6712a0913'.

Node Name	Status	Object Name
<a href="#">esxi-4.vcf.sddc.lab</a>	Ready	esxi-4.vcf.sddc.lab
<a href="#">esxi-1.vcf.sddc.lab</a>	Ready	esxi-1.vcf.sddc.lab
<a href="#">4200b7c56b369021b669a23 6712a0913</a>	Ready	SupervisorControlPlaneVM (3)
<a href="#">esxi-2.vcf.sddc.lab</a>	Ready	esxi-2.vcf.sddc.lab
<a href="#">4200e737786185a120ffb8b9 d0b4305b</a>	Ready	SupervisorControlPlaneVM (1)
<a href="#">esxi-3.vcf.sddc.lab</a>	Ready	esxi-3.vcf.sddc.lab
<a href="#">4200b85deb74d6a8daf04a 914a4e884</a>	Ready	SupervisorControlPlaneVM (2)

This will direct you In Monitor tab:

# Install Supervisor Services

## Task description and objectives

Supervisor Services are vSphere certified Kubernetes operators that deliver Infrastructure-as-a-Service components and tightly-integrated Independent Software Vendor services to developers. You can install and manage Supervisor Services on the vSphere IaaS control plane environment so that to make them available for use with workloads.

When Supervisor Services are installed on Supervisors, DevOps engineers can consume them in different ways:

Shared Supervisor Services such as Harbor, directly provide functionality to workloads running in TKG clusters, vSphere Pods, or VMs.

Supervisor Services that include an operator, such as MinIO, typically provide API or graphical interfaces, which DevOps engineers can use to create and manage instances of the service in a vSphere Namespace through CRDs. For example, to create a MinIO bucket, you use a CRD to create the bucket in a vSphere Namespace.

Supervisor Service	vSphere 7	vSphere 8
TKG Service	✗ *	 requires vSphere 8.0 Update 3 or later.
Consumption Interface	✗	
vSAN Data Persistence Platform Services - MinIO, Cloudian and Dell ObjectScale		
Backup & Recovery Service - Velero		
Certificate Management Service - cert-manager	✗	
Cloud Native Registry Service - Harbor	✗ *	
Kubernetes Ingress Controller Service - Contour	✗	
External DNS Service - ExternalDNS	✗	
* The embedded Harbor Registry and TKG Service features are still available and supported on vSphere 7 and onwards.		

New service will be added overtime with the goal to continue to empower your DevOps communities.

Prior to vSphere 8 Update 1, the Supervisor Services are only available with Supervisor Clusters enabled using VMware NSX-T. With vSphere 8 U1, Supervisor Services are also supported when using the vSphere Distributed Switch networking stack.

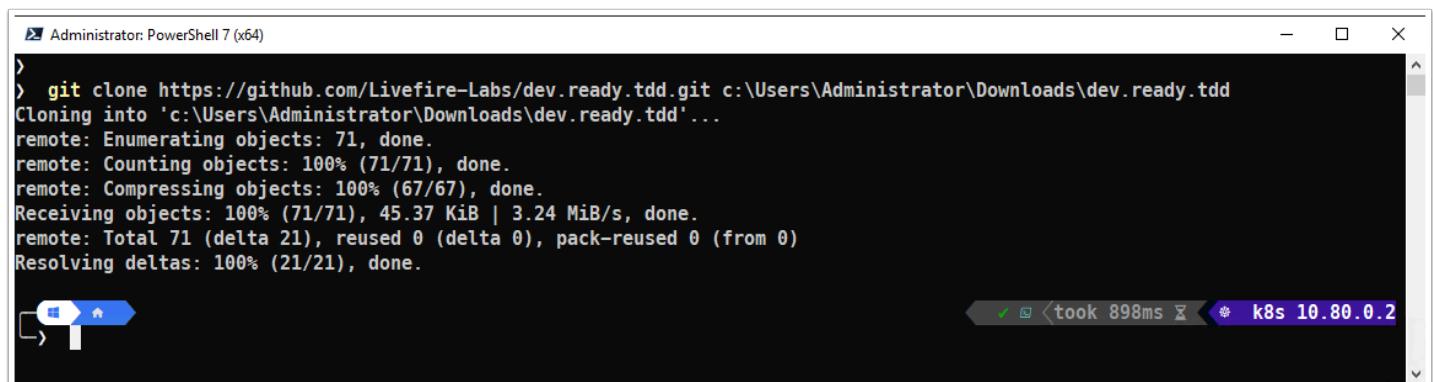
Supervisor services can be installed from their catalog, located at <https://vsphere-tmm.github.io/Supervisor-Services/>

Each one of these goes along with predefined manifest files of two stages: one manifest (service-name.yaml) for the service registration with the Supervisor, and second manifest (config-values.yaml) for particular service settings you need to apply in accordance with the environment.

With vSphere 8.0 u3 and VCF 5.2 onwards, couple of services are automatically deployed when you enable the Supervisor:

- VM Service
- Tanzu Kubernetes Grid Service
- Velero vSphere Operator

## Clone Github



```
Administrator: PowerShell 7 (x64)
> git clone https://github.com/Livefire-Labs/dev.ready.tdd.git c:\Users\Administrator\Downloads\dev.ready.tdd
Cloning into 'c:\Users\Administrator\Downloads\dev.ready.tdd'...
remote: Enumerating objects: 71, done.
remote: Counting objects: 100% (71/71), done.
remote: Compressing objects: 100% (67/67), done.
Receiving objects: 100% (71/71), 45.37 KiB | 3.24 MiB/s, done.
remote: Total 71 (delta 21), reused 0 (delta 0), pack-reused 0 (from 0)
Resolving deltas: 100% (21/21), done.

took 898ms
k8s 10.80.0.2
```

 First things first you will need some files. we've put them in a public github repository./ You will need to clone them to your DDownloads folder

Run the git clone command below:

```
git clone https://github.com/Livefire-Labs/dev.ready.tdd.git c:\Users\Administrator\Downloads\dev.ready.tdd
```

This will create a **dev.ready.tdd** folder in the **c:\Users\Administrator\Downloads** folder

## Install Kubernetes Ingress Controller Service (Contour)

For a **Kubernetes Ingress Controller Service** you can use **Contour**.

Contour is an Ingress controller for Kubernetes that works by deploying the Envoy proxy as a reverse proxy and load balancer. Contour supports dynamic configuration updates out of the box while maintaining a lightweight profile.

## Kubernetes Ingress Controller Service



CONTOUR

Contour is an Ingress controller for Kubernetes that works by deploying the Envoy proxy as a reverse proxy and load balancer. Contour supports dynamic configuration updates out of the box while maintaining a lightweight profile.

- Service install - Follow steps 1 - 5 in the [documentation](#).

### Contour Versions

- Download latest version: [Contour v1.28.2](#) (1)
- Download version: [Contour v1.24.4](#)
- Download version: [Contour v1.18.2](#)

Contour Sample `values.yaml`

- Download [values for all versions](#). (2) These values can be used *as-is* and require no configuration changes.

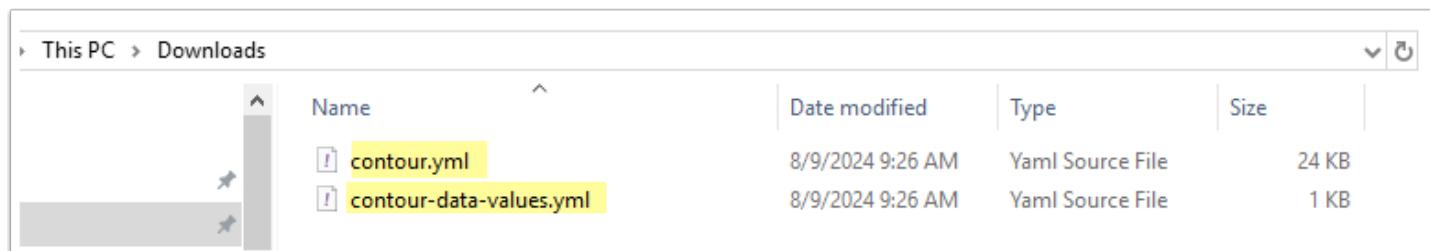
Download the latest available manifests for Contour to your Downloads folder

1. `contour.yml`

2. `contour-data-values.yml`

<https://vsphere-tmm.github.io/Supervisor-Services/#kubernetes-ingress-controller-service>

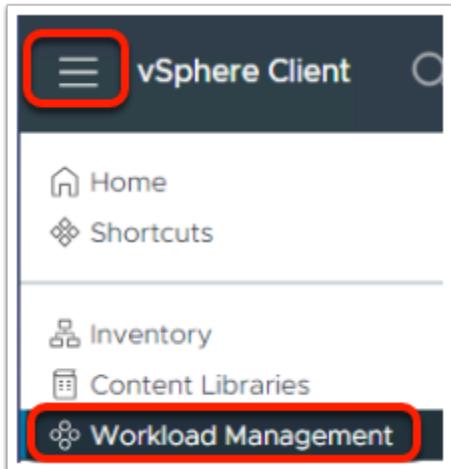
## Install Kubernetes Ingress Controller Service (Contour)



Name	Date modified	Type	Size
<code>contour.yml</code>	8/9/2024 9:26 AM	Yaml Source File	24 KB
<code>contour-data-values.yml</code>	8/9/2024 9:26 AM	Yaml Source File	1 KB

You are ready to deploy Contour as a Supervisor service!

# Install Kubernetes Ingress Controller Service (Contour)



From the vSphere Client home menu, select **Workload Management**.

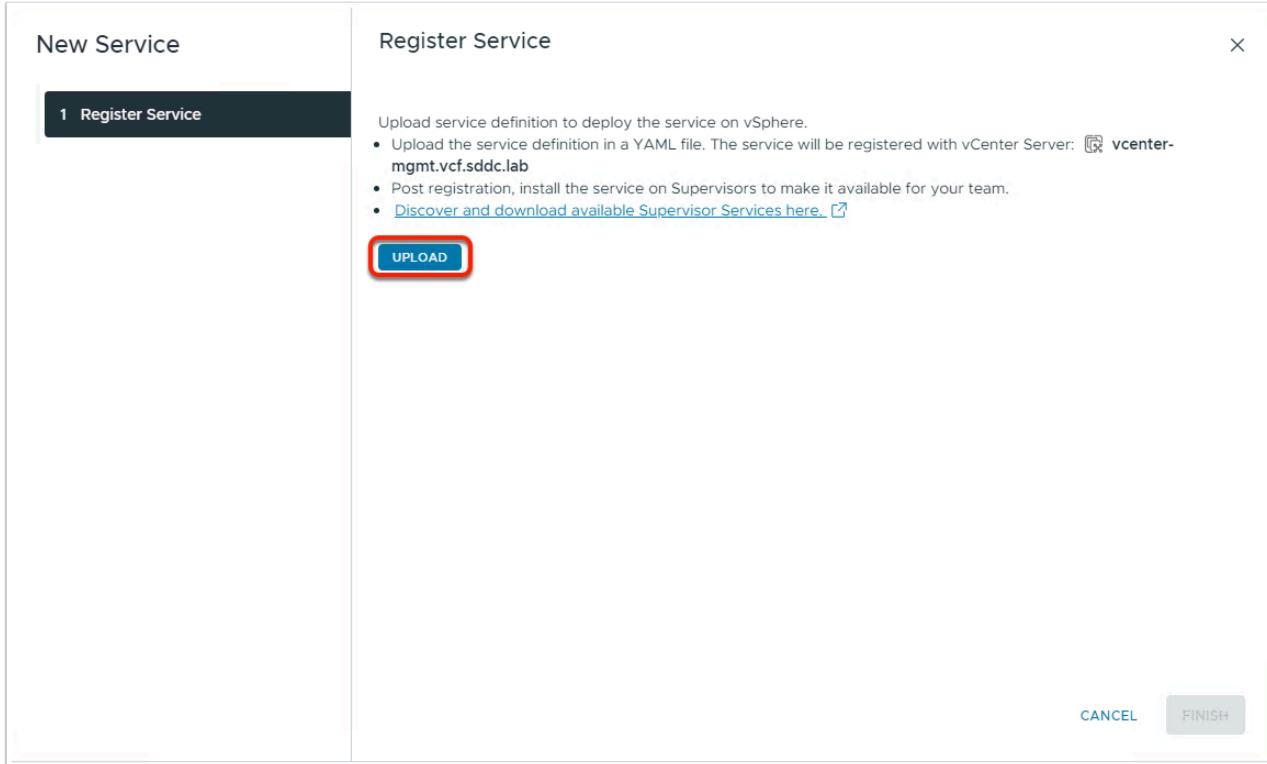
# Install Kubernetes Ingress Controller Service (Contour)

A screenshot of the 'Workload Management' section in the vSphere Client. At the top, there are tabs for 'Namespaces', 'Supervisors', 'Services' (which is highlighted with a red box), and 'Updates'. Below the tabs, it says 'Supervisor Services' and shows a connection to 'VCENTER-MGMT.VCF.SDDC.LAB'. It includes a 'Sort By' dropdown set to 'Recently added'. A note says 'Below are the services registered to this vCenter Server system. You can manage services with multiple versions from the same service card.' There are four service cards: 1) 'Add New Service' (with an 'ADD' button highlighted with a red box), 2) 'VM Service' (described as allowing self-service VMs and policies), 3) 'Tanzu Kubernetes Grid Service' (status: Active, Core Service, Active Versions: 1, Supervisors: 1), and 4) 'Velero vSphere Operator' (status: Active, Core Service, Active Versions: 1, Supervisors: 1).

In the **Workload Management** section, select **Services**.

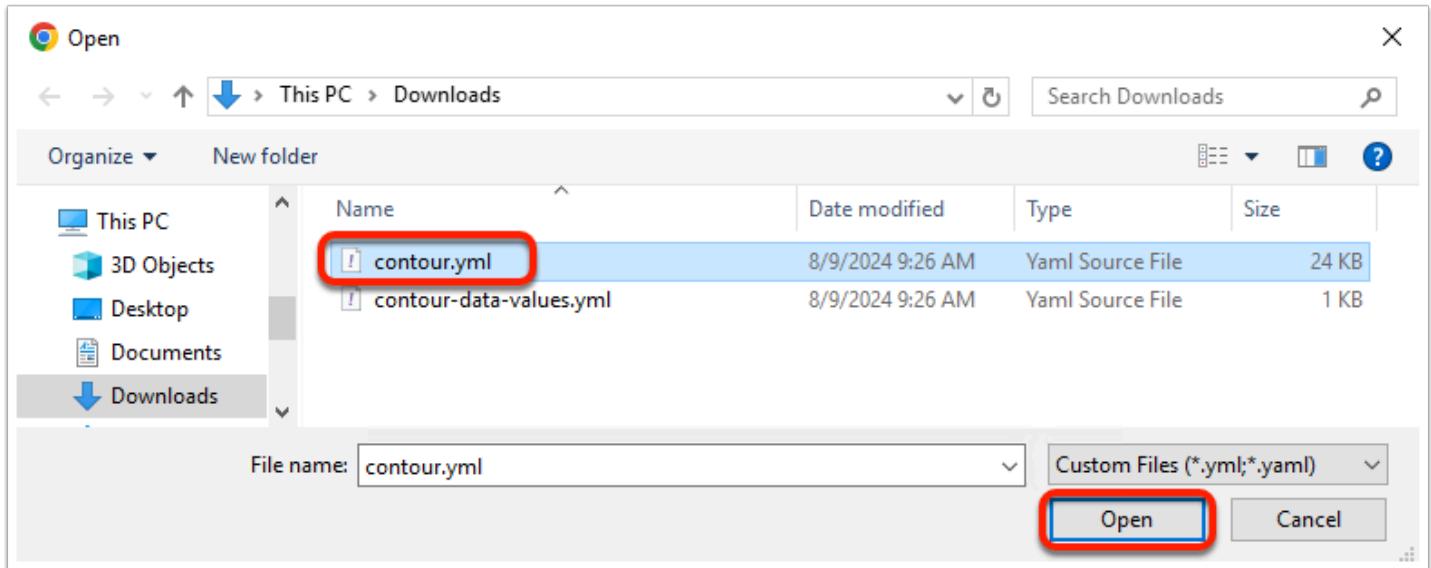
Locate the **Add New Service** card and click on **ADD**

# Install Kubernetes Ingress Controller Service (Contour)



In the **New Service** wizard, for **Register Service** select **UPLOAD**

# Install Kubernetes Ingress Controller Service (Contour)



In the Chrome **Open** menu go to **Downloads** folder and select the **contour.yml** you downloaded recently.

Confirm with clicking on **Open** button

## Install Kubernetes Ingress Controller Service (Contour)

New Service

1 Register Service

Register Service

⚠️ Running 3rd party services on user workloads has security risks. A 3rd party service has network access to user workloads, vSphere Pods, and exposed APIs.

ⓘ YAML was uploaded successfully. Note: YAML content is not verified and could fail during installation into a Supervisor.

Upload service definition to deploy the service on vSphere.

YAML File details [Upload new](#)  
contour.yml

Service Details

vCenter Server	vcenter-mgmt.vcf.sddc.lab
Service Name	contour
Service ID	contour.tanzu.vmware.com
Service Description	An ingress controller
Version	1.28.2+vmware.1-tkg.1

CANCEL FINISH

In the **New Service** wizard, for **Register Service** make sure you can see the aforementioned **contour.yml** file.

Its displayed version should match the one you downloaded.

Confirm with **FINISH**

# Install Kubernetes Ingress Controller Service (Contour)

The screenshot shows the vCenter Workload Management interface under the Supervisor Services tab. At the top, there are tabs for Namespaces, Supervisors, Services (which is selected), and Updates. Below the tabs, it says "Supervisor Services" and shows the URL "VCENTER-MGMT.VCF.SDDC.LAB".  
A message box at the top right states: "Service 'contour' is successfully registered. You can now install the service on Supervisors."  
The main area displays several service cards:

- VM Service**: Status: Active (Active Versions 1, Supervisors 0). A red box highlights the "contour" service name.
- Tanzu Kubernetes Grid Service**: Status: Active (Core Service, Active Versions 1, Supervisors 1). Cluster management.
- Velero vSphere Operator**: Status: Active (Core Service, Active Versions 1, Supervisors 1). Helps users install Velero and the vSphere plugi...

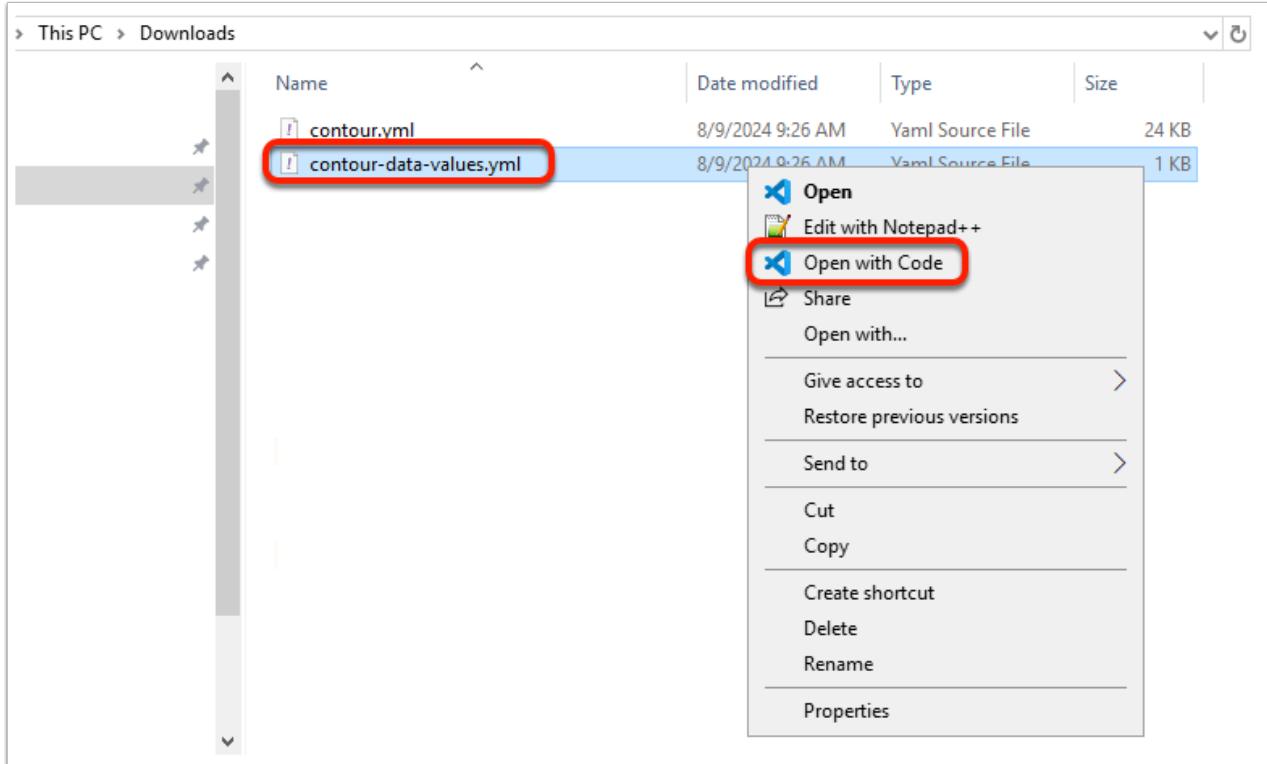
In a while you will see a displayed message for the Contour service successfully registered with your Supervisor.

You need to activate it with particular configuration settings.



You can safely close the message in green.

# Install Kubernetes Ingress Controller Service (Contour)



Let's examine the particular file. You can review/edit it with any available editor

In Windows Explorer, navigate to your **Downloads** folder and locate the recently downloaded file **contour-data-values.yml**.

Right click and **Open with Code**

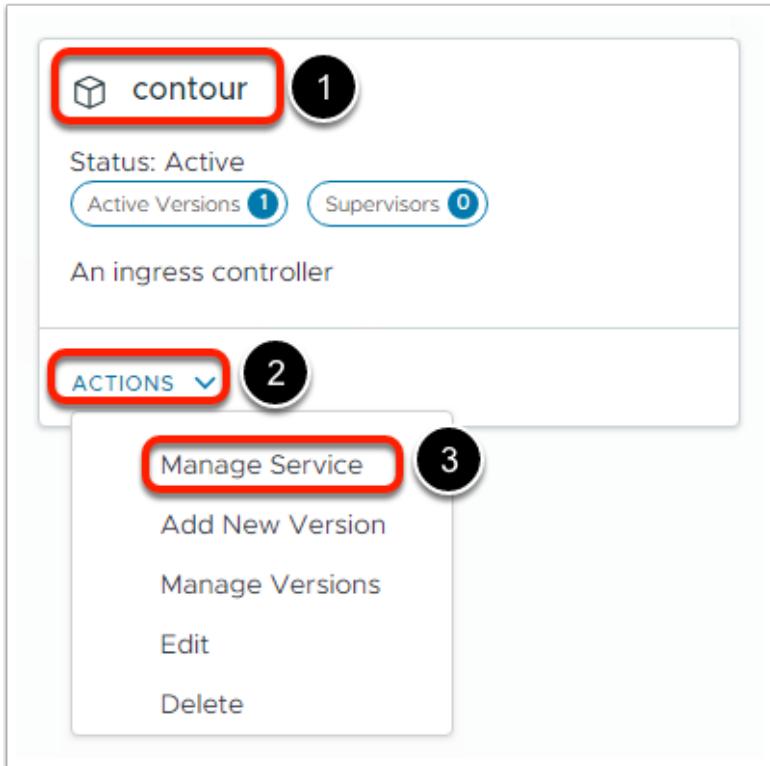
# Install Kubernetes Ingress Controller Service (Contour)

```
C: > Users > Administrator > Downloads > contour-data-values.yml >
1  contour:
2    configFileContents: {}
3    useProxyProtocol: false
4    replicas: 2
5    pspNames: ""
6    logLevel: info
7    envoy:
8      service:
9        type: LoadBalancer
10       externalTrafficPolicy: Cluster
11       disableWait: false
12     hostPorts:
13       enable: false
14       http: 80
15       https: 443
16     hostNetwork: false
17     terminationGracePeriodSeconds: 300
18     logLevel: info
19     pspNames: ""
20   certificates:
21     duration: 8760h
22     renewBefore: 360h
23
```

For Contour data values you don't need to edit anything.

Don't close the file, you will need the values shortly!

# Install Kubernetes Ingress Controller Service (Contour)



Go back to Chrome -> vSphere UI -> Workload Management -> Services

1. Locate your recently registered **contour** service
2. From **Actions** drop down menu
3. select **Manage Service**

# Install Kubernetes Ingress Controller Service (Contour)

The screenshot shows the 'Configure' step of the 'Manage' wizard. The left sidebar has '1 Configure' selected. The main area displays a table of supervisors:

Supervisor	Service Version Name	Version	Service Status
If-tdd-supervisor	--	--	-

Below the table are 'Manage Columns' and '1 item' buttons. At the bottom right are 'CANCEL' and 'NEXT' buttons, with 'NEXT' being highlighted by a red circle.

In the **Manage** wizard, from **Configure** select your **Supervisor** and confirm **NEXT**

# Install Kubernetes Ingress Controller Service (Contour)

The screenshot shows a 'Manage' interface with a 'Review' step selected. The review screen displays the following information:

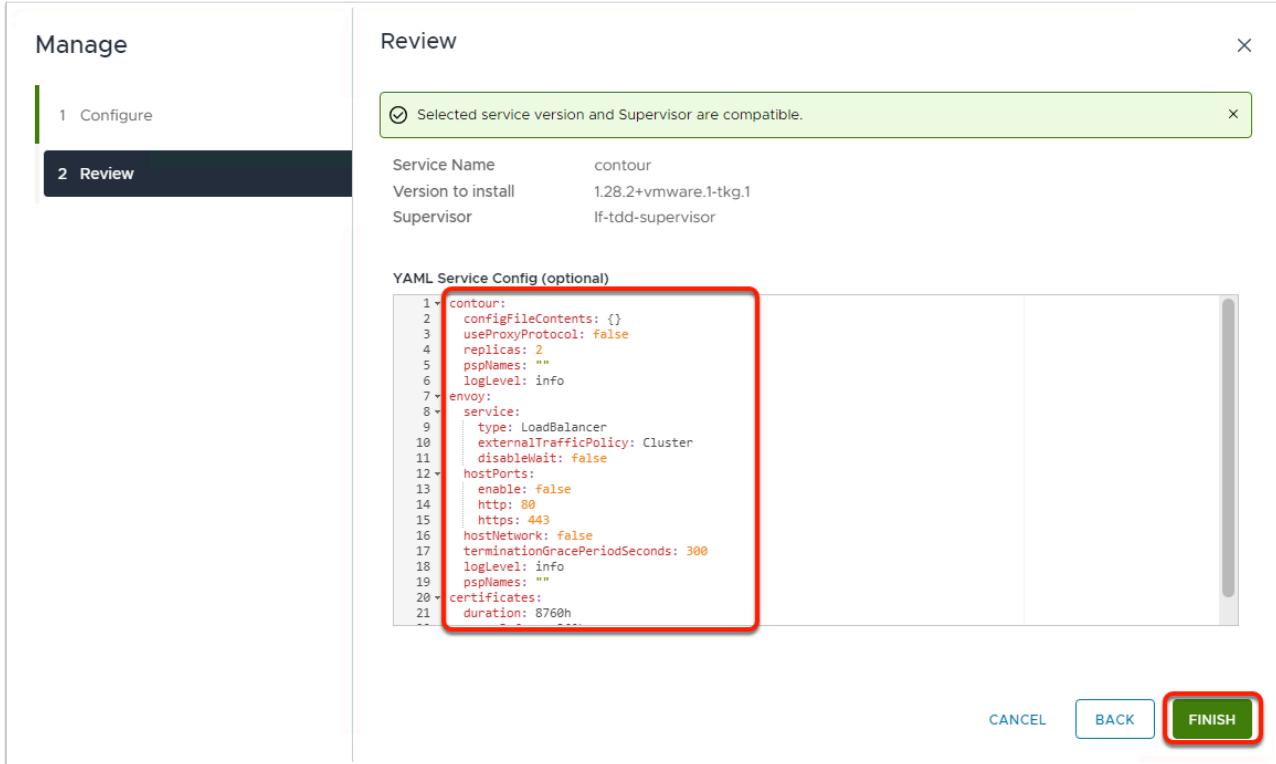
Service Name	contour
Version to install	1.28.2+vmware.1-tkg.1
Supervisor	lf-tdd-supervisor

A note at the top states: "Selected service version and Supervisor are compatible." Below this, there is a section titled "YAML Service Config (optional)" which contains a redacted YAML configuration. At the bottom right are buttons for "CANCEL", "BACK", and "FINISH".

You need to provide the service configuration values.

Copy these from the **Visual Studio Code**, where you lastly reviewed **contour-data-values.yml**

# Install Kubernetes Ingress Controller Service (Contour)



Select all and paste into the **YAML Service Config** section in the UI wizard.

Confirm with **FINISH**

# Install Kubernetes Ingress Controller Service (Contour)

Supervisor	Namespaces	Hosts	Services	Config Status	Host Config Status
lf-tdd-supervisor	3	4	View	Running	Running

Go back in the **Workload Management** section

Select **Supervisors** and click on **View** under **Services**

# Install Kubernetes Ingress Controller Service (Contour)

The screenshot shows the 'Supervisor Services' section of the If-tdd-supervisor interface. Under 'Service Version Name', the 'contour' service is selected. Its 'Deployment Namespace' is 'svc-contour-domain-c9'. The 'Status' column shows 'Configured' with a green checkmark. The 'Version' is '1.28.2+vmware.1-tkg.1'. The 'Desired version' is '1.28.2+vmware.1-tkg.1'. Other services listed include 'Tanzu Kubernetes Grid Service' and 'Velero vSphere Operator', both also in 'Configured' status.

In a couple of minutes, the **contour** service **Status** will change from Configuring to **Configured**  
Click on the **svc-contour-domain-c9** under **Deployment Namespace**

# Install Kubernetes Ingress Controller Service (Contour)

The screenshot shows the 'Compute' tab for the 'svc-contour-domain-c9' namespace. The tabs at the top are Summary, Monitor, Configure, Permissions, Compute (highlighted with a red box), Storage, Network, and Resources.

For the **svc-contour-domain-c9** Namespace, click on **Compute**

# Install Kubernetes Ingress Controller Service (Contour)

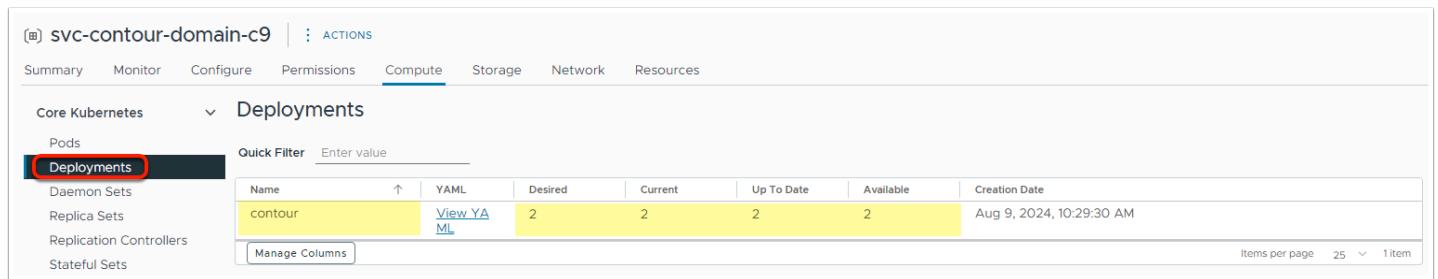
The screenshot shows the 'Pods' tab for the 'svc-contour-domain-c9' namespace. The tabs at the top are Summary, Monitor, Configure, Permissions, Compute, Storage, Network, and Resources. The 'Pods' section lists several pods:

Name	YAML	Phase	Creation Date	Cluster IP	Containers	Namespace	vSphere Pod
contour-7ff5b5db47-7mff9	<a href="#">View YAML</a>	Running	Aug 9, 2024, 10:29:30 AM	10.80.0.2	1/1	svc-contour-domain-c9	Yes
contour-7ff5b5db47-chpdx	<a href="#">View YAML</a>	Running	Aug 9, 2024, 10:29:30 AM	10.80.0.2	1/1	svc-contour-domain-c9	Yes
envoy-6x2lm	<a href="#">View YAML</a>	Running	Aug 9, 2024, 10:29:30 AM	10.80.0.2	2/2	svc-contour-domain-c9	Yes
envoy-drlt6	<a href="#">View YAML</a>	Running	Aug 9, 2024, 10:29:30 AM	10.80.0.2	2/2	svc-contour-domain-c9	Yes
envoy-tk99s	<a href="#">View YAML</a>	Running	Aug 9, 2024, 10:29:30 AM	10.80.0.2	2/2	svc-contour-domain-c9	Yes
envoy-xkj2g	<a href="#">View YAML</a>	Running	Aug 9, 2024, 10:29:30 AM	10.80.0.2	2/2	svc-contour-domain-c9	Yes

Under **Compute**, you can review all the **Kubernetes Core** configurations for harbor.

Select **Pods** and you will see all the **contour** needed pods, running as **vSphere Pods**

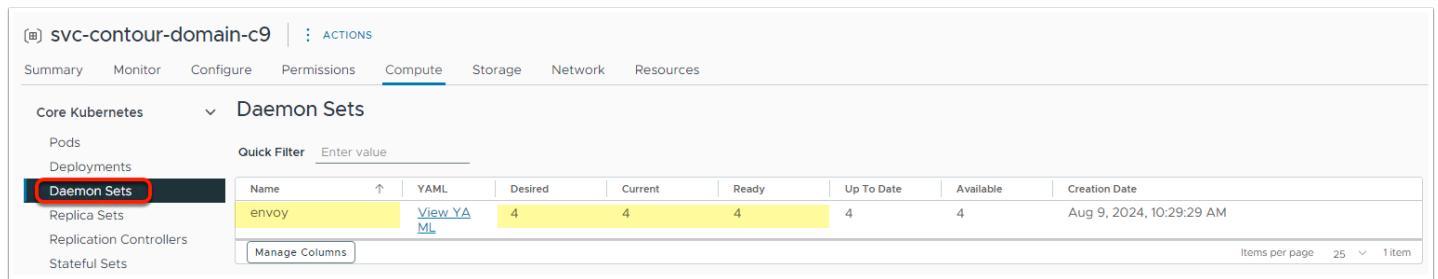
## Install Kubernetes Ingress Controller Service (Contour)



Name	YAML	Desired	Current	Up To Date	Available	Creation Date
contour	<a href="#">View YAML</a>	2	2	2	2	Aug 9, 2024, 10:29:30 AM

Select **Deployments** and you will see all the **contour** needed deployments in place

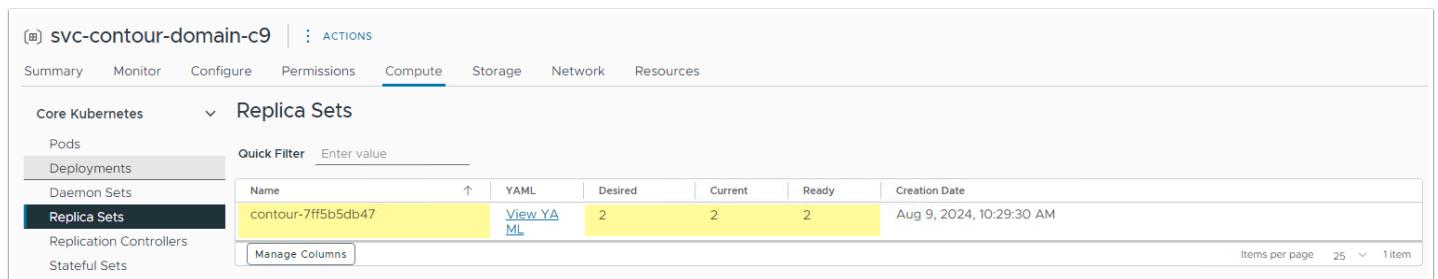
## Install Kubernetes Ingress Controller Service (Contour)



Name	YAML	Desired	Current	Ready	Up To Date	Available	Creation Date
envoy	<a href="#">View YAML</a>	4	4	4	4	4	Aug 9, 2024, 10:29:29 AM

Select **Daemon Sets** and you will see all the **contour** needed daemon sets in place: for envoy

## Install Kubernetes Ingress Controller Service (Contour)



Name	YAML	Desired	Current	Ready	Creation Date
contour-7ff5b5db47	<a href="#">View YAML</a>	2	2	2	Aug 9, 2024, 10:29:30 AM

Select **Replica Sets** and you will see all the **contour** needed replica sets in place

Okay, let's switch to the Network configurations for **contour** service:

## Install Kubernetes Ingress Controller Service (Contour)

The screenshot shows the 'svc-contour-domain-c9' namespace summary page. At the top, there is a 'Network' tab which is highlighted with a red box. Below the tabs, there are several other tabs: Summary, Monitor, Configure, Permissions, Compute, Storage, and Resources.

For the **svc-contour-domain-c9** Namespace, click on **Network**

## Install Kubernetes Ingress Controller Service (Contour)

The screenshot shows the 'Network' tab of the 'svc-contour-domain-c9' namespace. Under the 'Services' tab, the 'Services' section is highlighted with a red box. The table below lists two services: 'contour' and 'envoy'. The 'contour' service has a ClusterIP of 10.96.0.236 and is of type ClusterIP, with port 8001/TCP. The 'envoy' service has a ClusterIP of 10.96.1.51 and is of type LoadBalancer, with ports 80:30306/TCP, 443:31175/TCP, and an External IP of 10.80.0.3.

Name	YAML	Cluster IP	Type	Ports	External IPs
contour	<a href="#">View YAML</a>	10.96.0.236	ClusterIP	8001/TCP	
envoy	<a href="#">View YAML</a>	10.96.1.51	LoadBalancer	80:30306/TCP, 443:31175/TCP	10.80.0.3

Under **Network**, you can review all the configured settings.

Select **Services** and you will see the **External IP Address** that should be used as **LoadBalancer**, in this case it is **10.80.0.3**

Remember this value, you are going to use it shortly!

## Install ExternalDNS

For publishing **DNS records for Applications** to DNS servers, you can use **ExternalDNS** Service.



ExternalDNS publishes DNS records for applications to DNS servers, using a declarative, Kubernetes-native interface. This operator connects to your DNS server (not included here). For a list of supported DNS providers and their corresponding configuration settings, see the [upstream external-dns project](#).

- On Supervisors where Harbor is deployed with Contour, ExternalDNS may be used to publish a DNS hostname for the Harbor service.

#### ExternalDNS Versions

- Download latest version: [ExternalDNS v0.13.4](#)
- Download version: [ExternalDNS v0.11.0](#)

ExternalDNS data `values.yaml`

Download the latest available manifests for External DNS to your Downloads folder

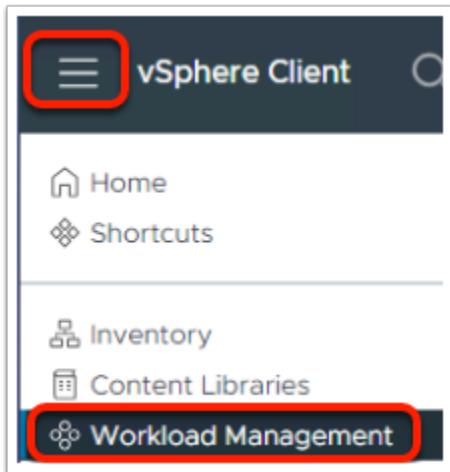
1. `external-dns.yml`

<https://vsphere-tmm.github.io/Supervisor-Services/#external-dns-service>

## Install ExternalDNS

This PC > Downloads				
	Name	Date modified	Type	Size
	contour.yml	8/9/2024 9:26 AM	Yaml Source File	24 KB
	contour-data-values.yml	8/9/2024 10:22 AM	Yaml Source File	1 KB
	external-dns.yml	8/9/2024 9:33 AM	Yaml Source File	42 KB

You are ready to deploy External DNS as a Supervisor service!



From the vSphere Client home menu, select **Workload Management**.

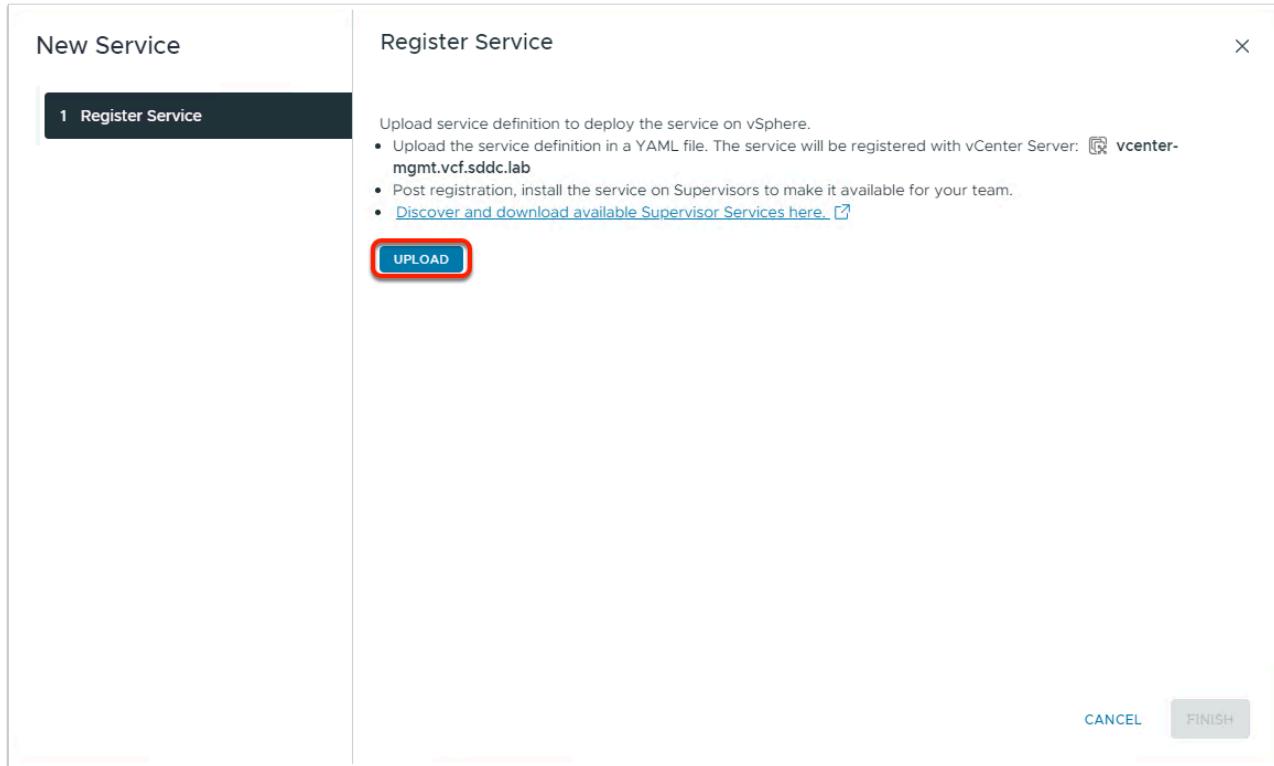
## Install ExternalDNS

A screenshot of the 'Workload Management' section in the vSphere Client. The 'Services' tab is selected, indicated by a red box. Under 'Supervisor Services', there's a note about Supervisor Services being a platform for managing core infrastructure components. It shows a list of registered services: 'VM Service', 'contour', 'Tanzu Kubernetes Grid Service', and 'Velero vSphere Operator'. Each service card has an 'ADD' button at the bottom, which is also highlighted with a red box. The 'Tanzu Kubernetes Grid Service' card shows its status as 'Active' with one active version and one supervisor.

In the **Workload Management** section, select **Services**.

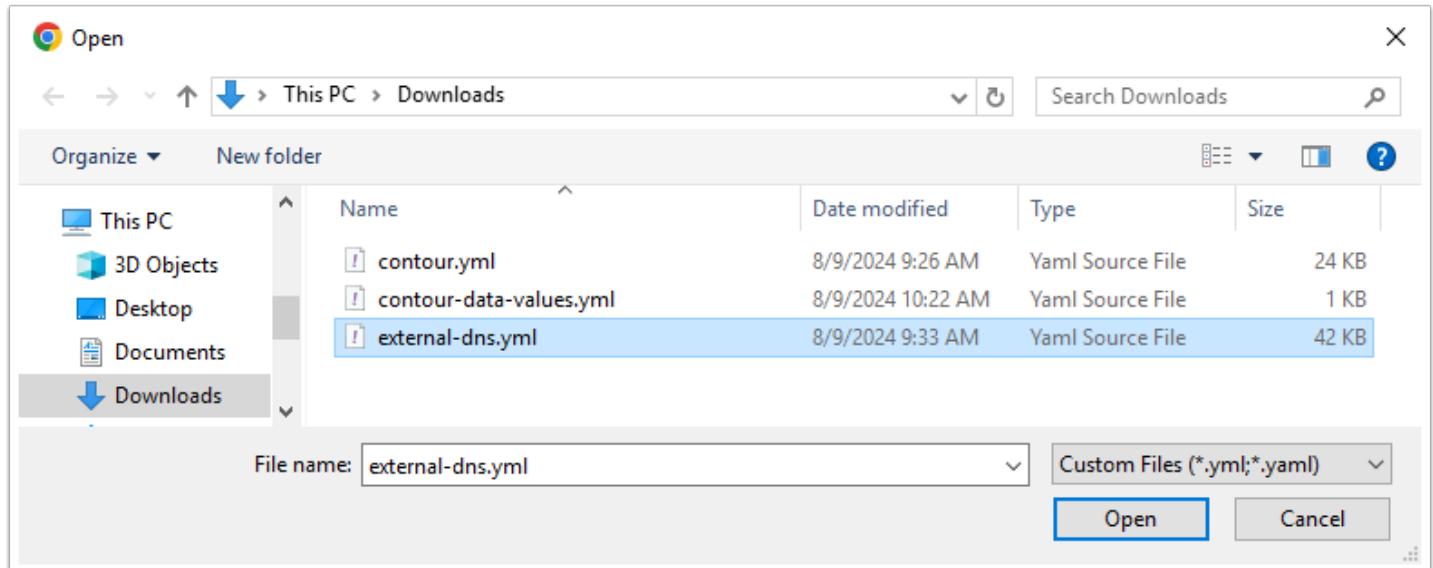
Locate the **Add New Service** card and click on **ADD**

# Install ExternalDNS



In the **New Service** wizard, for **Register Service** select **UPLOAD**

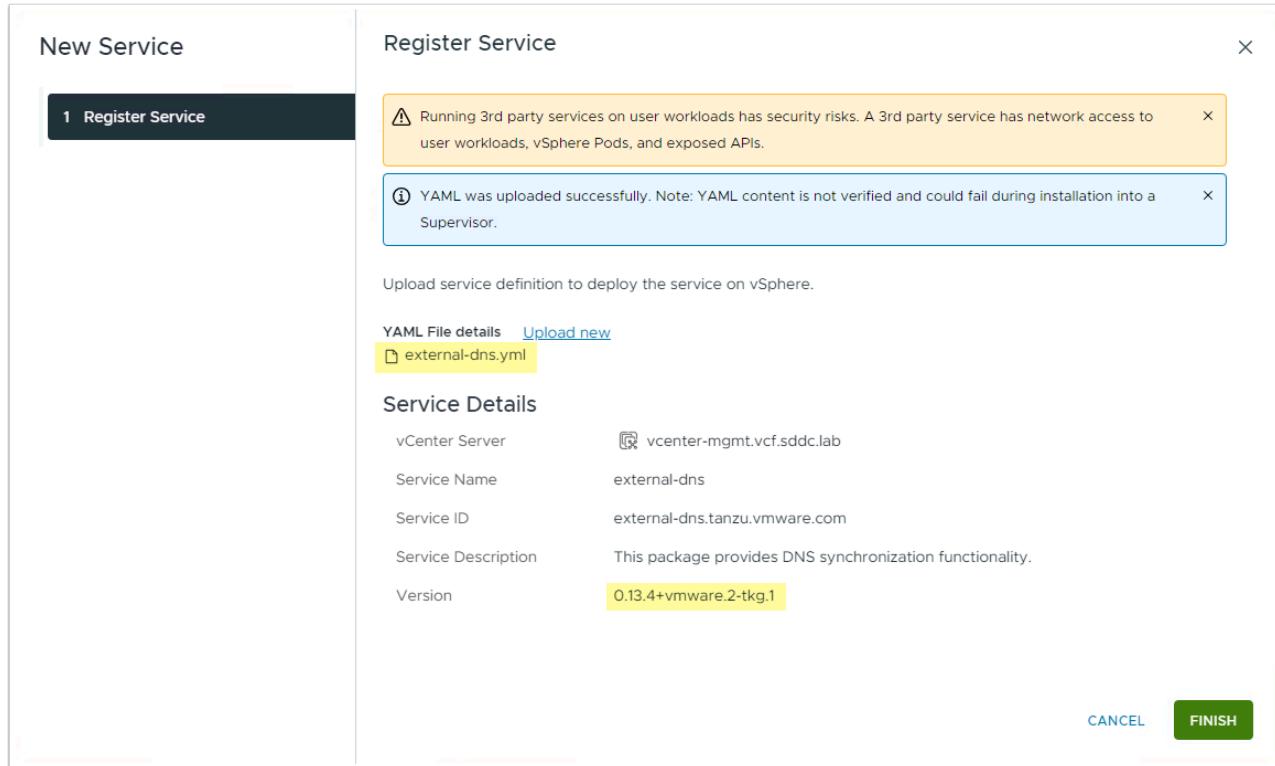
# Install ExternalDNS



In the Chrome **Open** menu go to **Downloads** folder and select the **external-dns.yml** you downloaded recently.

Confirm with clicking on **Open** button

## Install ExternalDNS



In the **New Service** wizard, for **Register Service** make sure you can see the aforementioned **external-dns.yml** file.

Its displayed version should match the one you downloaded.

Confirm with **FINISH**

# Install ExternalDNS

The screenshot shows the 'Supervisor Services' section of the vSphere Workload Management interface. At the top, there are tabs for 'Namespaces', 'Supervisors', 'Services' (which is selected), and 'Updates'. Below the tabs, it says 'Supervisor Services | VCENTER-MGMT.VCF.SDDC.LAB'. A green success message box contains the text: 'Service 'external-dns' is successfully registered. You can now install the service on Supervisors.' The 'external-dns' service card is highlighted with a red box. Other services listed include 'VM Service', 'contour', 'Tanzu Kubernetes Grid Service', and 'Velero vSphere Operator'. Each service card shows its status (Active), active versions (1), and supervisors (0).

In a while you will see a displayed message for the ExternalDNS service successfully registered with your Supervisor.

You need to activate it with particular configuration settings.

 You can safely close the message in green.

# Install ExternalDNS

ExternalDNS data `values.yaml`

- Because of the large list of supported DNS providers, we do not supply complete sample configuration values here. If you're deploying ExternalDNS with Harbor and Contour, make sure to include `source=contour-httpproxy` in the configuration values. An *incomplete* example of the service configuration is included below. Make sure to setup API access to your DNS server and include authentication details with the service configuration.

```
deployment:  
  args:  
    - --source=contour-httpproxy  
    - --source=service  
    - --log-level=debug
```

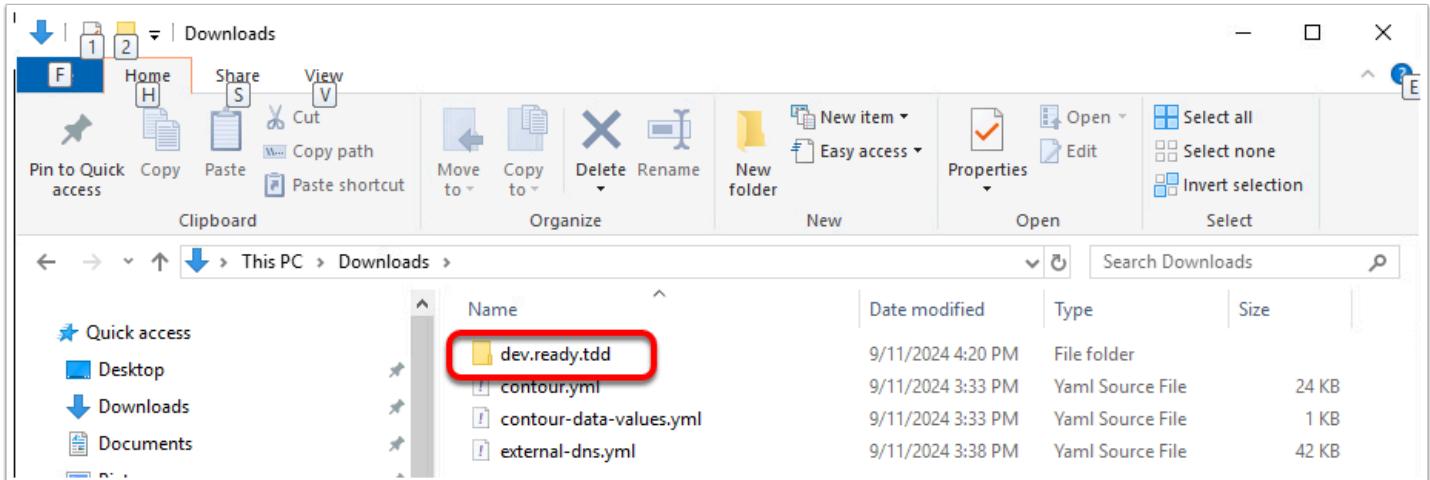
Normally you need to create the **ExternalDNS** data values manifest file on your own, to match your environment. But we have done that for you.

# Install ExternalDNS

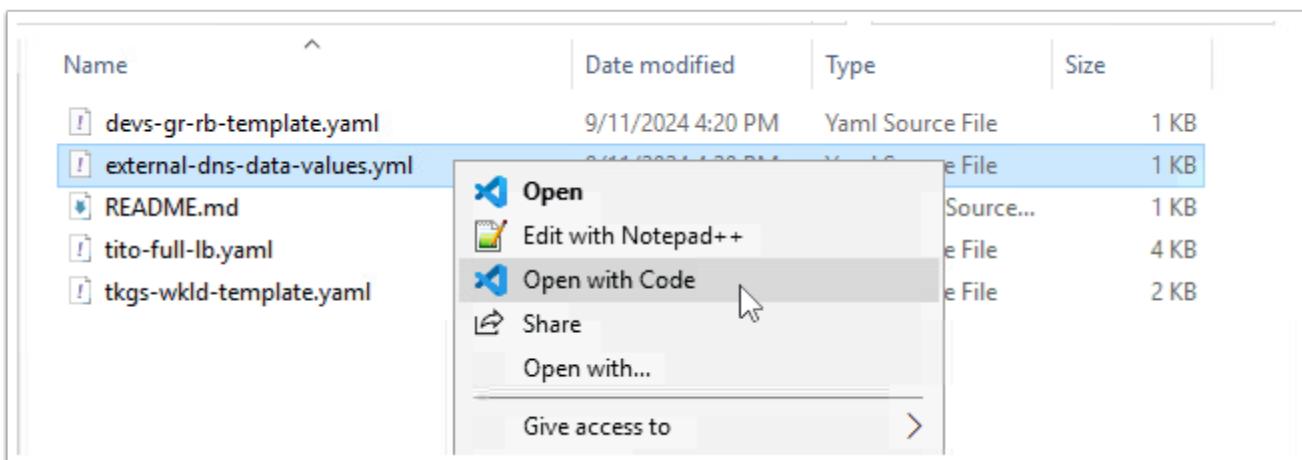
```
C: > Users > Administrator > Downloads > external-dns-data-values.yml  
1 deployment:  
2   args:  
3     - --registry=txt  
4     - --txt-prefix=external-dns-  
5     - --txt-owner-id=tanzu  
6     - --provider=rfc2136  
7     - --rfc2136-host=10.0.0.201  
8     - --rfc2136-port=53  
9     - --rfc2136-zone=vcf.holo.lab  
10    - --rfc2136-insecure  
11    - --rfc2136-tsig-axfr  
12    - --source=service  
13    - --source=contour-httpproxy  
14    - --source=ingress  
15    - --domain-filter=vcf.holo.lab  
16    namespace: svc-external-dns-domain-c9
```

The github repository you cloned earlier has the file you need to deploy this service to the Supervisor Cluster.

## dev.ready.tdd

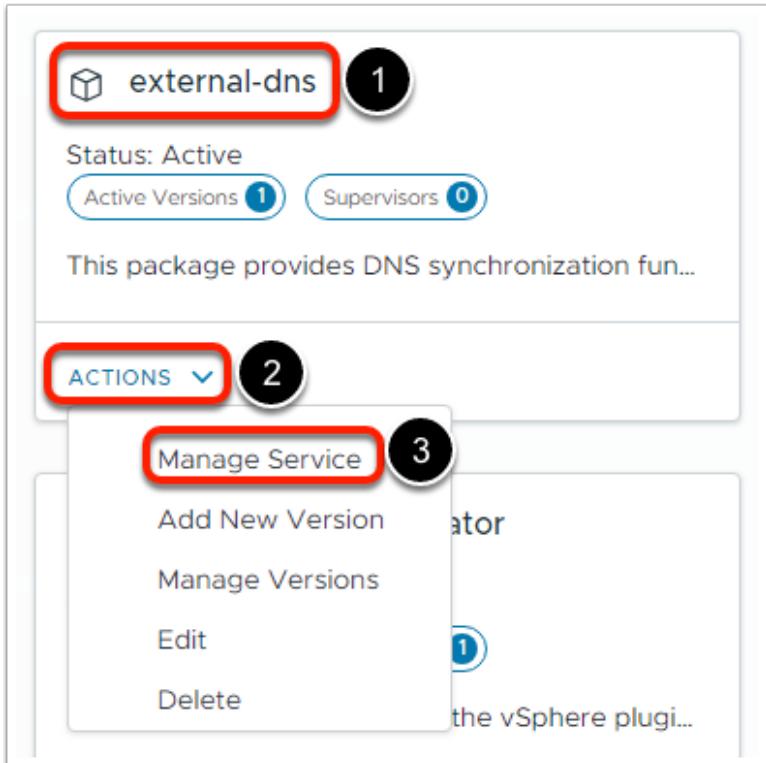


Have a look in this folder and you will find the **external-dns-data-values.yml** file that you need for this part of the lab. Remember this folder for later in the labs



Right click and open this file in VSCode. Go back to vCenter and add the external-dns service to the supervisor cluster using the values in your clipboard

# Install ExternalDNS



Go back to Chrome -> vSphere UI -> Workload Management -> Services

1. Locate your recently registered **external-dns** service
2. From **ACTIONS** drop down menu
3. select **Manage Service**

# Install ExternalDNS

The screenshot shows the 'Configure' step of the 'Manage' wizard. The left sidebar has '1 Configure' selected. The main area displays a table of supervisors:

Supervisor	Service Version Name	Version	Service Status
lf-tdd-supervisor	--	--	-

Below the table are 'Manage Columns' and '1 item' buttons. At the bottom right are 'CANCEL' and 'NEXT' buttons, with 'NEXT' being highlighted by a red box.

In the **Manage** wizard, from **Configure** select your Supervisor and confirm **NEXT**

# Install ExternalDNS

The screenshot shows the 'Review' step of a service installation process. On the left, a sidebar labeled 'Manage' shows steps 1 'Configure' and 2 'Review'. Step 2 is highlighted with a dark background and white text. The main area is titled 'Review' and contains a green notification bar with the message: 'Selected service version and Supervisor are compatible.' Below this, a table shows configuration details:

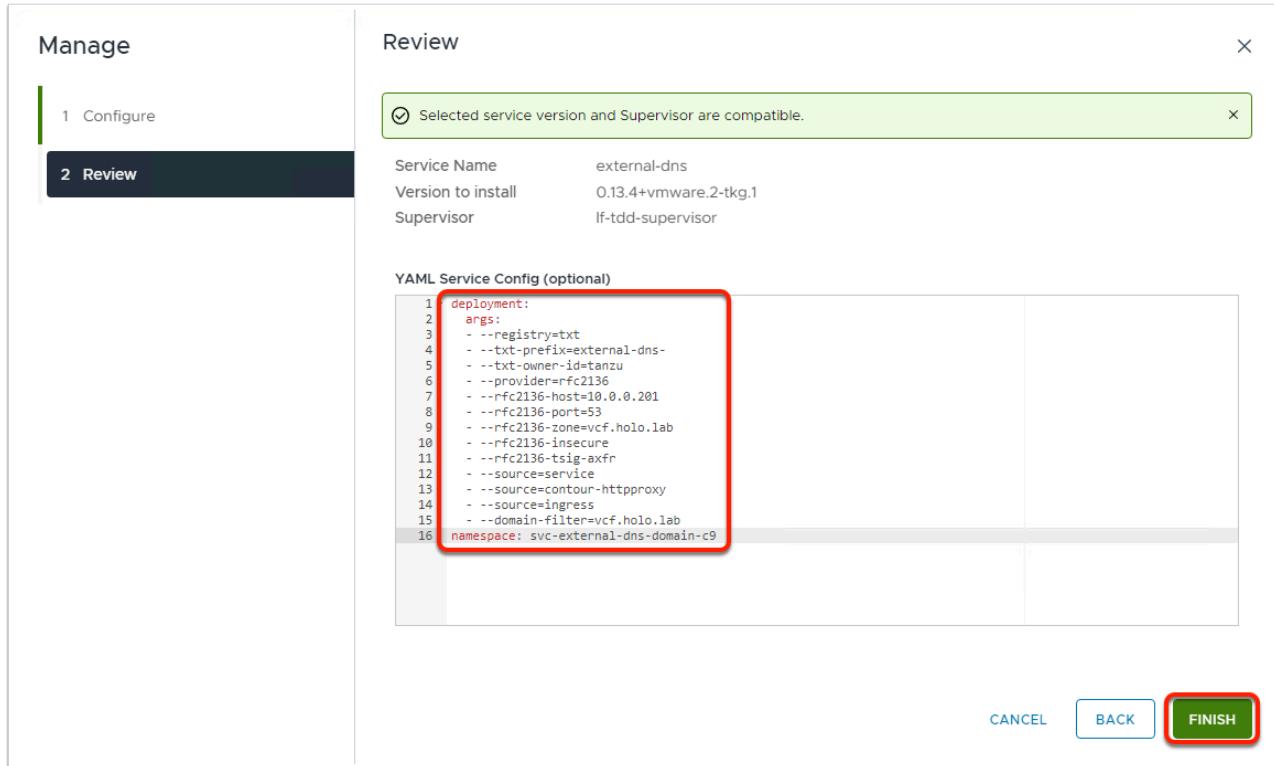
Service Name	external-dns
Version to install	0.13.4+vmware.2-tkg.1
Supervisor	lf-tdd-supervisor

Below the table is a section titled 'YAML Service Config (optional)' which contains a code editor window showing a single line of YAML: '1'. At the bottom right are three buttons: 'CANCEL', 'BACK' (in a blue box), and 'FINISH' (in a green box).

You need to provide the service configuration values.

Copy these from the Visual Studio Code, where you lastly edited **external-dns-data-values.yml**

# Install ExternalDNS



Select all and paste into the **YAML Service Config** section in the UI wizard.

Confirm with **FINISH**

# Install ExternalDNS

The screenshot shows the 'Workload Management' interface. The top navigation bar includes 'Namespaces', 'Supervisors' (which is highlighted with a red box), 'Services', and 'Updates'. Below the navigation are buttons for 'ADD SUPERVISOR', 'DEACTIVATE', 'CLONE CONFIG', 'EXPORT CONFIG', 'EXPORT LOGS', and 'RESTORE'. A 'Quick Filter' input field is present. The main table lists supervisors. One row for 'lf-tdd-supervisor' is selected and highlighted with a red box. The columns include 'Supervisor', 'Namespaces', 'Hosts', 'Services', 'Config Status', and 'Host Config Status'. The 'View' button in the 'Services' column for the selected supervisor is also highlighted with a red box.

Go back in the **Workload Management** section

Select **Supervisors** and click on **View** under **Services**

# Install ExternalDNS

The screenshot shows the 'Supervisor Services' section of the 'If-tdd-supervisor' interface. Under 'Supervisor Services', there is a table listing various services. The 'external-dns' service is highlighted with a red box around its row. Its status is shown as 'Configured'.

Service Version Name	Deployment Namespace	Status	Version	Desired version
contour	svc-contour-domain-c9	Configured	1.28.2+vmware.1-tkg.1	1.28.2+vmware.1-tkg.1
<b>external-dns</b>	<b>svc-external-dns-domain-c9</b>	<b>Configured</b>	0.13.4+vmware.2-tkg.1	0.13.4+vmware.2-tkg.1
Tanzu Kubernetes Grid Service	svc-tkg-domain-c9	Configured	3.0.0-embedded	3.0.0-embedded
Velero vSphere Operator	svc-velero-domain-c9	Configured	1.6.1-embedded+23741747	1.6.1-embedded+23741747

In a while the **external-dns** service Status will change from Configuring to **Configured**

Click on the **svc-external-dns-domain-c9** under **Deployment Namespace**

# Install ExternalDNS

The screenshot shows the 'Compute' tab for the 'svc-external-dns-domain-c9' namespace. The 'Compute' tab is highlighted with a red box.

For the **svc-external-dns-domain-c9** Namespace, click on **Compute**

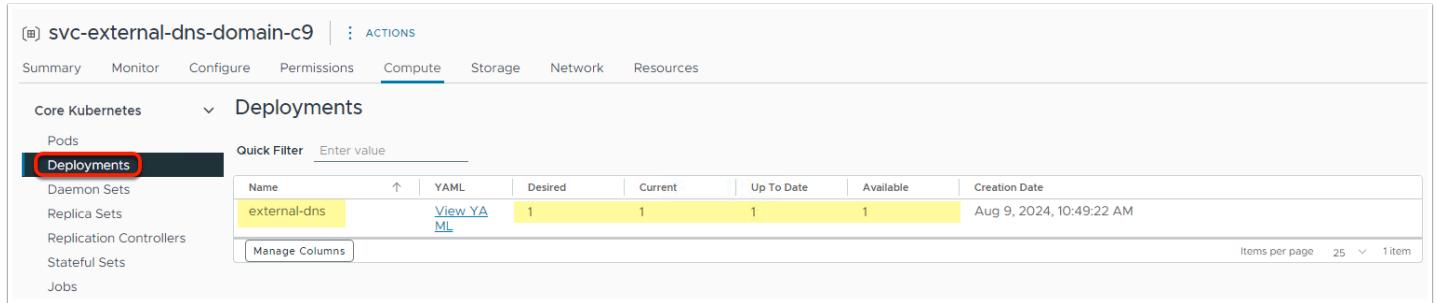
# Install ExternalDNS

The screenshot shows the 'Pods' tab for the 'svc-external-dns-domain-c9' namespace. The 'Pods' section is highlighted with a red box. A specific pod, 'external-dns-56b4fdbc57-sh8c7', is selected and highlighted in yellow.

Under **Compute**, you can review all the **Kubernetes Core** configurations for ExternalDNS.

Select **Pods** and you will see all the **ExternalDNS** needed pods, running as **vSphere Pods**

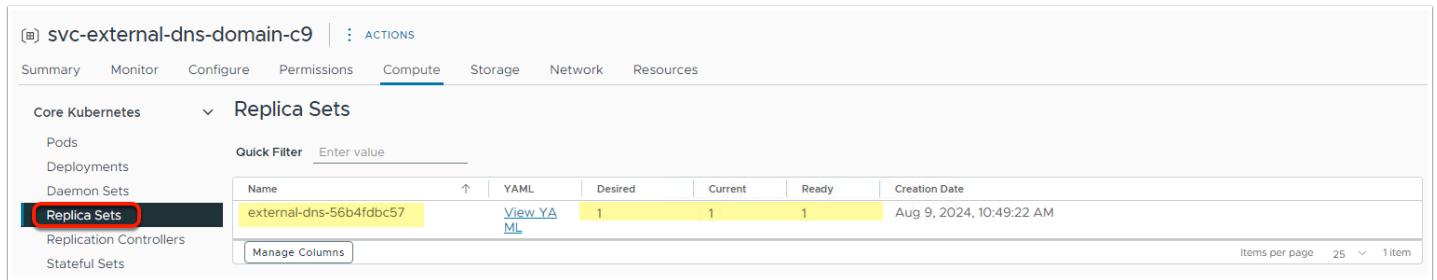
# Install ExternalDNS



Name	YAML	Desired	Current	Up To Date	Available	Creation Date
external-dns	<a href="#">View YAML</a>	1	1	1	1	Aug 9, 2024, 10:49:22 AM

Select Deployments and you will see all the **external-dns** needed deployments in place

# Install ExternalDNS



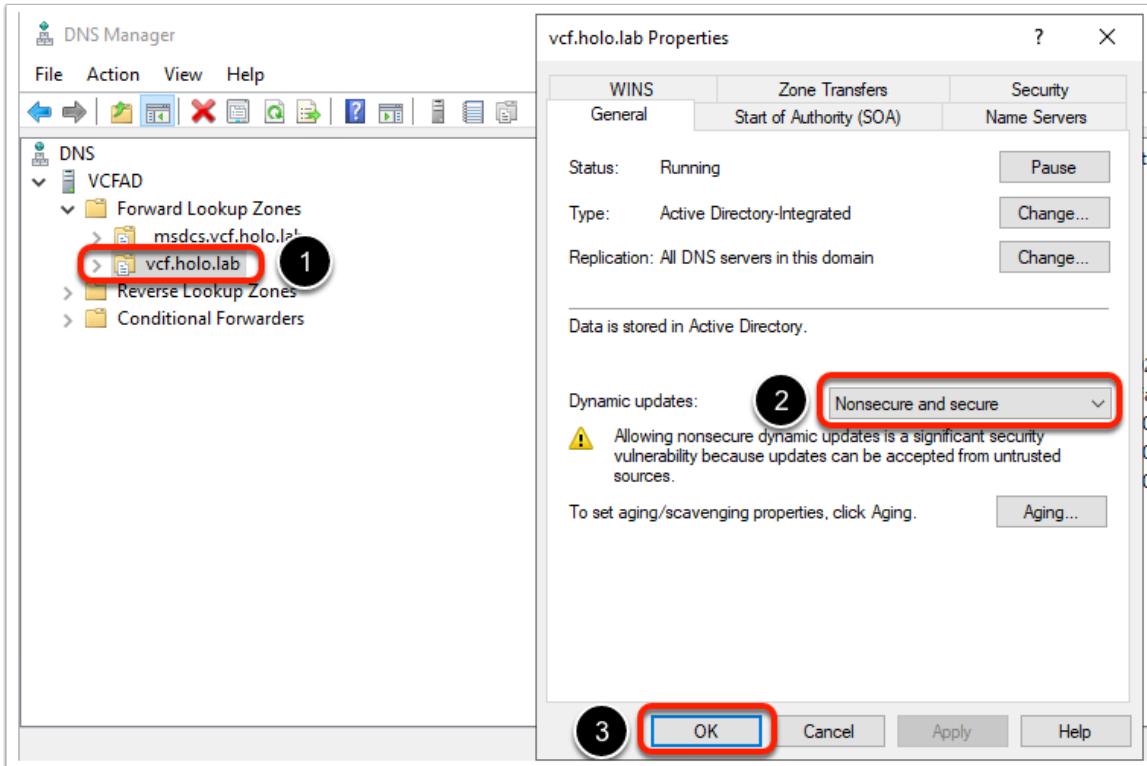
Name	YAML	Desired	Current	Ready	Creation Date
external-dns-56b4fdbc57	<a href="#">View YAML</a>	1	1	1	Aug 9, 2024, 10:49:22 AM

Select Replica Sets and you will see all the ExternalDNS needed replica sets in place



Please, note: With the configuration provided, ExternalDNS is on par with [RFC2139](#), hence you might need to consider the default behaviour, which is providing non-secure dynamic Updates in Domain Name System.

# Install ExternalDNS



From your Windows desktop: start the DNS Manager, expand the Forward Lookup Zones under VCFAD

1. right click on the **vcf.holo.lab** domain, select **Properties**
2. in the **Properties** window, under **General**, Change the **Dynamic updates** option to **Nonsecure and secure**
3. confirm with **OK**

Now the DNS server is ready to accept Dynamic updates from our **external-dns** service.

# Install Cloud Native Registry Service (Harbor)

For a **Cloud Native Registry Service** you can use **Harbor**.

Harbor is an open source trusted cloud native registry project that stores, signs, and scans content. Harbor extends the open source Docker Distribution by adding the functionalities usually required by users such as security, identity and management. Having a registry closer to the build and run environment can improve the image transfer efficiency. Harbor supports replication of images between registries, and also offers advanced security features such as user management, access control and activity auditing.

## Cloud Native Registry Service



Harbor is an open source trusted cloud native registry project that stores, signs, and scans content. Harbor extends the open source Docker Distribution by adding the functionalities usually required by users such as security, identity and management. Having a registry closer to the build and run environment can improve the image transfer efficiency. Harbor supports replication of images between registries, and also offers advanced security features such as user management, access control and activity auditing.

- The [contour package](#) is a prerequisite for Harbor, so that must be installed first.
- Follow the instructions under [Installing and Configuring Harbor on a Supervisor](#).

### Harbor Versions

- Download latest version [Harbor v2.9.1](#) 1
- Download version: [Harbor v2.8.2](#)
- Download version: [Harbor v2.5.3](#)

Harbor Sample [values.yaml](#)

- Download latest version [values for v2.9.1](#). For details about each of the required properties, see the configuration details page.
- Download version: [values for v2.8.2](#). For details about each of the required properties, see the configuration details page.
- Download version: [values for v2.5.3](#). For details about each of the required properties, see the configuration details page.

Download the latest available manifests for **harbor** to your **Downloads** folder

1. harbor.yml

2. harbor-data-values.yml

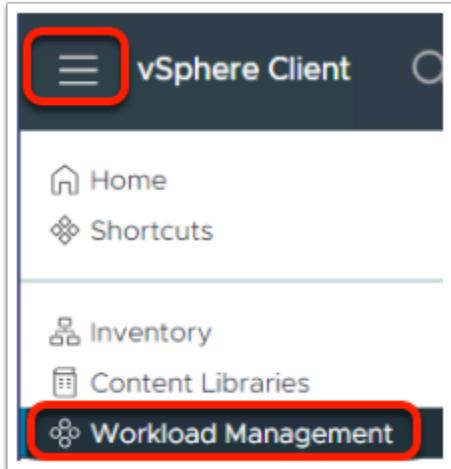
<https://vsphere-tmm.github.io/Supervisor-Services/#cloud-native-registry-service>

## Install Cloud Native Registry Service (Harbor)

This PC > Downloads				
	Name	Date modified	Type	Size
	contour.yml	8/9/2024 9:26 AM	Yaml Source File	24 KB
	contour-data-values.yml	8/9/2024 10:22 AM	Yaml Source File	1 KB
	external-dns.yml	8/9/2024 9:33 AM	Yaml Source File	42 KB
	external-dns-data-values.yml	8/9/2024 10:43 AM	Yaml Source File	1 KB
	harbor.yml	8/9/2024 9:42 AM	Yaml Source File	46 KB
	harbor-data-values.yml	8/9/2024 9:46 AM	Yaml Source File	4 KB

You are ready to deploy Harbor as a Supervisor service!

# Install Cloud Native Registry Service (Harbor)



From the vSphere Client home menu, select **Workload Management**.

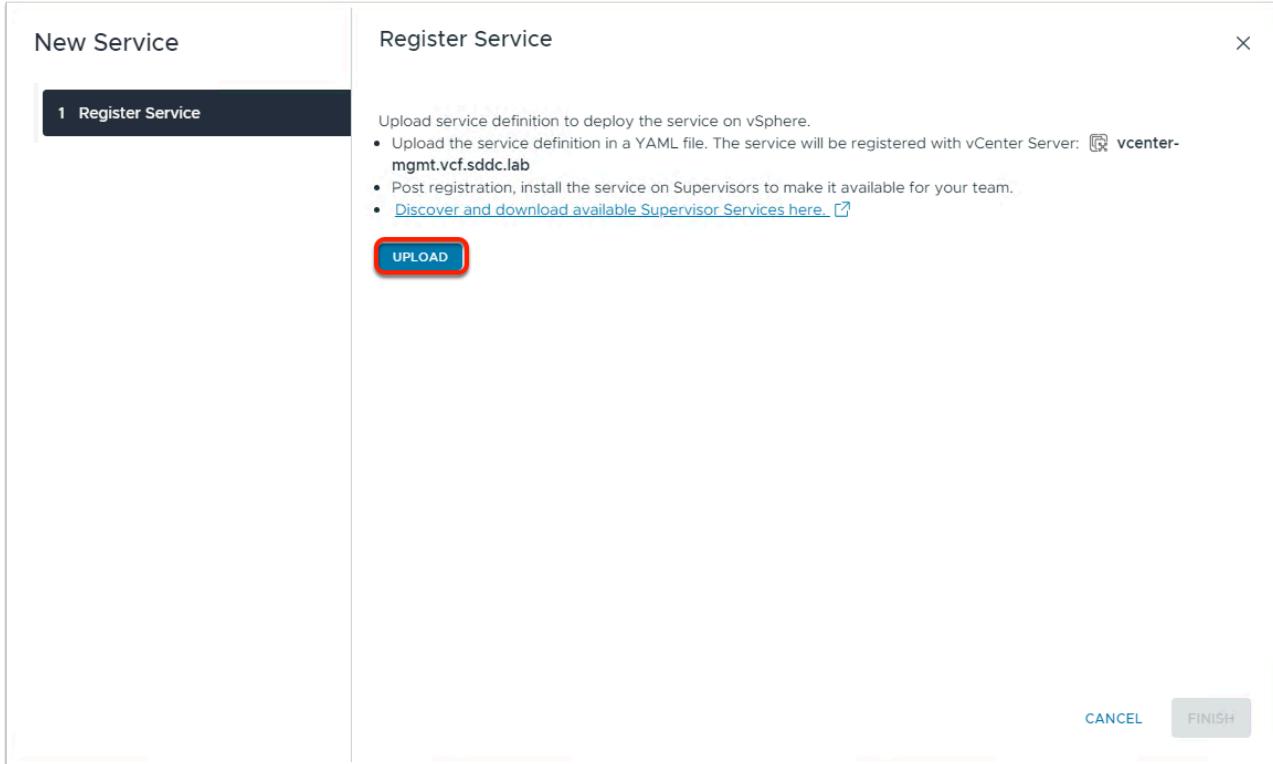
# Install Cloud Native Registry Service (Harbor)

A screenshot of the 'Workload Management' section in the vSphere Client. At the top, there are tabs for 'Namespaces', 'Supervisors', 'Services' (which has a red box around it), and 'Updates'. Below the tabs, it says 'Supervisor Services' and shows a connection to 'VCENTER-MGMT.VCF.SDDC.LAB'. It explains Supervisor Services as a platform for managing infrastructure components. A 'Sort By' dropdown is set to 'Recently added'. Below this, it says 'Below are the services registered to this vCenter Server system.' There are four service cards: 1) 'VM Service' (with an 'ADD' button highlighted with a red box), 2) 'contour' (status: Active, Active Versions 1, Supervisors 1), 3) 'Tanzu Kubernetes Grid Service' (Status: Active, Active Version 1, Supervisors 1), and 4) 'Velero vSphere Operator' (Status: Active, Active Version 1, Supervisors 1).

In the **Workload Management** section, select **Services**.

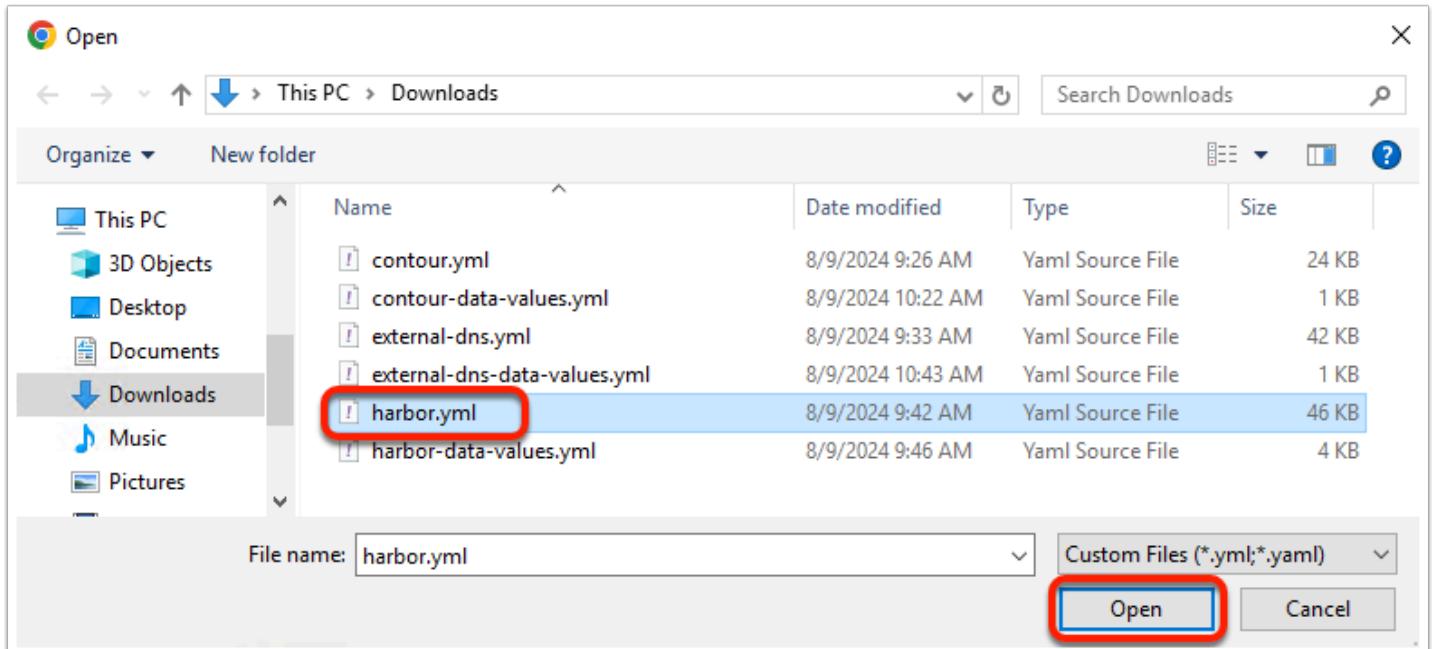
Locate the **Add New Service** card and click on **ADD**

# Install Cloud Native Registry Service (Harbor)



In the **New Service** wizard, for **Register Service** select **UPLOAD**

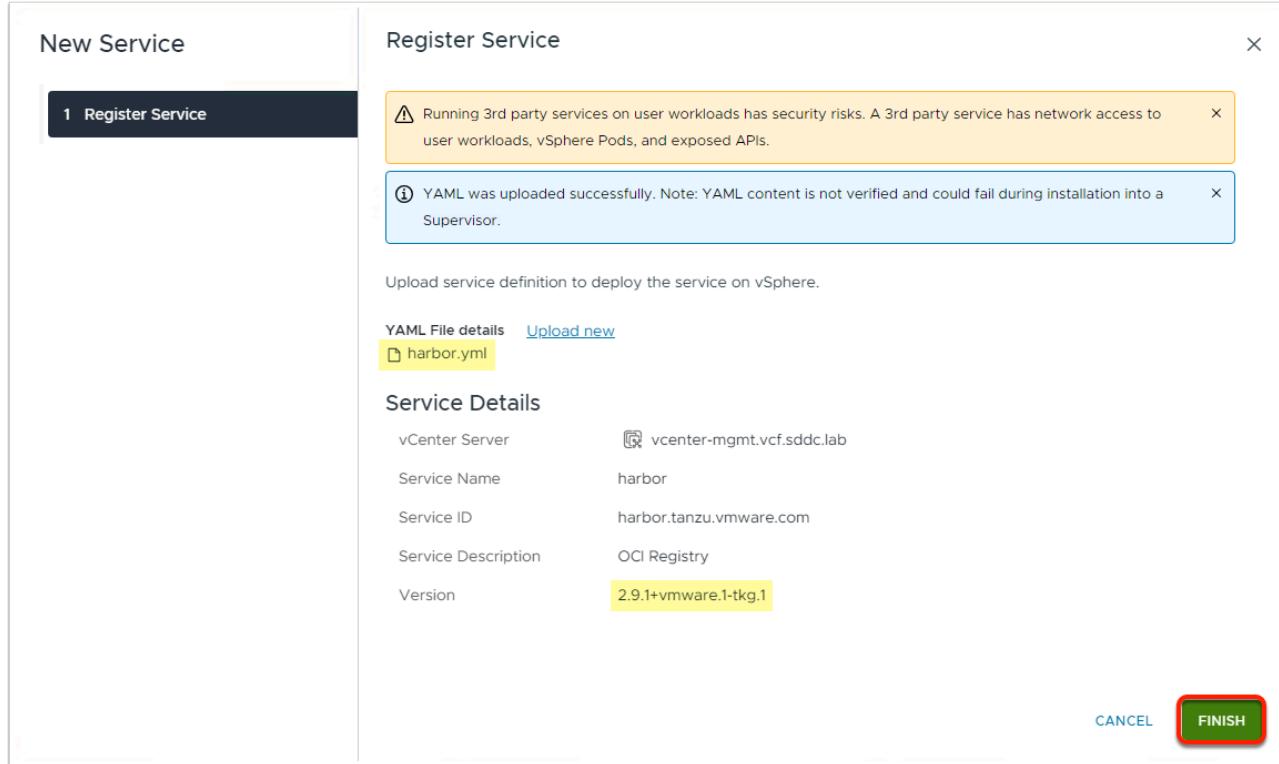
# Install Cloud Native Registry Service (Harbor)



In the Chrome **Open** menu go to **Downloads** folder and select the **harbor.yml** you downloaded recently.

Confirm with clicking on **Open** button

## Install Cloud Native Registry Service (Harbor)



In the **New Service** wizard, for **Register Service** make sure you can see the aforementioned **harbor.yml** file.

Its displayed version should match the one you downloaded.

Confirm with **FINISH**

# Install Cloud Native Registry Service (Harbor)

The screenshot shows the 'Supervisor Services' page under 'Workload Management'. The 'Services' tab is highlighted with a red box. A green message box at the top right states: 'Service 'harbor' is successfully registered. You can now install the service on Supervisors.' Below this, there are several service cards:

- VM Service**: Status: Active. Active Versions: 1. Supervisors: 0.
- harbor**: Status: Active. Active Versions: 1. Supervisors: 0. This card is also highlighted with a red box.
- external-dns**: Status: Active. Active Versions: 1. Supervisors: 1.
- contour**: Status: Active. Active Versions: 1. Supervisors: 1.
- Tanzu Kubernetes Grid Service**: Status: Active. Core Service. Active Versions: 1. Supervisors: 1.

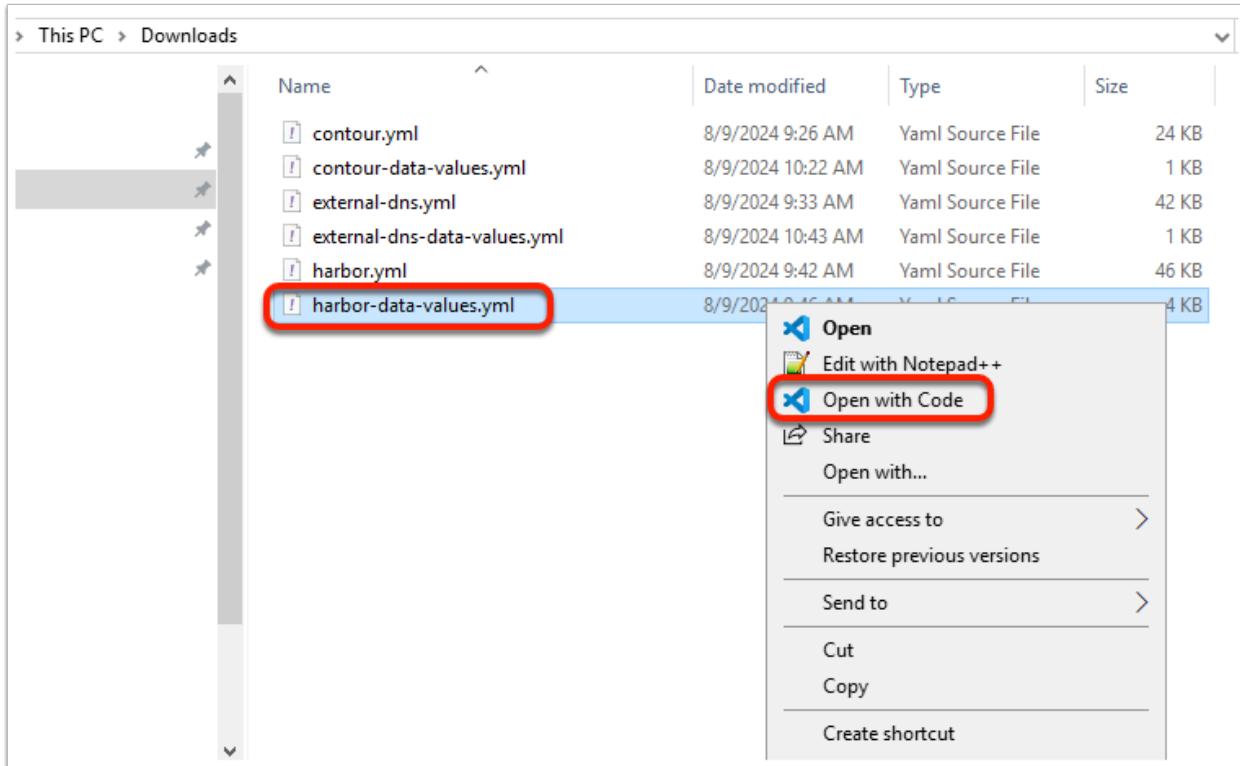
In a while you will see a displayed message for the **harbor** service successfully registered with your **Supervisor**.

You need to activate it with particular configuration settings.



You can close the message in green field.

# Install Cloud Native Registry Service (Harbor)



Let's examine the particular file. You can review/edit it with any available editor

In Windows Explorer, navigate to your **Downloads** folder and locate the recently downloaded file **harbor-data-values.yml**.

Right click and **Open with Code**

For Harbor data values you need to edit a little bit.

# Install Cloud Native Registry Service (Harbor)

```
C:\> Users > Administrator > Downloads > harbor-data-values.yml > ...

1  #! The FQDN for accessing Harbor admin UI and Registry service.
2  hostname: harbor.vcf.holo.lab
3  #! The network port of the envoy service in Contour or other Ingress Controller.
4  port:
5    https: 443
6
7  #! Do not change tlsSecretLabels. It is required for TKG integration to work.
8  tlsCertificate:
9    |  tlsSecretLabels: {"managed-by": "vmware-vRegistry"}
10
11 #! [Required] The initial password of Harbor admin.
12 harborAdminPassword: VMware1!
13
14 #! [Required] The secret key used for encryption. Must be a string of 16 chars.
15 secretKey: 0123456789ABCDEF
16
17 database:
18  #! if external database is used, set "type" to "external"
19  #! and fill the connection information in "external" section
20  type: internal
21  #! [Required] The initial password of the internal postgres database.
22  #! if external database is used, please fill the external.password
23  password: change-it
24
25 core:
26  replicas: 1
27  #! [Required] Secret is used when core server communicates with other components.
28  secret: change-it
29  #! [Required] The XSRF key. Must be a string of 32 chars.
30  xsrfKey: 0123456789ABCDEF0123456789ABCDEF
31 jobservice:
32  replicas: 1
33  #! [Required] Secret is used when job service communicates with other components.
34  secret: change-it
35 registry:
36  replicas: 1
37  #! [Required] Secret is used to secure the upload state from client
38  #! and registry storage backend.
39  #! See: https://github.com/docker/distribution/blob/master/docs/configuration.md#http
40  secret: change-it
41
```

```
C:\> Users > Administrator > Downloads > harbor-data-values.yml > ...

42 #! The persistence is always enabled and a default StorageClass
43 #! is needed in the k8s cluster to provision volumes dynamically.
44 #! Specify another StorageClass in the "storageClass" or set "existingClaim"
45 #! if you have already existing persistent volumes to use
46 #!
47 #! For storing images and charts, you can also use "azure", "gcs", "s3",
48 #! "swift" or "oss". Set it in the "imageChartStorage" section
49 persistence:
50  persistentVolumeClaim:
51    registry:
52      #! Use the existing PVC which must be created manually before bound,
53      #! and specify the "subPath" if the PVC is shared with other components
54      existingClaim: ""
55      #! Specify the "storageClass" used to provision the volume. Or the default
56      #! StorageClass will be used(the default).
57      #! Set it to "" to disable dynamic provisioning
58      storageClass: "vsan-tanzu-storage"
59      subPath: ""
60      accessMode: ReadWriteOnce
61      size: 10Gi
62 jobservice:
63  joblog:
64    existingClaim: ""
65    storageClass: "vsan-tanzu-storage"
66    subPath: ""
67    accessMode: ReadWriteOnce
68    size: 10Gi
69 database:
70  existingClaim: ""
71  storageClass: "vsan-tanzu-storage"
72  subPath: ""
73  accessMode: ReadWriteOnce
74  size: 10Gi
75 redis:
76  existingClaim: ""
77  storageClass: "vsan-tanzu-storage"
78  subPath: ""
79  accessMode: ReadWriteOnce
80  size: 10Gi
81 trivy:
82  existingClaim: ""
83  storageClass: "vsan-tanzu-storage"
84  subPath: ""
85  accessMode: ReadWriteOnce
86  size: 5Gi
87
```

You need to edit accordingly, from the default values:

```
hostname: yourdomain.com  
  
harborAdminPassword: Harbor12345  
  
storageClass: "insert-storage-class-name-here"
```

to new values as following:

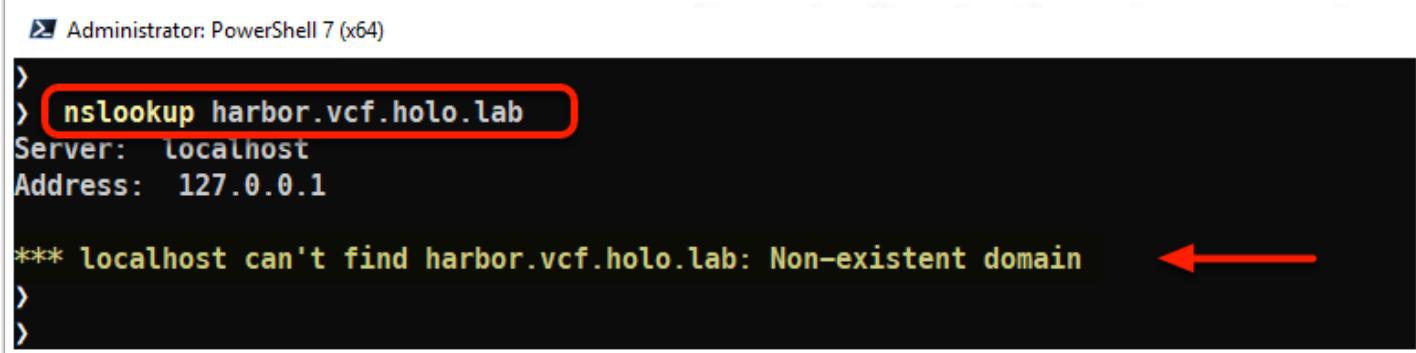
```
hostname: harbor.vcf.holo.lab  
  
harborAdminPassword: VMware1!  
  
storageClass: "vsan-tanzu-storage"
```

Save the file!

Don't close it, you will need the values shortly!

## Install Cloud Native Registry Service (Harbor)

OPTIONAL: Check there is no DNS record for harbor service, available before its install:



```
> Administrator: PowerShell 7 (x64)  
> nslookup harbor.vcf.holo.lab  
Server:  localhost  
Address: 127.0.0.1  
  
*** localhost can't find harbor.vcf.holo.lab: Non-existent domain
```

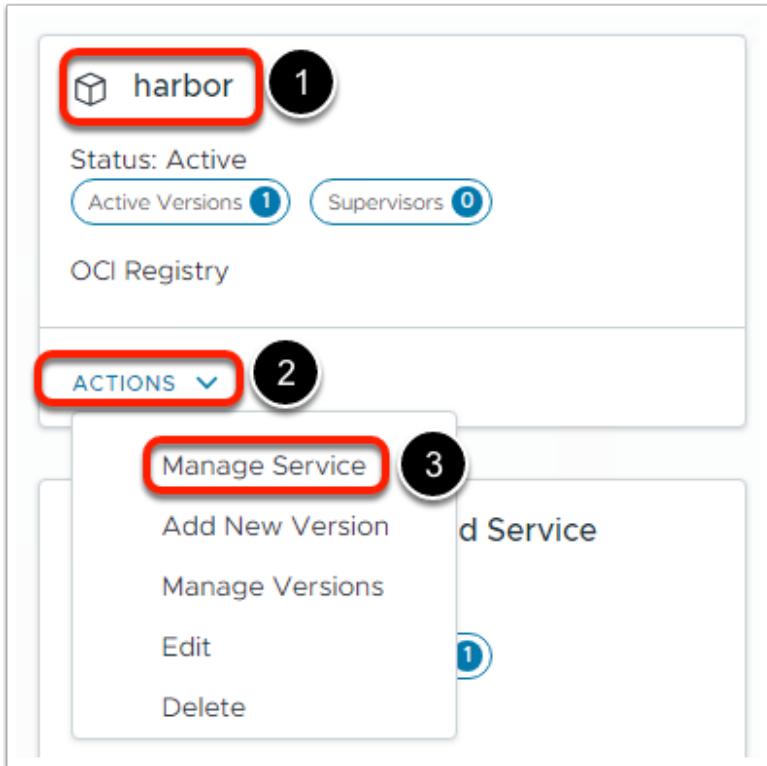
Open a Command Prompt in your Windows desktop

enter the check command

```
nslookup harbor.vcf.holo.lab
```

Once the harbor install is finished, you can check if external-dns has created the dynamic DNS update for it.

# Install Cloud Native Registry Service (Harbor)



Go back to Chrome -> vSphere UI -> Workload Management -> Services

1. Locate your recently registered **harbor** service
2. From **Actions** drop down menu
3. select **Manage Service**

# Install Cloud Native Registry Service (Harbor)

The screenshot shows the 'Manage' wizard in progress, specifically the 'Configure' step. On the left, a sidebar lists '1 Configure' (which is active) and '2 Review'. The main area is titled 'Configure' and contains the instruction: 'Select a version and a supervisor on which to install the service.' Below this, there are two input fields: 'Service Name' set to 'harbor' and 'Install Version' set to '2.9.1+vmware.1-tkg.1'. A table titled 'Supervisors' is displayed, showing a single item: 'lft-dd-supervisor' with status '--'. The entire table row is highlighted with a red circle. At the bottom right of the configuration area, there are 'CANCEL' and 'NEXT' buttons, with the 'NEXT' button also highlighted with a red circle.

In the **Manage** wizard, from **Configure** select your **Supervisor** and confirm **NEXT**

# Install Cloud Native Registry Service (Harbor)

The screenshot shows a software interface for managing cloud native registry services. On the left, a vertical sidebar labeled 'Manage' contains two items: '1 Configure' and '2 Review'. The '2 Review' item is highlighted with a dark background and white text. The main area is titled 'Review' and displays the following information:

Service Name	harbor
Version to install	2.9.1+vmware.1-tkg.1
Supervisor	lf-tdd-supervisor

Below this, there is a section titled 'YAML Service Config (optional)' which contains a large, empty text area. A red box highlights this text area. At the bottom right of the review screen, there are three buttons: 'CANCEL' (light blue), 'BACK' (light blue), and 'FINISH' (green).

You need to provide the service configuration values.

Copy these from the Visual Studio Code, where you lastly reviewed **harbor-data-values.yml**

# Install Cloud Native Registry Service (Harbor)

Manage

Review

Selected service version and Supervisor are compatible.

Service Name: harbor  
Version to install: 2.9.1+vmware.1-tkg.1  
Supervisor: lf-tdd-supervisor

YAML Service Config (optional)

```
1 #! The FQDN for accessing Harbor admin UI and Registry service.
2 hostname: harbor.vcf.holo.lab
3 #! The network port of the Envoy service in Contour or other Ingress Controller.
4 port:
5   https: 443
6
7 #! Do not change tlsSecretLabels. It is required for TKG integration to work.
8 tlsCertificate:
9   tlsSecretLabels: {"managed-by": "vmware-vRegistry"}
10
11 #! [Required] The initial password of Harbor admin.
12 harborAdminPassword: VMware1!
13
14 #! [Required] The secret key used for encryption. Must be a string of 16 chars.
15 secretKey: 0123456789ABCDEF
16
17 database:
18   #! if external database is used, set "type" to "external"
19   #! and fill the connection information in "external" section
20   type: internal
21   #! [Required] The initial password of the internal postgres database.
```

CANCEL BACK FINISH

Select all and paste into the **YAML Service Config** section in the UI wizard.

Confirm with **FINISH**

# Install Cloud Native Registry Service (Harbor)

Workload Management

Namespaces Supervisors Services Updates

ADD SUPERVISOR DEACTIVATE CLONE CONFIG EXPORT CONFIG EXPORT LOGS RESTORE

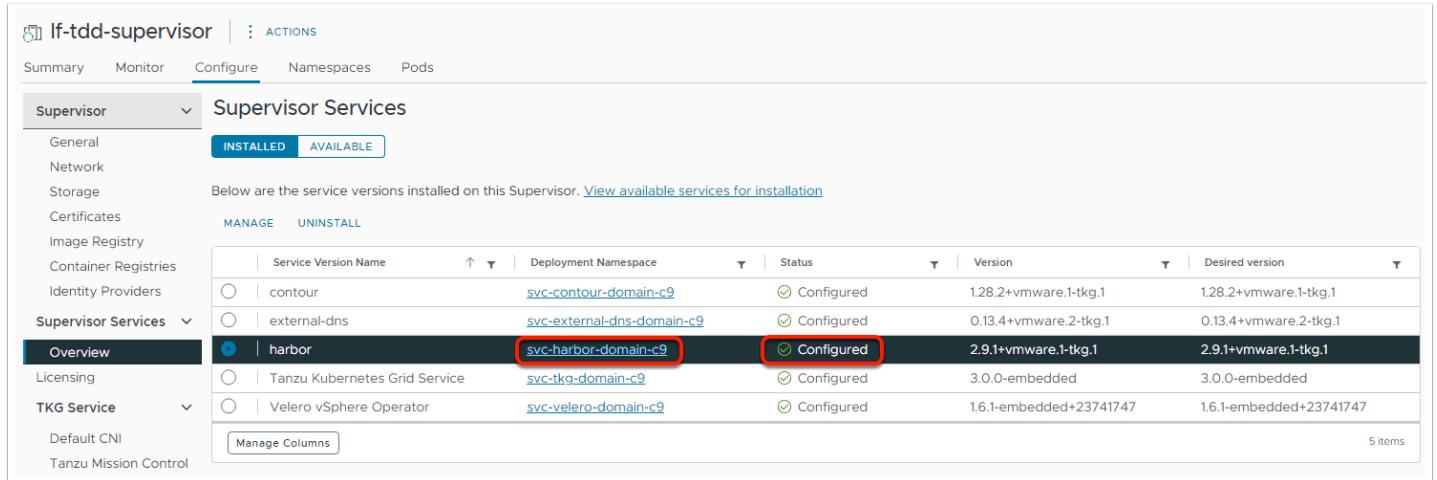
Quick Filter Enter value

Supervisor	Namespaces	Hosts	Services	Config Status	Host Config Status
lf-tdd-supervisor	5	4	View	Running	Running

Go back in the **Workload Management** section

Select **Supervisors** and click on **View** under **Services**

# Install Cloud Native Registry Service (Harbor)

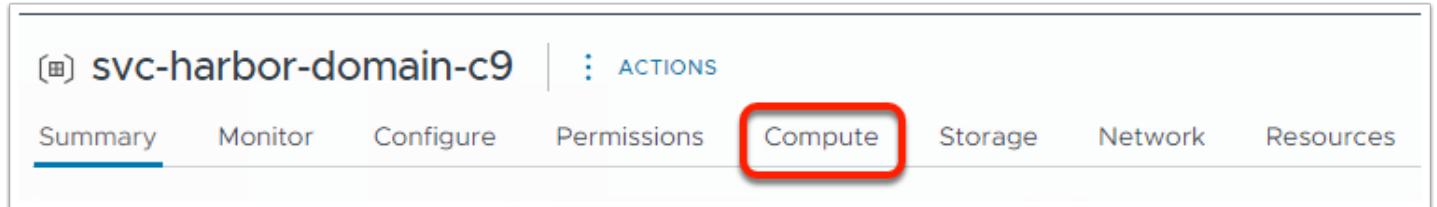


The screenshot shows the Tanzu Management UI for the 'If-tdd-supervisor' cluster. Under the 'Supervisor Services' tab, the 'harbor' service is selected. The table shows the following details:

Service Version Name	Deployment Namespace	Status	Version	Desired version
contour	svc-contour-domain-c9	Configured	1.28.2+vmware.1-tkg.1	1.28.2+vmware.1-tkg.1
external-dns	svc-external-dns-domain-c9	Configured	0.13.4+vmware.2-tkg.1	0.13.4+vmware.2-tkg.1
<b>harbor</b>	<b>svc-harbor-domain-c9</b>	<b>Configured</b>	<b>2.9.1+vmware.1-tkg.1</b>	<b>2.9.1+vmware.1-tkg.1</b>
Tanzu Kubernetes Grid Service	svc-tkg-domain-c9	Configured	3.0.0-embedded	3.0.0-embedded
Velero vSphere Operator	svc-velero-domain-c9	Configured	1.6.1-embedded+23741747	1.6.1-embedded+23741747

In a couple of minutes the **harbor** service Status will change from Configuring to **Configured**  
Click on the **svc-harbor-domain-c9** under **Deployment Namespace**

# Install Cloud Native Registry Service (Harbor)



The screenshot shows the Tanzu Management UI for the 'svc-harbor-domain-c9' namespace. The 'Compute' tab is highlighted with a red box.

For the **svc-harbor-domain-c9** Namespace, click on **Compute**

# Install Cloud Native Registry Service (Harbor)

The screenshot shows the vSphere Web Client interface for the cluster `svc-harbor-domain-c9`. The **Compute** tab is active. In the **Pods** section, the **Pods** tab is selected, indicated by a red box. A table lists several pods, all of which are running:

Name	YAML	Phase	Creation Date	Cluster IP	Containers	Namespace	vSphere Pod
harbor-core-69d47f944b-d2vp7	<a href="#">View YAML</a>	Running	Aug 9, 2024, 11:34:31 AM	10.80.0.2	1/1	svc-harbor-domain-c9	Yes
harbor-database-0	<a href="#">View YAML</a>	Running	Aug 9, 2024, 11:34:32 AM	10.80.0.2	1/1	svc-harbor-domain-c9	Yes
harbor-exporter-7c856bd68c-5lnbk	<a href="#">View YAML</a>	Running	Aug 9, 2024, 11:34:32 AM	10.80.0.2	1/1	svc-harbor-domain-c9	Yes
harbor-jobservice-65bf598447-mjhzw	<a href="#">View YAML</a>	Running	Aug 9, 2024, 11:34:32 AM	10.80.0.2	1/1	svc-harbor-domain-c9	Yes
harbor-portal-68fc674f96-4n7hq	<a href="#">View YAML</a>	Running	Aug 9, 2024, 11:34:33 AM	10.80.0.2	1/1	svc-harbor-domain-c9	Yes
harbor-redis-0	<a href="#">View YAML</a>	Running	Aug 9, 2024, 11:34:34 AM	10.80.0.2	1/1	svc-harbor-domain-c9	Yes
harbor-registry-579fb785d-pxk96	<a href="#">View YAML</a>	Running	Aug 9, 2024, 11:34:34 AM	10.80.0.2	2/2	svc-harbor-domain-c9	Yes
harbor-trivy-0	<a href="#">View YAML</a>	Running	Aug 9, 2024, 11:34:36 AM	10.80.0.2	1/1	svc-harbor-domain-c9	Yes

Under **Compute**, you can review all the **Kubernetes Core** configurations for harbor.

Select **Pods** and you will see all the **harbor** needed pods, running as **vSphere Pods**

# Install Cloud Native Registry Service (Harbor)

The screenshot shows the vSphere Web Client interface for the cluster `svc-harbor-domain-c9`. The **Compute** tab is active. In the **Deployments** section, the **Deployments** tab is selected, indicated by a red box. A table lists five deployments, all of which are currently at version 1:

Name	YAML	Desired	Current	Up To Date	Available	Creation Date
harbor-core	<a href="#">View YAML</a>	1	1	1	1	Aug 9, 2024, 11:34:31 AM
harbor-exporter	<a href="#">View YAML</a>	1	1	1	1	Aug 9, 2024, 11:34:31 AM
harbor-jobservice	<a href="#">View YAML</a>	1	1	1	1	Aug 9, 2024, 11:34:32 AM
harbor-portal	<a href="#">View YAML</a>	1	1	1	1	Aug 9, 2024, 11:34:33 AM
harbor-registry	<a href="#">View YAML</a>	1	1	1	1	Aug 9, 2024, 11:34:34 AM

Select **Deployments** and you will see all the **harbor** needed deployments in place

# Install Cloud Native Registry Service (Harbor)

Name	YAML	Desired	Current	Ready	Creation Date
harbor-core-69d47f994b	<a href="#">View YAML</a>	1	1	1	Aug 9, 2024, 11:34:31 AM
harbor-exporter-7c856bd68c	<a href="#">View YAML</a>	1	1	1	Aug 9, 2024, 11:34:31 AM
harbor-jobservice-65bf598447	<a href="#">View YAML</a>	1	1	1	Aug 9, 2024, 11:34:32 AM
harbor-portal-68fc674f96	<a href="#">View YAML</a>	1	1	1	Aug 9, 2024, 11:34:33 AM
harbor-registry-579ffbb785d	<a href="#">View YAML</a>	1	1	1	Aug 9, 2024, 11:34:34 AM

Select **Replica Sets** and you will see all the **harbor** needed replica sets in place

# Install Cloud Native Registry Service (Harbor)

Name	YAML	Desired	Current	Creation Date
harbor-database	<a href="#">View YAML</a>	1	1	Aug 9, 2024, 11:34:31 AM
harbor-redis	<a href="#">View YAML</a>	1	1	Aug 9, 2024, 11:34:34 AM
harbor-trivy	<a href="#">View YAML</a>	1	1	Aug 9, 2024, 11:34:35 AM

Select **Stateful Sets** and you will see all the **harbor** needed stateful sets in place: for harbor-database, harbor-redis and harbor-trivy

Since the harbor service got installed, let's check if external-dns had provided a dynamic update for it in DNS

# Install Cloud Native Registry Service (Harbor)

```
>
> nslookup harbor.vcf.holo.lab
Server: localhost
Address: 127.0.0.1

*** localhost can't find harbor.vcf.holo.lab: Non-existent domain
>
>
> nslookup harbor.vcf.holo.lab
Server: localhost
Address: 127.0.0.1

Name:   harbor.vcf.holo.lab      ←
Address: 10.80.0.3
>
>
```

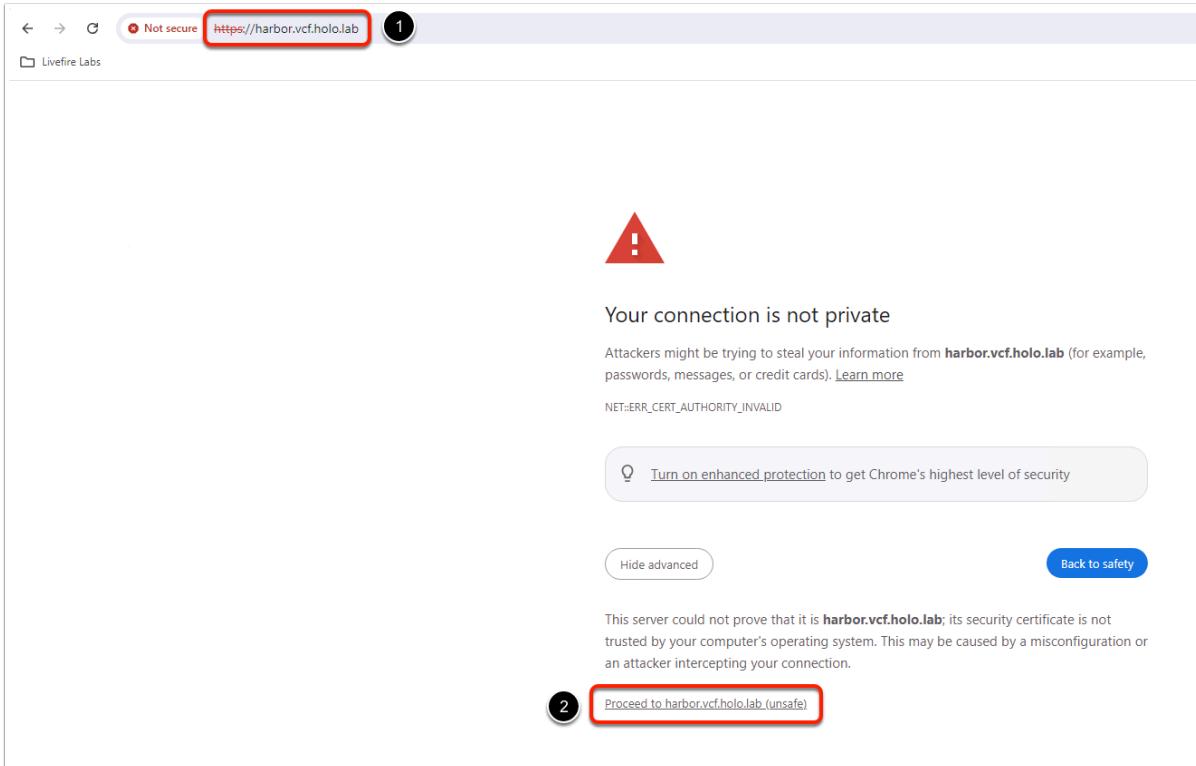
Go back to your Windows command prompt box and check again the resolution

```
nslookup harbor.vcf.holo.lab
```

Looks good!

Now, let's get to the harbor UI

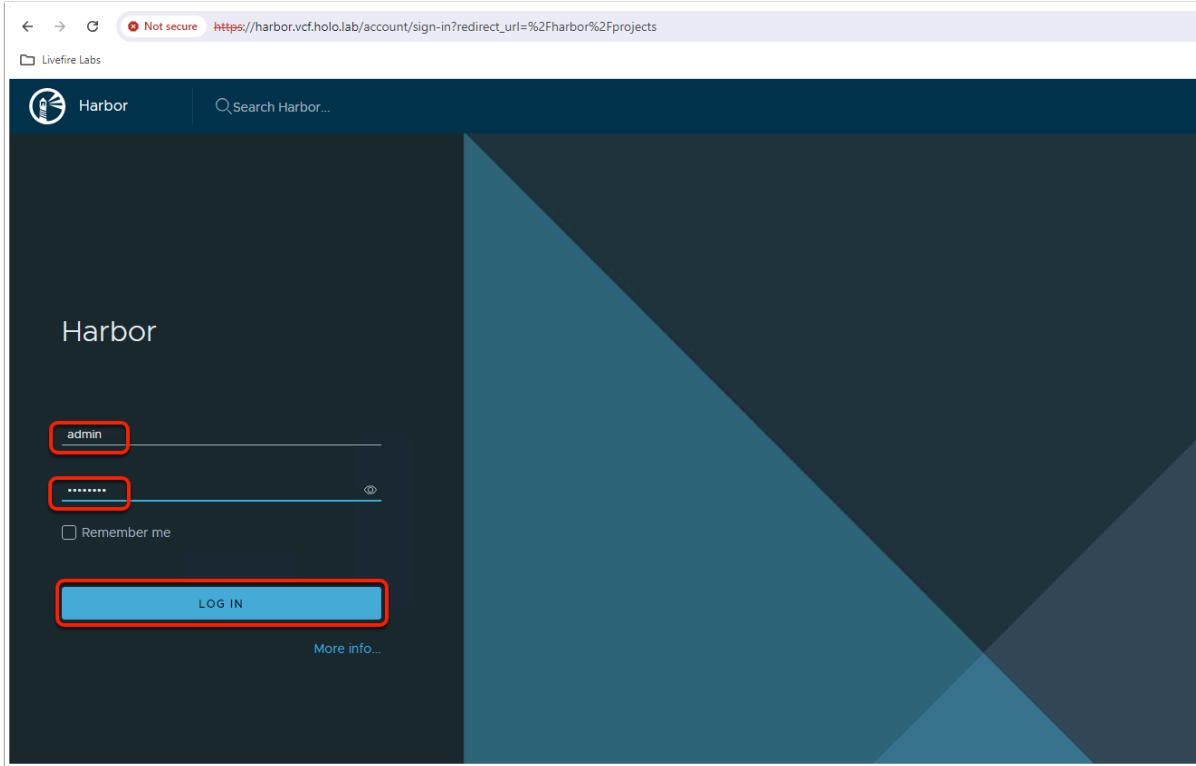
# Install Cloud Native Registry Service (Harbor)



Open a **new** tab in **Chrome** and:

1. Go to <https://harbor.vcf.holo.lab>
2. Click on **Proceed**

# Install Cloud Native Registry Service (Harbor)



In Harbor Log In page, Populate the credentials and click on **LOG IN**

```
user: admin  
password: VMware1!
```

The default scanner has been installed. To install other scanners refer to the [documentation](#).

Project Name	Access Level	Role	Type	Repositories Count
library	Public	Project Admin	Project	0

The Harbor is ready!

## Install (Local) Consumption Interface

Provides the Local Consumption Interface (LCI) for Namespaces within vSphere Client. This also includes the Single Sign On (SSO) component required by the Cloud Consumption Interface (CCI) in Aria Automation within VMware Cloud Foundation.

The minimum required version for using this interface is vSphere 8 Update 3.

## Consumption Interface

Provides the Local Consumption Interface (LCI) for Namespaces within vSphere Client. This also includes the Single Sign On (SSO) component required by the Cloud Consumption Interface (CCI) in Aria Automation within VMware Cloud Foundation.

The minimum required version for using this interface is vSphere 8 Update 3.

### Consumption Interface Versions

Installation instructions can be found [here in VMware documentation](#).

NOTE: Occasionally, the plug-in might fail to load on the initial attempt. To check if the plug-in has loaded correctly, click the vSphere Client menu icon, then to Administration -> Client -> Plug-ins. Check the Status column of the Namespace UI plug-in, and in case you see a "Plug-in configuration with Reverse Proxy failed." Message, reinstall the plug-in.

Download latest version:

- [Consumption Interface v1.0.1](#)
- [Release notes](#)

### OSS information

LCI OSS

SSO OSS Refer to the Open Source Tab

Download the latest available manifests for Consumption Interface to your **Downloads** folder

cci-supervisor-service.yml

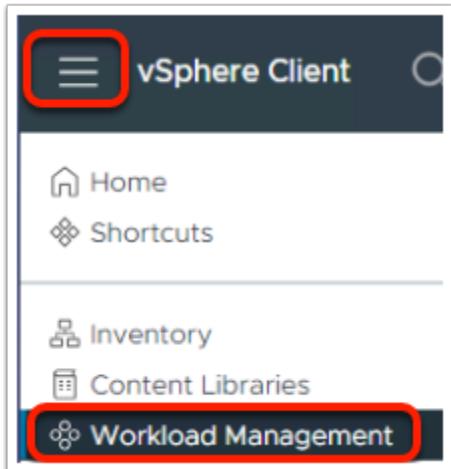
<https://vsphere-tmm.github.io/Supervisor-Services/#consumption-interface>

## Install (Local) Consumption Interface

PC > Windows (C:) > Users > Administrator > Downloads					Search Downloads
	Name	Date modified	Type	Size	
	cci-supervisor-service.yml	8/10/2024 12:01 AM	Yaml Source File	2 KB	
	contour.yml	8/9/2024 11:08 PM	Yaml Source File	24 KB	
	contour-data-values.yml	8/9/2024 11:12 PM	Yaml Source File	1 KB	
	external-dns.yml	8/9/2024 11:13 PM	Yaml Source File	42 KB	
	external-dns-data-values.yml	8/9/2024 10:42 PM	Yaml Source File	1 KB	
	harbor.yml	8/9/2024 11:13 PM	Yaml Source File	46 KB	
	harbor-data-values.yml	8/9/2024 11:38 PM	Yaml Source File	4 KB	

You are ready to deploy Local Consumption Interface as a Supervisor service!

# Install (Local) Consumption Interface



From the vSphere Client home menu, select **Workload Management**.

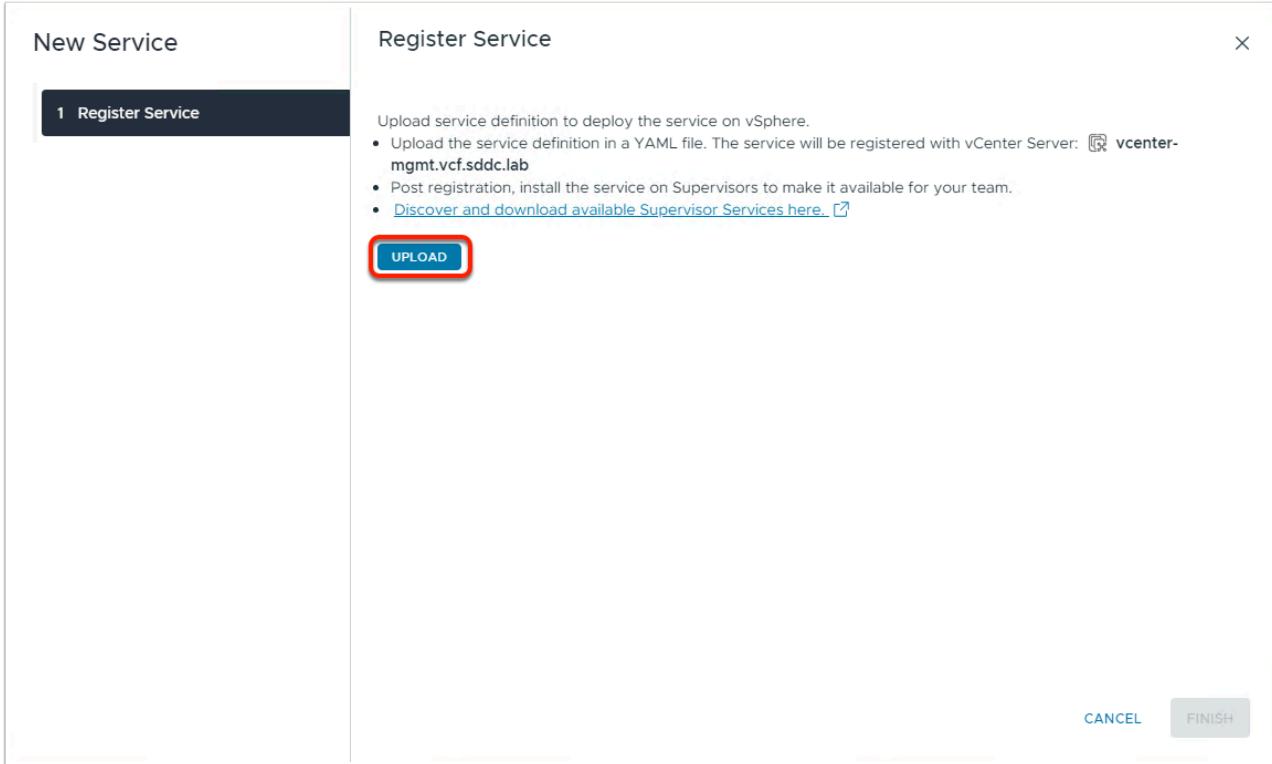
# Install (Local) Consumption Interface

A screenshot of the 'Workload Management' section in the vSphere Client. At the top, there are tabs for 'Namespaces', 'Supervisors', 'Services' (which is highlighted with a red square), and 'Updates'. Below that, it says 'Supervisor Services' and shows a dropdown for 'VCENTER-MGMT.VCF SDDC.LAB'. A note says 'Supervisor Services is a platform for managing core infrastructure components, such as virtual machines. Application teams are able to deploy instances of Supervisor Services within their own Namespaces using industry standard tools and practices.' Below this, there's a 'Sort By:' dropdown set to 'Recently added'. A note says 'Below are the services registered to this vCenter Server system. You can manage services with multiple versions from the same service card.' There are several service cards: 'Add New Service' (with an 'ADD' button highlighted with a red square), 'VM Service', 'harbor', 'external-dns', 'contour', and 'Tanzu Kubernetes Grid Service'. Each card shows its status (e.g., Active), active versions, supervisors, and a brief description.

In the **Workload Management** section, select **Services**.

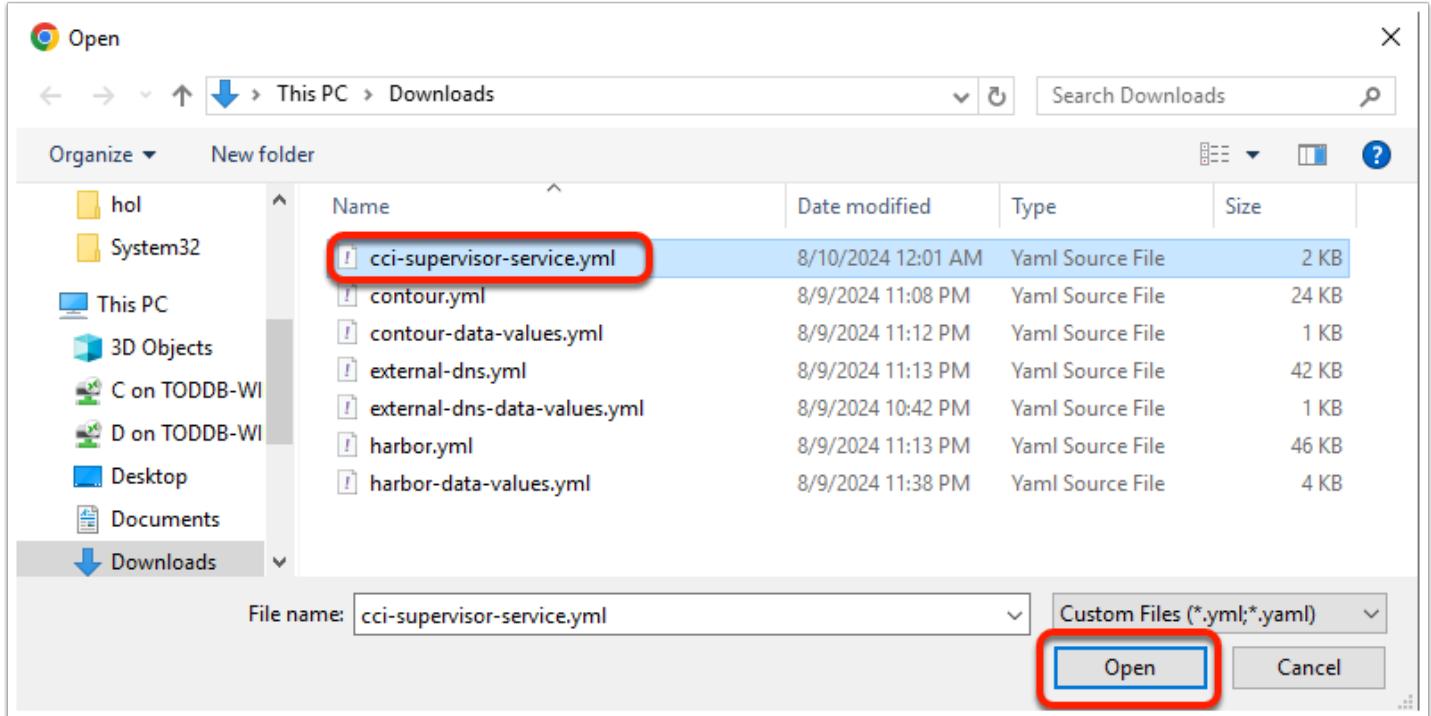
Locate the **Add New Service** card and click on **ADD**

# Install (Local) Consumption Interface



In the **New Service** wizard, for **Register Service** select **UPLOAD**

# Install (Local) Consumption Interface



In the Chrome **Open** menu go to **Downloads** folder and select the **cci-supervisor-service.yml** you downloaded recently.

Confirm with clicking on **Open** button

# Install (Local) Consumption Interface

New Service

1 Register Service

Register Service

⚠️ Running 3rd party services on user workloads has security risks. A 3rd party service has network access to user workloads, vSphere Pods, and exposed APIs.

ℹ️ YAML was uploaded successfully. Note: YAML content is not verified and could fail during installation into a Supervisor.

Upload service definition to deploy the service on vSphere.

YAML File details [Upload new](#)  
[cci-supervisor-service.yml](#)

Service Details

vCenter Server	vcenter-mgmt.vcf.sddc.lab
Service Name	Consumption Interface
Service ID	cci-service.flng.vsphere.vmware.com
Service Description	Provides the Local Consumption Interface (LCI) for Namespaces within vSphere Client. This also includes the Single Sign On (SSO) component required by the Cloud Consumption Interface (CCI) in Aria Automation within VMware Cloud Foundation.
Version	1.0.1

CANCEL FINISH

In the **New Service** wizard, for **Register Service** make sure you can see the aforementioned **cci-supervisor-service.yml** file.

Its displayed version should match the one you downloaded.

Confirm with **FINISH**

# Install (Local) Consumption Interface

The screenshot shows the 'Workload Management' interface in vSphere. The 'Services' tab is selected (highlighted with a red box). A message at the top right says: 'Service 'Consumption Interface' is successfully registered. You can now install the service on Supervisors.' Below this, there are several service cards:

- VM Service**: This service allows developers to self-service VMs and allows you to set policies for VM deployment.
- Consumption Interface**: Status: Active. Active Versions: 1. Supervisors: 0. Provides the Local Consumption Interface (LCI) f...
- harbor**: Status: Active. Active Versions: 1. Supervisors: 1. OCI Registry.
- external-dns**: Status: Active. Active Versions: 1. Supervisors: 1. This package provides DNS synchronization fun...
- contour**: Status: Active. Active Versions: 1. Supervisors: 1. An ingress controller.

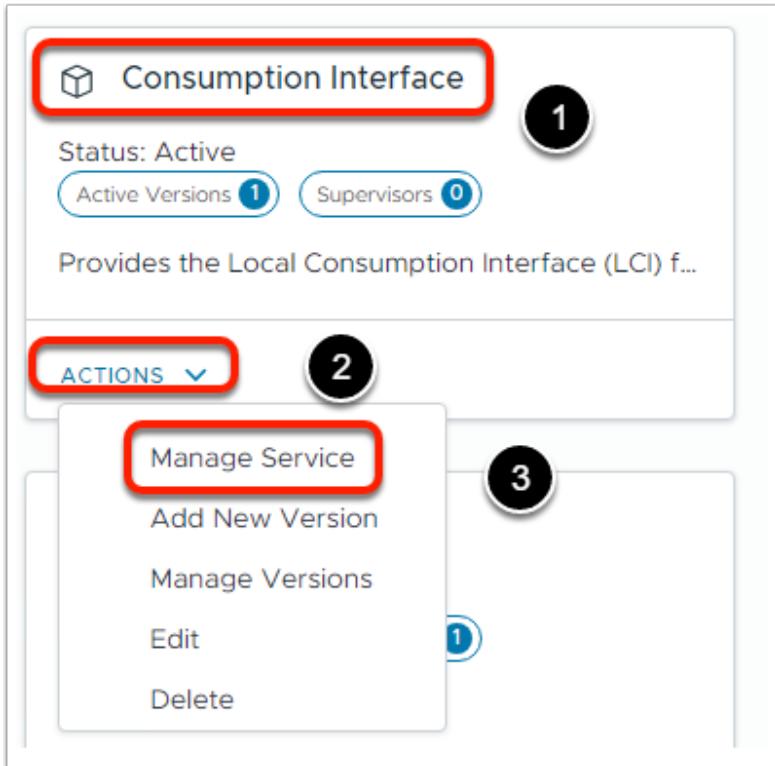
In a while you will see a displayed message for the **Consumption Interface** service successfully registered with your **Supervisor**.

You need to activate it with particular configuration settings.



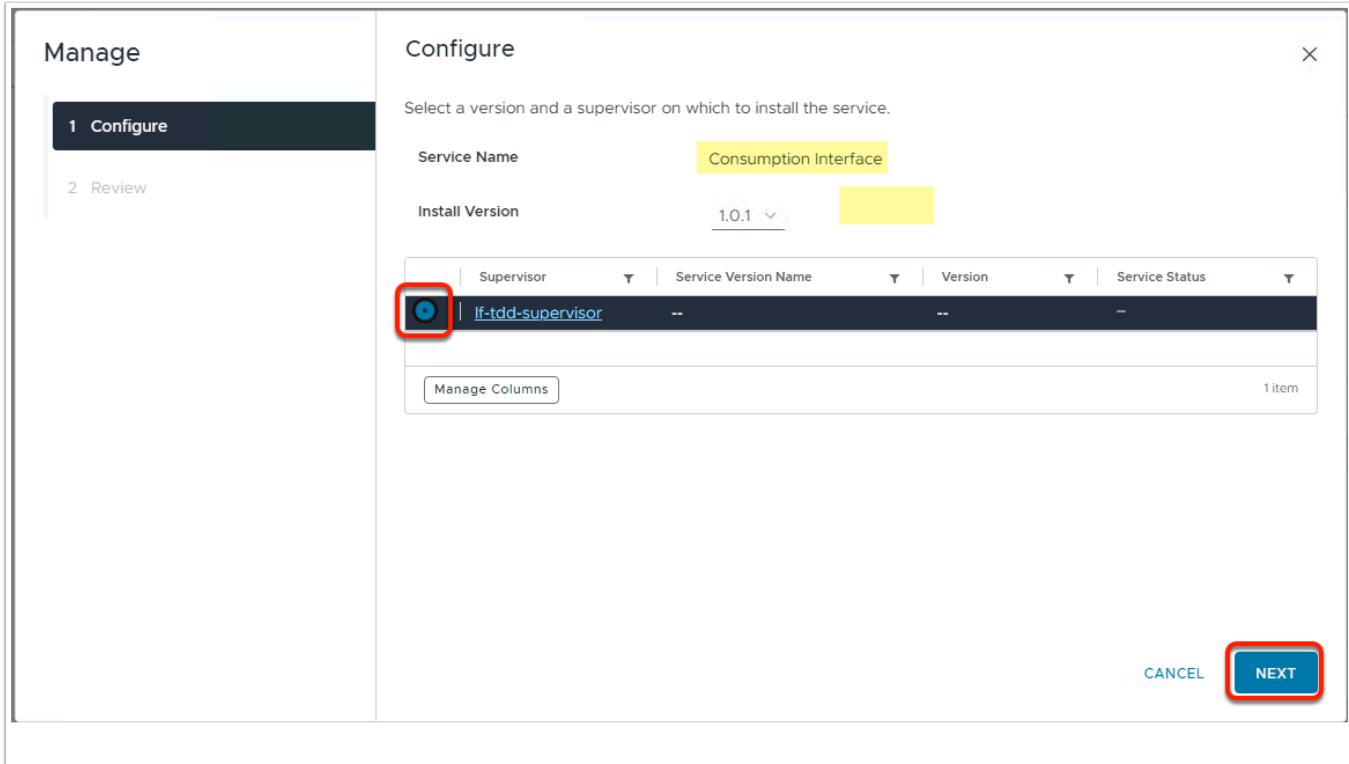
You can close the message in green field.

# Install (Local) Consumption Interface



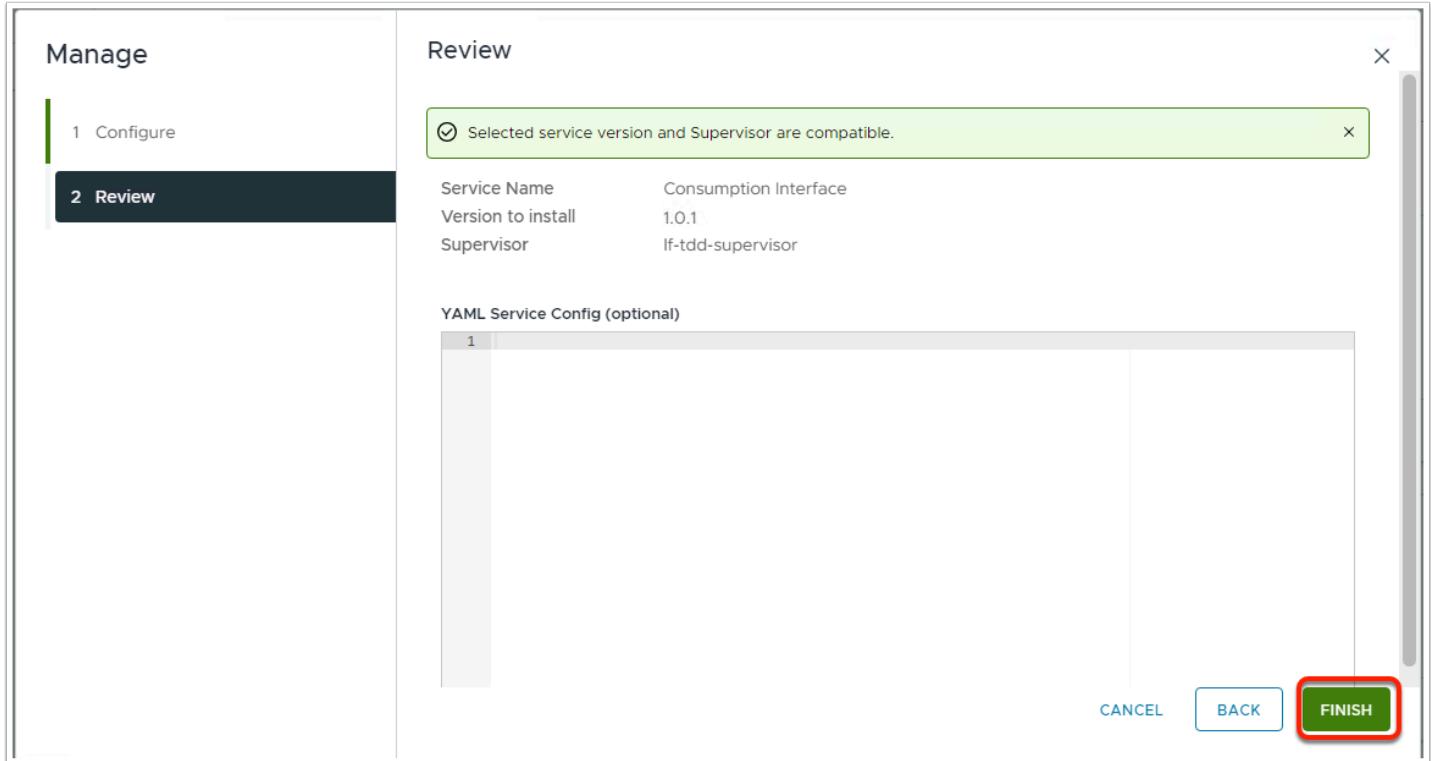
1. Locate your recently registered **Consumption Interface** service
2. From **Actions** drop down menu
3. select **Manage Service**

# Install (Local) Consumption Interface



In the **Manage** wizard, from **Configure** select your **Supervisor** and confirm **NEXT**

# Install (Local) Consumption Interface



No data file to use

Click **FINISH**

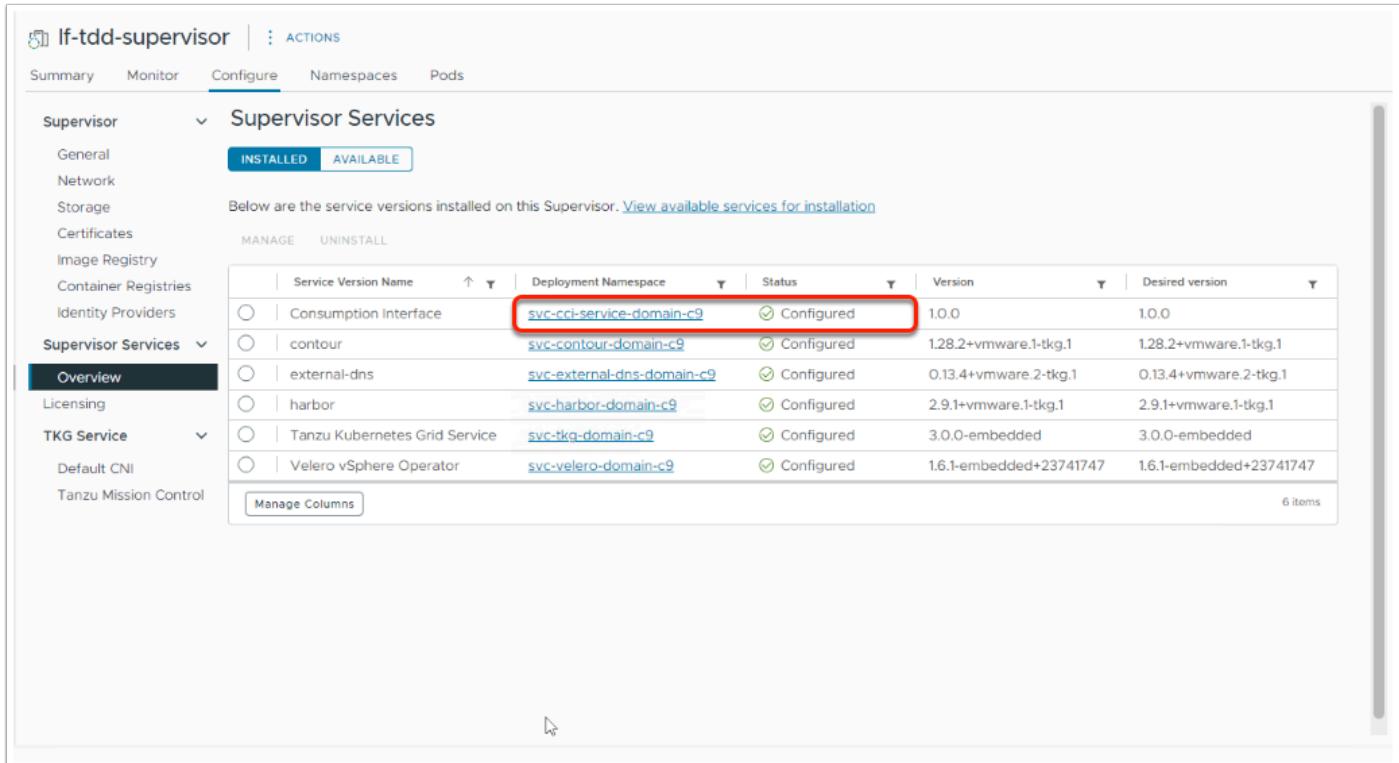
# Install (Local) Consumption Interface

The screenshot shows the 'Workload Management' interface with the 'Supervisors' tab selected. A red box highlights the 'Supervisors' tab. Below it, a table lists a supervisor named 'lf-tdd-supervisor'. A red box highlights the 'View' button under the Services column for this supervisor. Other columns include Supervisor, Namespaces, Hosts, Services, Config Status, and Host Config Status.

Go back in the **Workload Management** section

Select **Supervisors** and click on **View** under **Services**

# Install (Local) Consumption Interface



The screenshot shows the Tanzu Management UI for a supervisor named 'lf-tdd-supervisor'. The 'Supervisor Services' section is open, displaying a list of installed services. The 'Consumption interface' service is listed with the following details:

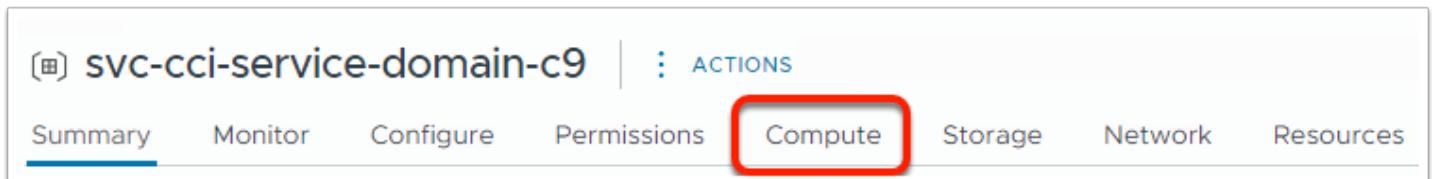
Service Version Name	Deployment Namespace	Status	Version	Desired version
Consumption interface	svc-cci-service-domain-c9	Configured	1.0.0	1.0.0
contour	svc-contour-domain-c9	Configured	1.28.2+vmware.1-tkg.1	1.28.2+vmware.1-tkg.1
external-dns	svc-external-dns-domain-c9	Configured	0.13.4+vmware.2-tkg.1	0.13.4+vmware.2-tkg.1
harbor	svc-harbor-domain-c9	Configured	2.9.1+vmware.1-tkg.1	2.9.1+vmware.1-tkg.1
Tanzu Kubernetes Grid Service	svc-tkg-domain-c9	Configured	3.0.0-embedded	3.0.0-embedded
Velero vSphere Operator	svc-velero-domain-c9	Configured	1.6.1-embedded+23741747	1.6.1-embedded+23741747

A red box highlights the row for 'Consumption interface' under the 'Status' column, which is labeled 'Configured'.

In a couple of minutes the **Consumption Interface** service Status will change from Configuring to **Configured**

Click on the **svc-cci-service-domain-c9** under Deployment Namespace

# Install (Local) Consumption Interface



The screenshot shows the Tanzu Management UI for the 'svc-cci-service-domain-c9' namespace. The 'Compute' tab is highlighted with a red box.

For the **svc-cci-service-domain-c9** Namespace, click on **Compute**

# Install (Local) Consumption Interface

Name	YAML	Phase	Creation Date	Cluster IP	Containers	Namespace
cci-ns-controller-manager-7bcf978dc7-8vfb6z	<a href="#">View YAML</a>	Running	Aug 10, 2024, 1:30:27 AM	10.80.0.2	1/1	svc-cci-service-n-c9
cci-service-7bc74fb5b6-dfbfr	<a href="#">View YAML</a>	Running	Aug 10, 2024, 1:30:27 AM	10.80.0.2	1/1	svc-cci-service-n-c9
masterproxy-cci-ns-plugin-9rt2f	<a href="#">View YAML</a>	Running	Aug 10, 2024, 1:30:38 AM	10.80.0.2	1/1	svc-cci-service-n-c9
masterproxy-cci-ns-plugin-zqngq	<a href="#">View YAML</a>	Running	Aug 10, 2024, 1:30:38 AM	10.80.0.2	1/1	svc-cci-service-n-c9
masterproxy-cci-ns-plugin-zsjs	<a href="#">View YAML</a>	Running	Aug 10, 2024, 1:30:37 AM	10.80.0.2	1/1	svc-cci-service-n-c9

Under **Compute**, you can review all the **Kubernetes Core** configurations for **Consumption Interface**.

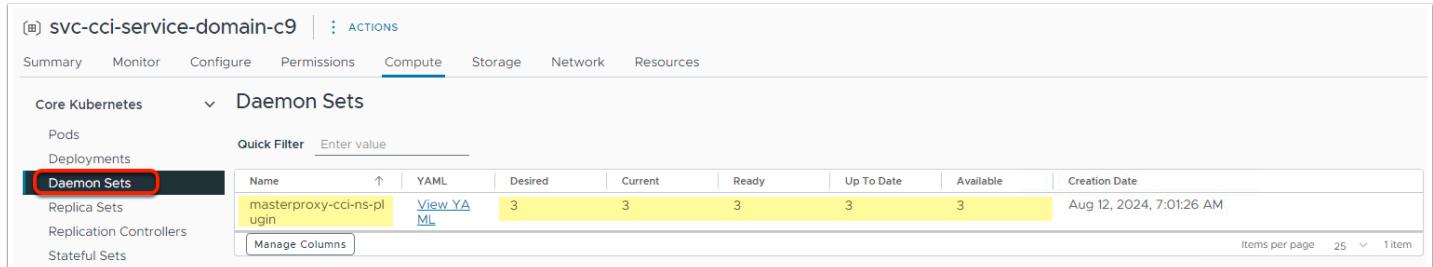
Select **Pods** and you will see all the **Consumption Interface** needed pods, running as **vSphere Pods**

# Install (Local) Consumption Interface

Name	Desired	Current	Up To Date	Available	Creation Date
cci-ns-controller-manager	1	1	1	1	Aug 10, 2024, 1:30:26 AM
cci-service	1	1	1	1	Aug 10, 2024, 1:30:26 AM

Select **Deployments** and you will see all the **Consumption Interface** needed deployments in place

# Install (Local) Consumption Interface



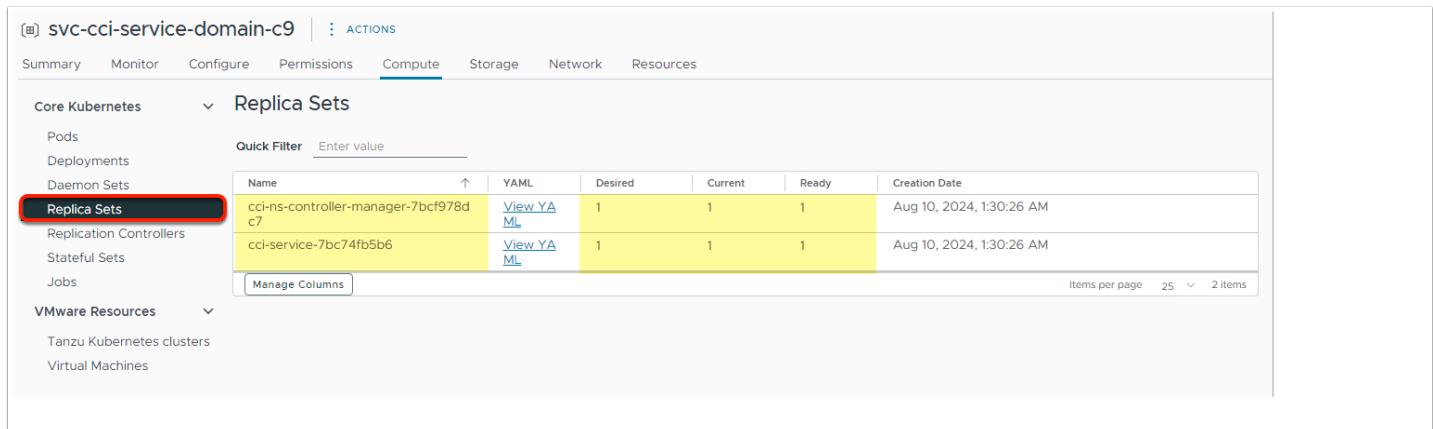
The screenshot shows the Tanzu Kubernetes Service (TKS) dashboard. The top navigation bar includes Summary, Monitor, Configure, Permissions, Compute (selected), Storage, Network, and Resources. Under Core Kubernetes, the Daemon Sets section is selected. A table lists a single entry: masterproxy-cci-ns-plug-in, with columns for Name, YAML, Desired, Current, Ready, Up To Date, Available, and Creation Date. The creation date is Aug 12, 2024, 7:01:26 AM. The table has 8 columns and 1 row.

Name	YAML	Desired	Current	Ready	Up To Date	Available	Creation Date
masterproxy-cci-ns-plug-in	<a href="#">View YAML</a>	3	3	3	3	3	Aug 12, 2024, 7:01:26 AM

Items per page: 25 ▾ 1 item

Select **Daemon Sets** and you will see all the **Consumption Interface** needed replica sets in place

# Install (Local) Consumption Interface



The screenshot shows the Tanzu Kubernetes Service (TKS) dashboard. The top navigation bar includes Summary, Monitor, Configure, Permissions, Compute (selected), Storage, Network, and Resources. Under Core Kubernetes, the Replica Sets section is selected. A table lists two entries: cci-ns-controller-manager-7bcf978dc7 and cci-service-7bc74fb5b6, with columns for Name, YAML, Desired, Current, Ready, and Creation Date. The creation dates are Aug 10, 2024, 1:30:26 AM. The table has 6 columns and 2 rows.

Name	YAML	Desired	Current	Ready	Creation Date
cci-ns-controller-manager-7bcf978dc7	<a href="#">View YAML</a>	1	1	1	Aug 10, 2024, 1:30:26 AM
cci-service-7bc74fb5b6	<a href="#">View YAML</a>	1	1	1	Aug 10, 2024, 1:30:26 AM

Items per page: 25 ▾ 2 items

Select **Replica Sets** and you will see all the **Consumption Interface** needed replica sets in place

# Install (Local) Consumption Interface

## ! IMPORTANT for LCI : Local Consumption Interface

- LCI is not supported on Multi-Zone Supervisors.
- After installing the service into a supervisor, you should expect a delay while the plugin is installed. If you do not see the plugin listed in the Resources tab under Workload Management -> Namespaces -> your namespace, please wait and then refresh the page.

- After installing the service into a supervisor, **you will need to reload the web page or login in a new window to allow the plugin to load properly.**

① Plugin VMware Namespace UI:1.0.0.0 has been successfully deployed. Refresh the browser to enable. **REFRESH BROWSER**

② svc-tkg-domain-c9

③ Resources

- Reload/Refresh** the vSphere client in your browser
- Under **Namespaces** section go to **svc-tkg-domain-c9**
- Notice the new plugin installed under the **Resources** tab

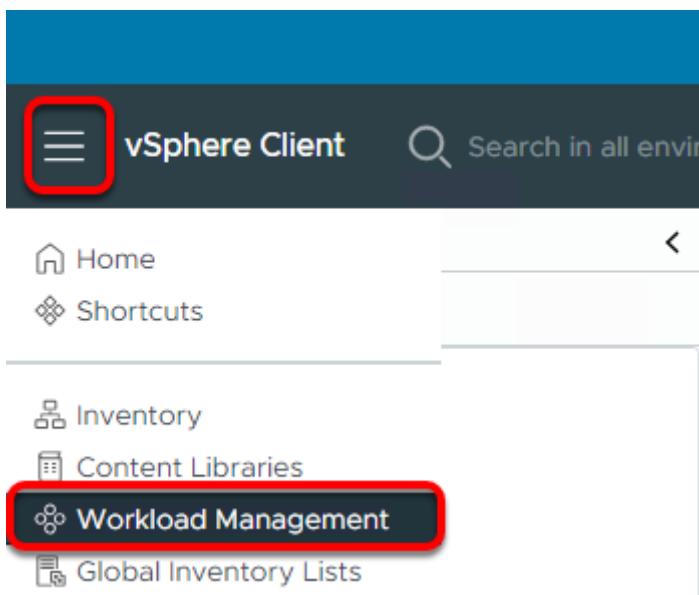
You are now ready to proceed to the next section!

# Create and Configure a Namespace

A vSphere Namespace sets the resource boundaries where vSphere Pods, VMs, and TKG clusters can run. As a vSphere administrator, you create and configure vSphere Namespaces through the vSphere Client.

When initially created, a vSphere Namespace has unlimited resources within the Supervisor. As a vSphere administrator, you can set limits for CPU, memory, storage, as well as the number of Kubernetes objects that can run within the vSphere Namespace. Storage limitations are represented as storage quotas in Kubernetes. A resource pool is created in vSphere per each vSphere Namespace on the Supervisor.

## Use the vSphere client to Create a Namespace



Click the Hamburger Menu in vCenter

Click Workload Management

# Use the vSphere client to Create a Namespace

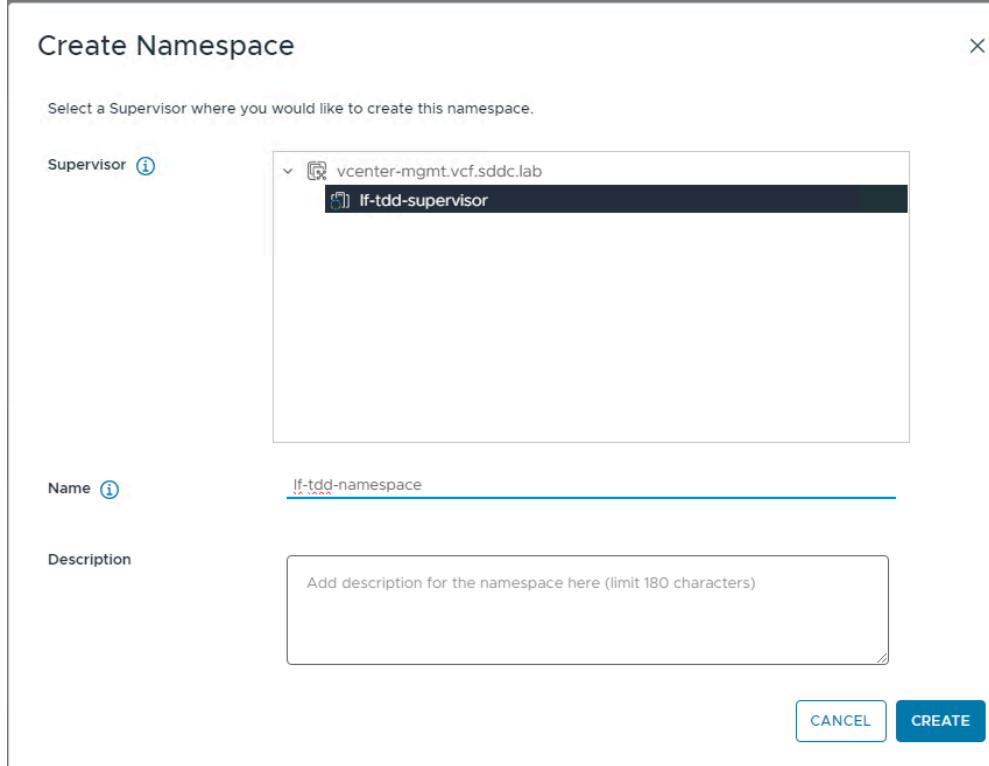
The screenshot shows the vSphere Client interface with the title "Workload Management". On the left, there is a sidebar with a list of namespaces: svc-cci-service-domain-c9, svc-contour-domain-c9, svc-external-dns-domain-c9, svc-harbor-domain-c9, svc-tkg-domain-c9, and svc-velero-domain-c9. The "Namespaces" tab is selected, and the "NEW NAMESPACE" button is highlighted with a red box. Below the table, there is a "Quick Filter" input field with the placeholder "Enter value".

	Namespaces	Supervisor	Config Status
○	<a href="#">svc-cci-service-domain-c9</a>	<a href="#">lf-tdd-supervisor</a>	<span>✓ Running</span>
○	<a href="#">svc-contour-domain-c9</a>	<a href="#">lf-tdd-supervisor</a>	<span>✓ Running</span>
○	<a href="#">svc-external-dns-domain-c9</a>	<a href="#">lf-tdd-supervisor</a>	<span>✓ Running</span>
○	<a href="#">svc-harbor-domain-c9</a>	<a href="#">lf-tdd-supervisor</a>	<span>✓ Running</span>
○	<a href="#">svc-tkg-domain-c9</a>	<a href="#">lf-tdd-supervisor</a>	<span>✓ Running</span>
○	<a href="#">svc-velero-domain-c9</a>	<a href="#">lf-tdd-supervisor</a>	<span>✓ Running</span>

Click the **Namespaces** tab

Click **NEW NAMESPACE**

# Use the vSphere client to Create a Namespace



Choose your current Supervisor Cluster

Give your new **NAMESPACE** a name, i.e.

lf-tdd-namespace

Click **CREATE**

# Configure New Namespace

The screenshot shows the vSphere Client interface with the title bar "vSphere Client" and a search bar "Search in all environments". The user is logged in as "Administrator@VSPHERE.LOCAL". The left sidebar lists namespaces under "Namespaces", including "If-tdd-namespace" which is selected. The main content area shows a success message: "Your namespace If-tdd-namespace has been successfully created." Below it, instructions for sharing the namespace with a devops team are listed, along with a "GOT IT" button and a checkbox for "Don't show for future workloads". A red box highlights this checkbox. To the right is a cartoon illustration of a character holding balloons. The summary tab displays status information: "Status" (Created 8/10/24), "Config Status" (Running), "Kubernetes Status" (Active), and "Location" (If-tdd-supervisor). The permissions and storage sections indicate no access or policies have been added yet.

Click the **Don't Show for future workloads**

Click **GOT IT**

*This will save you some screen real estate*

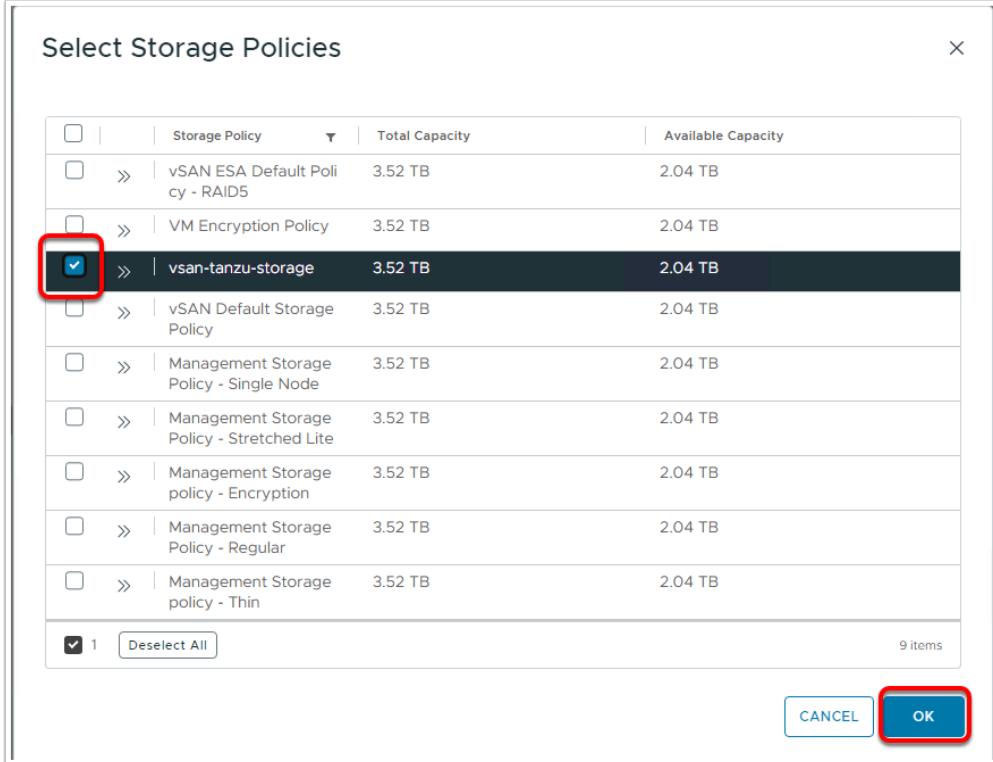
# Configure New Namespace

The screenshot shows the vSphere Client interface for managing namespaces. The left sidebar lists several namespaces, with 'lf-tdd-namespace' selected. The main pane displays the 'lf-tdd-namespace' configuration screen. The top navigation bar includes 'vSphere Client', a search bar, and user information ('Administrator@VSPHERE.LOCAL'). The main content area has tabs for 'Summary', 'Monitor', 'Configure', 'Permissions', 'Compute', 'Storage', 'Network', and 'Resources'. The 'Storage' tab is currently active, showing a message: 'You haven't added any storage policies for this namespace. Add some policies to let your devops team access persistent storage.' A red box highlights the 'ADD STORAGE' button.

Add the tanzu storage policy to this namespace.

Click **ADD STORAGE**

# Configure New Namespace



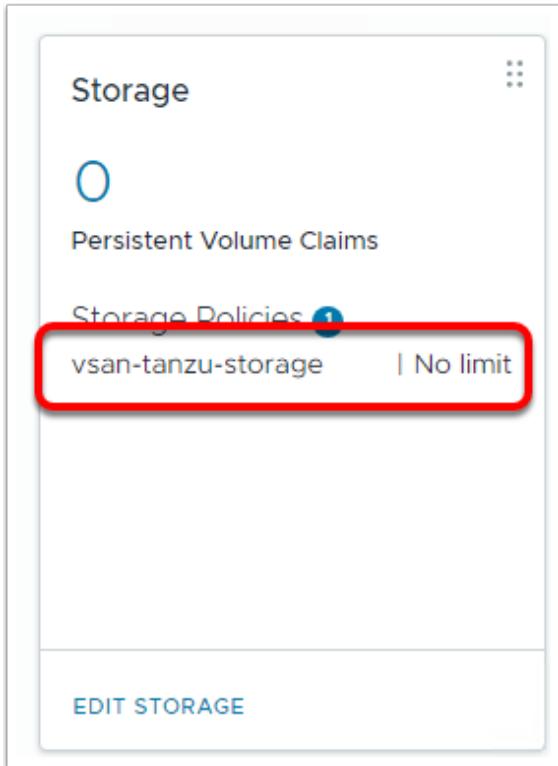
To deploy VMs, vSphere PODs, TKG Clusters, etc to this namespace some settings need to be set..

Start with Storage

Click the selection box for **vsan-tanzu-storage** storage policy

Click **OK**

## Configure New Namespace



Verify the tanzu storage policy is now attached to the namespace

# Configure New Namespace

The screenshot shows the vSphere Client interface for managing namespaces. On the left, a sidebar lists namespaces under 'Namespaces'. A specific namespace, 'lf-tdd-namespace', is selected and shown in detail on the right. The 'Summary' tab is active, displaying information about the namespace's location (linked to 'lf-tdd-supervisor' and 'vccenter-mgmt.vcf.sddc.lab'), its status as 'Active', and its capacity and usage (CPU: 0 MHz used, Memory: 0 MB used, Storage: 0 MB used). Below this, there are sections for 'Tanzu Kubernetes Grid Service' and 'VM Service'. In the 'VM Service' section, a button labeled 'ADD VM CLASS' is highlighted with a red box. At the bottom of the screen, there are tabs for 'Recent Tasks' and 'Alarms', with a note indicating 7 items.

Next you have to give the namespace permission to deploy certain VM Classes.

Click **ADD VM CLASS**

# Configure New Namespace

Add VM Class | If-tdd-namespace X

Add a VM Class for your developers to self-service VMs on this Namespace. VM Classes shown here were created using VM Service.

MANAGE VM CLASSES

	VM Class Name	CPU	CPU Reservation	Memory	Memory Reservation	PCI Devices	Namespaces	VMs
<input type="checkbox"/>	best-effort-2xlarge	8 vCPUs	--	64 GB	--	No	0	0
<input type="checkbox"/>	best-effort-4xlarge	16 vCPUs	--	128 GB	--	No	0	0
<input type="checkbox"/>	best-effort-8xlarge	32 vCPUs	--	128 GB	--	No	0	0
<input type="checkbox"/>	best-effort-large	4 vCPUs	--	16 GB	--	No	0	0
<input checked="" type="checkbox"/>	best-effort-medium	2 vCPUs	--	8 GB	--	No	0	0
<input checked="" type="checkbox"/>	best-effort-small	2 vCPUs	--	4 GB	--	No	0	0
<input type="checkbox"/>	best-effort-xlarge	4 vCPUs	--	32 GB	--	No	0	0
<input type="checkbox"/>	best-effort-xsmall	2 vCPUs	--	2 GB	--	No	0	0
<input type="checkbox"/>	guaranteed-2xlarge	8 vCPUs	100%	64 GB	100%	No	0	0
<input type="checkbox"/>	guaranteed-4xlarge	16 vCPUs	100%	128 GB	100%	No	0	0

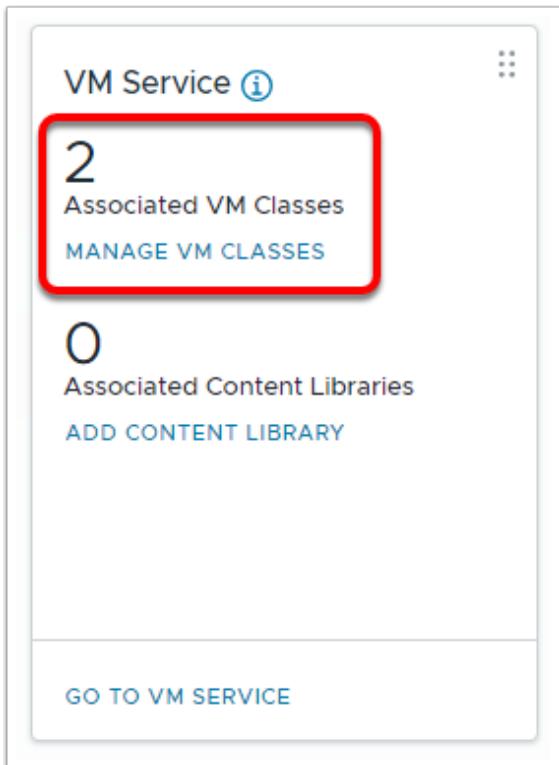
2 Manage Columns Deselect All Items per page: 10 1-10 of 16 items < < 1 / 2 > >

CANCEL OK

For a lab environment like this one we will use **best-effort small** and **medium** VM Classes, and **best-effort** since this is a lab and will not be deploying any large enterprise applications. **Best Effort** since we do not need to worry about resource contention.

Click **OK**

## Configure New Namespace



Verify that you now have two **VM CLASSES** assigned for use to the namespace

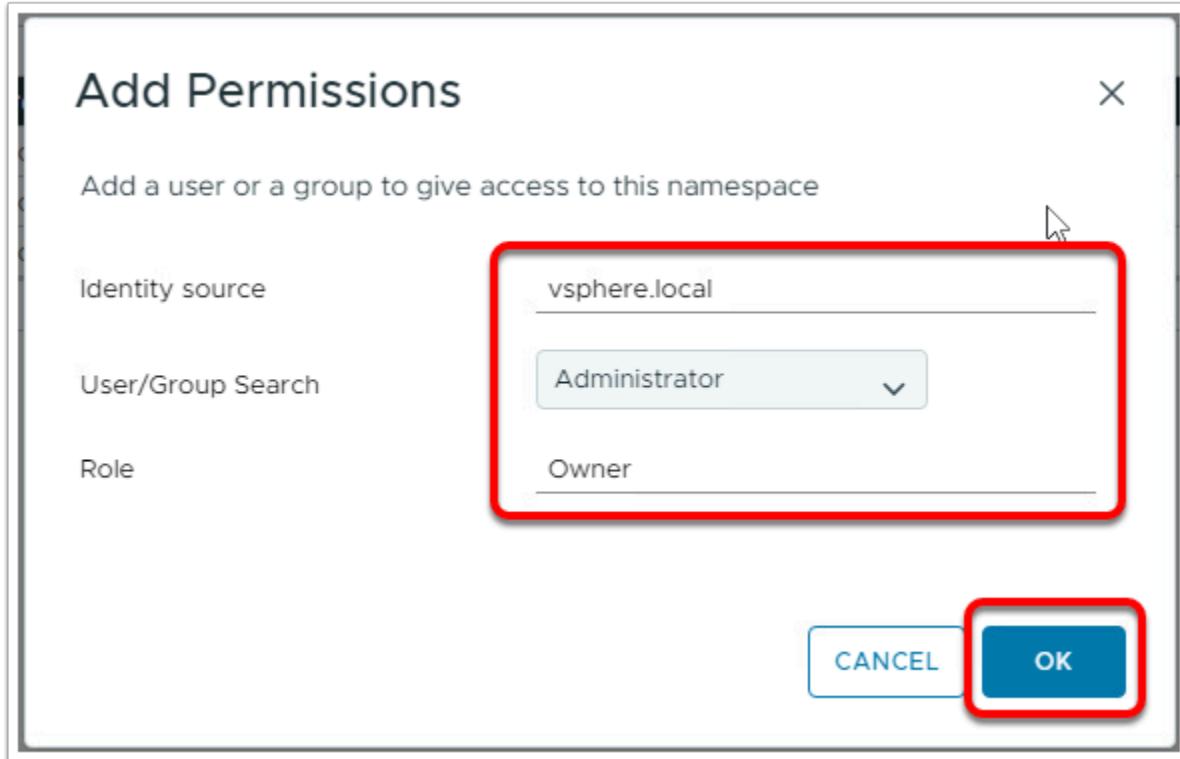
# Configure New Namespace

The screenshot shows the vSphere Client interface for managing namespaces. The left sidebar lists several namespaces, with 'If-tdd-namespace' selected. The main pane displays the details for this namespace, including its status (Running, Active), location (If-tdd-supervisor, vcenter-mgmt.vcf.sddc.lab), and capacity usage (CPU: 0 MHz used, Memory: 0 MB used, Storage: No limit). The 'Permissions' section is highlighted, showing a message: 'You haven't given any devops access to this namespace. Add some permissions to let your devops team directly manage this namespace.' A red box surrounds the 'ADD PERMISSIONS' button. Other sections visible include 'Storage' (Persistent Volume Claims, vsan-tanzu-storage, No limit) and 'VM Service' (2 Associated VM Classes, MANAGE VM CLASSES).

Now you need to give team members permission to use the namespace and the attached services.

Click **ADD PERMISSIONS**

# Configure New Namespace

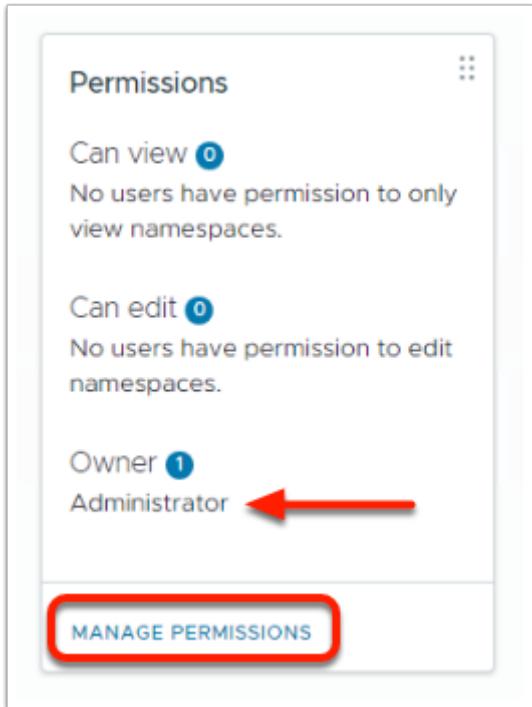


You are going to add groups from **SSO** to the three different Roles for this namespace

- Identity Source: vsphere.local
- User/Group Search: Administrator
- Role: Owner

Click **OK**

# Configure New Namespace



Two more groups/roles

Click **MANAGE PERMISSIONS**

# Configure New Namespace

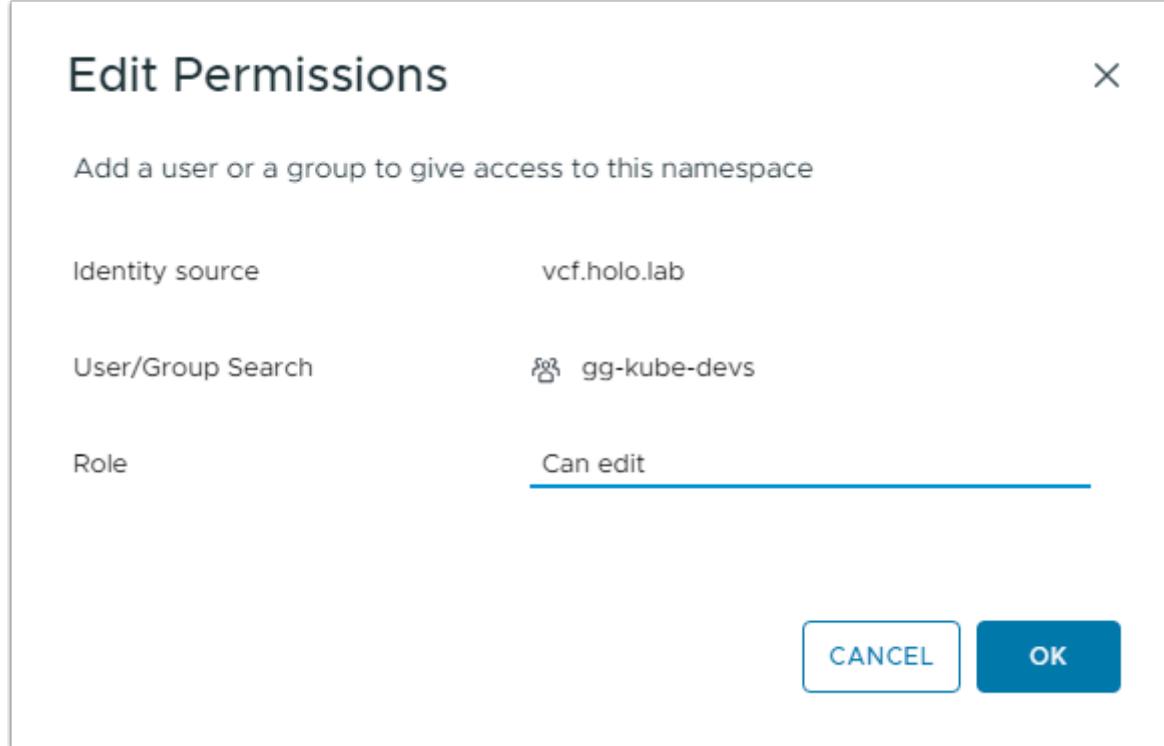
The screenshot shows the 'Permissions' tab for the 'lf-tdd-namespace'. The top navigation bar includes 'Actions', 'Summary', 'Monitor', 'Configure', 'Permissions' (which is selected and underlined), 'Compute', 'Storage', and 'Network'. Below the navigation is a toolbar with 'ADD', 'REMOVE', and 'EDIT' buttons, where 'ADD' is highlighted with a red rectangle. A 'Quick Filter' input field is present. The main area displays a table of permissions:

User/Group	Role	Identity source
Administrator	Owner	vsphere.local

A 'Manage Columns' button is located at the bottom left of the table.

Click **ADD**

## Configure New Namespace



Follow the same workflow and add two more **AD** groups from **SSO** and their roles as follows

- Identity Source: vcf.holo.lab
  - User/Group Search: gg-kube-devs
  - Role: Can Edit
- 
- Identity Source: vcf.holo.lab
  - User/Group Search: gg-kub-readonly
  - Role: Can view

# Configure New Namespace

The screenshot shows the 'Permissions' tab selected in a cloud provider's namespace configuration interface. The top navigation bar includes 'Summary', 'Monitor', 'Configure', 'Permissions' (which is underlined), 'Compute', 'Storage', 'Network', and 'Resources'. Below the navigation is a toolbar with 'ADD', 'REMOVE', and 'EDIT' buttons. A 'Quick Filter' input field is present. The main area displays a table of permissions:

	User/Group	Role	Identity source
<input type="radio"/>	Administrator	Owner	vsphere.local
<input type="radio"/>	gg-kube-devs	Can edit	vcf.holo.lab
<input type="radio"/>	gg-kub-readonly	Can view	vcf.holo.lab

A 'Manage Columns' button is located at the bottom left of the table.

When done your permissions should look like this.

There are a few more Namespace configurations you could address, but for the sake of time and the scope of this lab you can move on.

# Deploy a TKG Workload Cluster

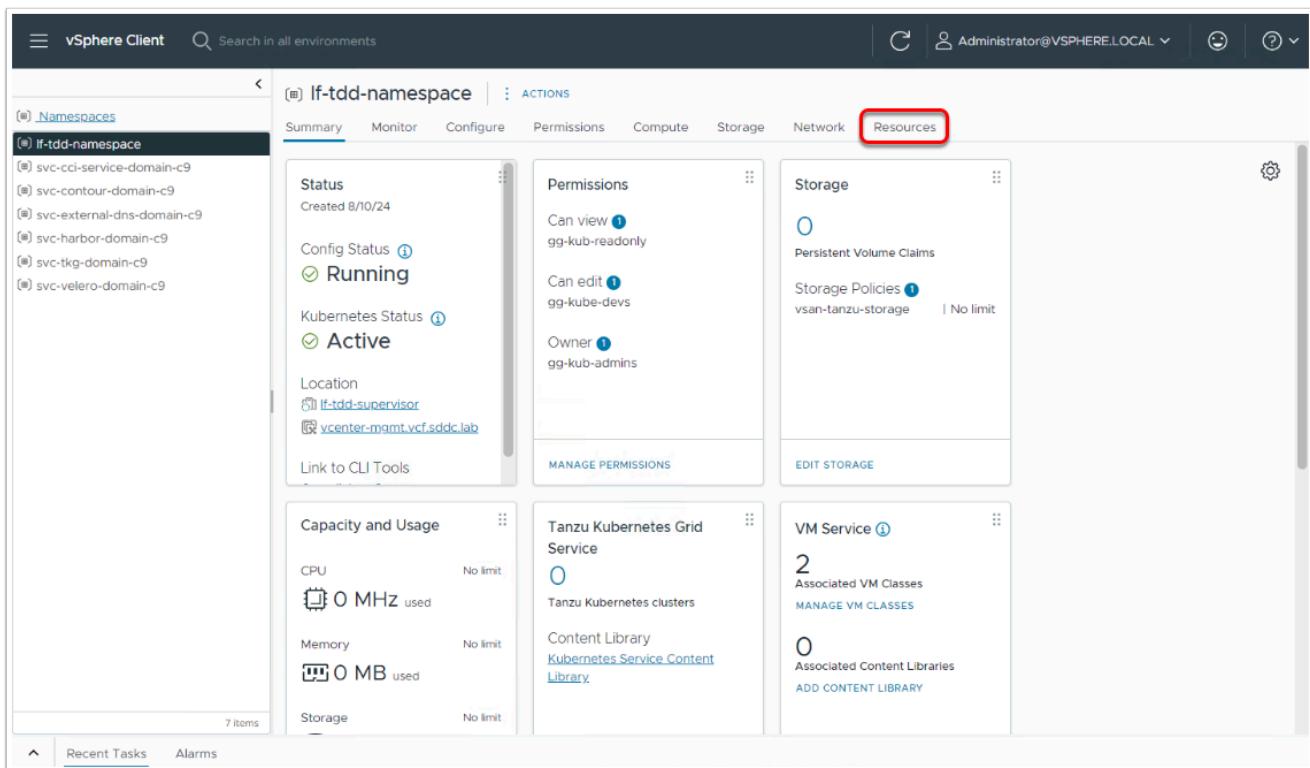
## Task description and objectives

A Tanzu Kubernetes cluster is **a full distribution of the open-source Kubernetes container orchestration platform that is built, signed, and supported by VMware**. You can provision and operate Tanzu Kubernetes clusters on the Supervisor Cluster by using the Tanzu Kubernetes Grid Service.

In this lab you are going to use the previously deployed Supervisor Service the Local Consumption Interface (LCI). The LCI is a requirement of each Supervisor Cluster that will be used as an endpoint by the **VCF Automation** Feature, the **Cloud Consumption Interface**.

VCF Automation is out of scope and focus of this Technical Deep Dive and we only have a single Supervisor Cluster so you will be working in LCI.

## Deploy a TKG Workload Cluster using LCI

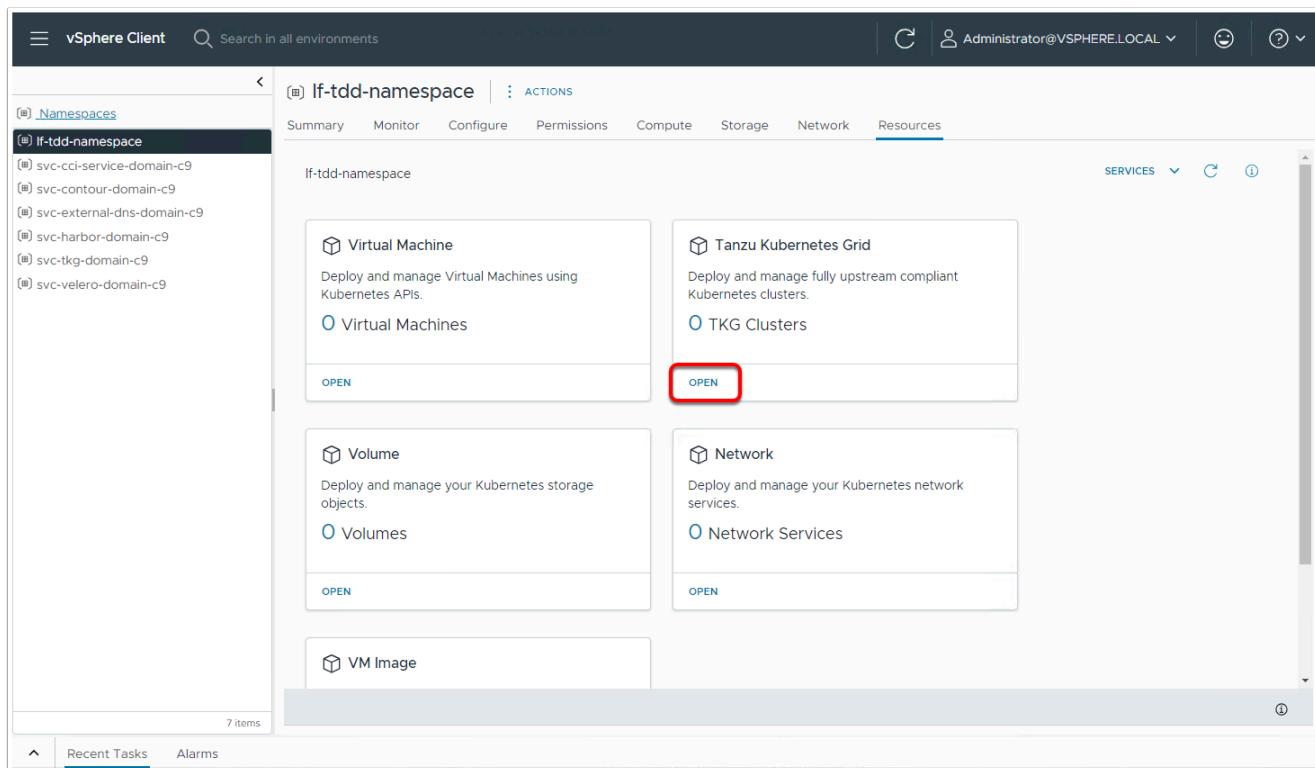


The screenshot shows the vSphere Client interface with the LCI tab selected for the 'If-tdd-namespace' workload cluster. The 'Resources' tab is highlighted with a red box. The interface displays various resource-related metrics and configurations:

- Summary:** Status (Created 8/10/24), Config Status (Running), Kubernetes Status (Active), Location (If-tdd-supervisor, vcenter-mgmt.vcf.sddc.lab).
- Permissions:** Can view gg-kub-readonly, Can edit gg-kube-devs, Owner gg-kub-admins.
- Storage:** Persistent Volume Claims (0), Storage Policies (vsan-tanzu-storage | No limit).
- Capacity and Usage:** CPU (0 MHz used, No limit), Memory (0 MB used, No limit), Storage (No limit).
- Tanzu Kubernetes Grid Service:** Tanzu Kubernetes clusters (0), Content Library (Kubernetes Service Content Library).
- VM Service:** Associated VM Classes (2), Associated Content Libraries (0).

**⚠** If when you click the **Resources** Tab under the **If-tdd-namespace** NAMESPACE, you get no activity, or a form that tells you to install the **Local Consumption Interface** on this Supervisor, Log out and back into the vCenter.

## Deploy a TKG Workload Cluster using LCI



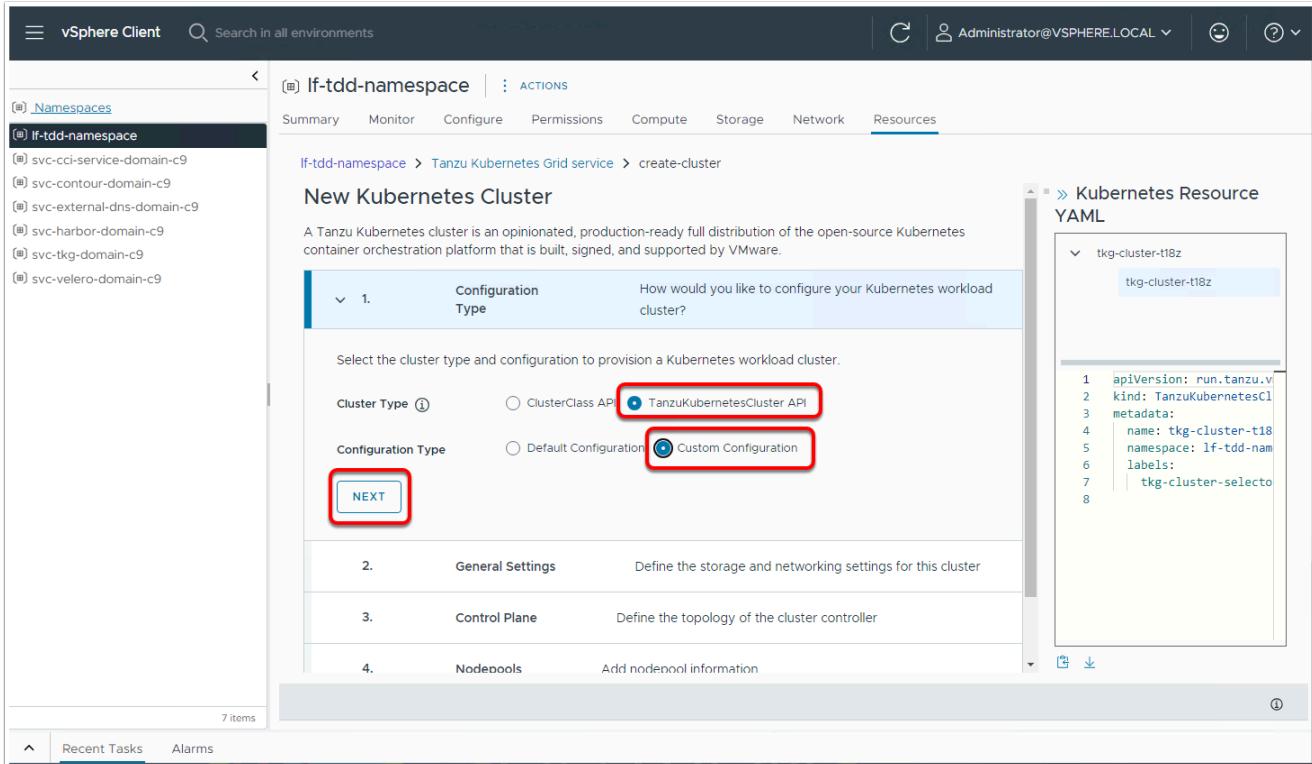
Click **OPEN** on the **Tanzu Kubernetes Grid** Box

# Deploy a TKG Workload Cluster using LCI

The screenshot shows the vSphere Client interface. In the left sidebar, under 'Namespaces', the 'lf-tdd-namespace' is selected. The main content area is titled 'Tanzu Kubernetes Grid service'. It displays a message about Tanzu Kubernetes Grid providing a self-service way to create and manage the lifecycle of Kubernetes clusters. Below this, it says 'Below is the list of Kubernetes clusters that exist on the namespace 'lf-tdd-namespace''. A table is present with one row, which has a red box around the '+ CREATE' button. The table columns are Name, Phase, Tanzu Kubernetes Release, and Age. At the bottom of the table, there are buttons for 'Manage Columns' and 'Clusters per page' set to 25. A filter bar is also visible at the top of the table.

Click **CREATE**

# Deploy a TKG Workload Cluster using LCI



The screenshot shows the 'vSphere Client' interface with the 'Namespaces' section open. A 'New Kubernetes Cluster' wizard is displayed, specifically step 1: 'Configuration Type'. The 'TanzuKubernetesCluster API' radio button is selected and highlighted with a red box. Below it, the 'Custom Configuration' radio button is also highlighted with a red box. A large red box highlights the 'NEXT' button at the bottom of the step. To the right, a 'Kubernetes Resource YAML' panel shows a partial YAML configuration for a cluster named 'tkg-cluster-t18z'.

```
1 apiVersion: run.tanzu.v
2 kind: TanzuKubernetesCl
3 metadata:
4   name: tkg-cluster-t18
5   namespace: lf-tdd-nam
6   labels:
7     tkg-cluster-selecto
```

Select the **TanzuKubernetesCluster API** Radio button

Select the **Custom Configuration** Radio button

Click **NEXT**

# Deploy a TKG Workload Cluster using LCI

The screenshot shows the vSphere Client interface for deploying a TKG cluster. The left sidebar lists namespaces, and the main area shows the 'create-cluster' wizard for 'tkg-cluster-lf001'. The 'General Settings' step is active, with the following details:

- Cluster Name: tkg-cluster-lf001 (1)
- Tanzu Kubernetes Release: v1.29.4---vmware-3-fips-1-tkg.1 (2)
- TKR OSImage Format: Photon (3)
- Persistent Volume Storage: vsan-tanzu-storage (4)

A right-hand panel titled 'Kubernetes Resource YAML' shows the generated YAML code:

```
1  apiVersion: run.tanzu.vmware.com/v1alpha1
2  kind: TanzuKubernetesCluster
3  metadata:
4    name: tkg-cluster-lf001
5    namespace: lf-tdd-namespace
6    labels:
7      tkg-cluster-select
8      annotations:
9        run.tanzu.vmware.com/tkg-cluster: tkg-cluster-lf001
10 
```

1. Give the cluster a name, i.e.

tkg-cluster-lf001

2. Leave Tanzu Kubernetes Release as default

3. Select **Photon** from the **TKR OSImage Format** dropdown

For the **Persistent Storage Classes (optional)** section, select

vsan-tanzu-storage

Click **ADD**

Click **NEXT**

# Deploy a TKG Workload Cluster using LCI

The screenshot shows the Tanzu Management UI for creating a Tanzu Kubernetes Grid service. The current step is "3. Control Plane". The "Replicas" field is set to 1. Under "VM Class", the "best-effort-small" option is selected. The "Storage Class" dropdown is set to "vsan-tanzu-storage". There is an "ADD VOLUME" button for optional volumes. To the right, a "Kubernetes Resource YAML" panel shows the generated YAML for the Tanzu Kubernetes Cluster:

```
1 apiVersion: run.tanzu.v1alpha1
2 kind: TanzuKubernetesCluster
3 metadata:
4   name: tkg-cluster-t18
5   namespace: lf-tdd-namespace
6   labels:
7     tkg-cluster-selectable: true
8   annotations:
9     run.tanzu.vmware.com/spec:
10    settings:
11      storage:
12        classes:
13          - vsan-tanzu-storage
14        defaultClass: vsan
15    topology:
16      controlPlane:
17        replicas: 1
18        vmClass: best-effort
```

Leave the **Control Plane Replica** count at 1

Check the Radio button for **best-effort-small**

# Deploy a TKG Workload Cluster using LCI

The screenshot shows the 'create-cluster' step in the Tanzu Kubernetes Grid service. It displays the following configuration:

- VM Class:** best-effort+ medium (2 vCPUs, 8 GiB)
- Storage Class:** vsan-tanzu-storage
- Volumes (Optional):** No item found

A red box highlights the **NEXT** button at the bottom left of the form.

**Kubernetes Resource YAML** pane (partial view):

```
apiVersion: run.tanzu.v1
kind: TanzuKubernetesCluster
metadata:
  name: tkg-cluster-t18z
  namespace: lf-tdd-namespace
  labels:
    tkg-cluster-selectors: run.tanzu.vmware.com
  annotations:
    run.tanzu.vmware.com/spec.settings.storage.classes: - vsan-tanzu-storage
spec:
  topology:
    controlPlane:
      replicas: 1
      vmClass: best-effort
```

Scroll down and click **NEXT**

# Deploy a TKG Workload Cluster using LCI

A nodepool is a group of worker nodes sharing the same resource allocation and storage.

+ ADD NODEPOOL

Name	Replicas	VM Class	Storage Class

No item found

Manage Columns Items per page 10

NEXT

5. Review and Confirm

Kubernetes Resource YAML

```
1 apiVersion: run.tanzu.vmware.com/v1alpha1
2 kind: TanzuKubernetesCluster
3 metadata:
4   name: tkg-cluster-t18
5   namespace: lf-tdd-namespace
6   labels:
7     tkg-cluster-selectors: "true"
8   annotations:
9     run.tanzu.vmware.com/tkg-cluster: "t18"
10 spec:
11   settings:
12     storage:
13       classes:
14         vsan-tanzu-storage:
15           defaultClass: vsan-tanzu-storage
16   topology:
17     controlPlane:
18       replicas: 1
19       vmClass: best-effort
```

Worker nodes are grouped in NODEPOOLS

Click + ADD NODEPOOL

# Deploy a TKG Workload Cluster using LCI

≡ 1. Configuration

Set the configuration for this nodepool.

Name (i) tkg-cluster-nodepool-fxaz

Replicas (i) 3

VM Class (i) best-effort-small (2 vCPUs and 4Gi RAM) ▾  
2 vCPUs - 0%, 4 GiB - 0%

Storage Class (i) vsan-tanzu-storage ▾

Labels (i) key:value  
use:lf-tdd X ADD

Volumes (i) I would like to configure volumes...

CANCEL NEXT

Set the Replicas count to 3

In Labels enter

use:lf-tdd

Click ADD to add this key:value pair as a label to the **NODEPOOL**

Click **NEXT**

# Deploy a TKG Workload Cluster using LCI

≡ 2. Review and Confirm X

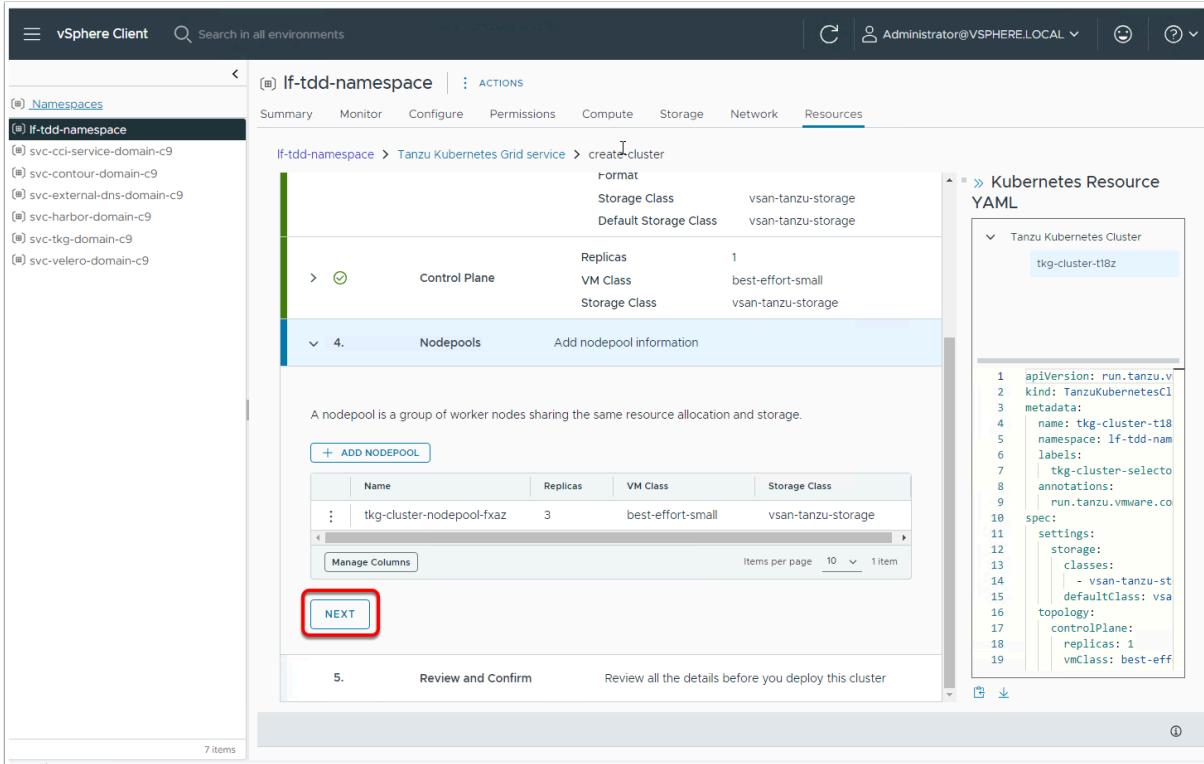
✓ Configuration

Name	tkg-cluster-nodepool-fxaz
Replicas	3
VM Class	best-effort-small
Storage Class	vSAN-tanzu-storage
Labels	use: lf-tdd

[CANCEL](#) [BACK](#) [FINISH](#)

Click **FINISH**

# Deploy a TKG Workload Cluster using LCI



The screenshot shows the vSphere Client interface for deploying a Tanzu Kubernetes Grid service. The left sidebar lists namespaces, and the main pane shows the 'If-tdd-namespace' namespace. A sub-menu 'Tanzu Kubernetes Grid service' is open, and 'create-cluster' is selected. The 'Resources' tab is active, showing cluster configuration details like storage classes and node pool settings. A 'NEXT' button is highlighted with a red box. To the right, a 'Kubernetes Resource YAML' panel displays the generated YAML code for the Tanzu Kubernetes Cluster.

```
1  apiVersion: run.tanzu.v
2  kind: TanzuKubernetesCl
3  metadata:
4    name: tkg-cluster-t18z
5    namespace: lf-tdd-nam
6    labels:
7      | tkg-cluster-selecto
8      annotations:
9        run.tanzu.vmware.co
10       spec:
11         settings:
12           storage:
13             classes:
14               - vsan-tanzu-st
15             defaultClass: vsa
16           topology:
17             controlPlane:
18               replicas: 1
19               vmClass: best-eff
```

Click **NEXT**

# Deploy a TKG Workload Cluster using LCI

The screenshot shows the vSphere Client interface for deploying a TKG Workload Cluster. The left sidebar lists namespaces, and the main pane shows the 'create-cluster' wizard. The configuration details are as follows:

- Configuration Type:** TanzuKubernetesCluster API
- Cluster Type:** Custom Configuration
- General Settings:**
  - Cluster Name: tkg-cluster-t18z
  - Tanzu Kubernetes Release: v1.29.4---vmware.3-fips.1-tkg.1
  - TKR OSImage Format: Photon
  - Storage Class: vsan-tanzu-storage
  - Default Storage Class: vsan-tanzu-storage
- Control Plane:**
  - Replicas: 1
  - VM Class: best-effort-small
  - Storage Class: vsan-tanzu-storage
- Nodepools:** tkg-cluster-nodepool-fxaz

A 'Review and Confirm' step is shown, with a note to review the configuration and the generated YAML file before deploying. The right side shows the generated Kubernetes Resource YAML:

```
1 apiVersion: run.tanzu.vmware.com/v1alpha1
2 kind: TanzuKubernetesCluster
3 metadata:
4   name: tkg-cluster-t18z
5   namespace: lf-tdd-namespace
6   labels:
7     | tkg-cluster-selecto
8   annotations:
9     | run.tanzu.vmware.co
10    spec:
11      settings:
12        storage:
13          classes:
14            | - vsan-tanzu-st
15          defaultClass: vsan-tanzu-storage
16        topology:
17          controlPlane:
18            replicas: 1
19            vmClass: best-effort-small
```

Click **FINISH**

# Deploy a TKG Workload Cluster using LCI

The screenshot shows the vSphere Client interface for the 'If-tdd-namespace' namespace. The 'Summary' tab is selected. In the 'Tanzu Kubernetes Grid Service' section, there is one cluster listed. The 'Compute' tab is highlighted with a red box.

From here you can make sure you are on the correct Namespace

Then Confirm under **Summary** --> **Tanzu Kubernetes Grid** shows 1 new cluster

# Deploy a TKG Workload Cluster using LCI

The screenshot shows the vSphere Client interface for the 'If-tdd-namespace' namespace. The 'Compute' tab is selected. In the 'Tanzu Kubernetes clusters' section, a cluster named 'tkg-cluster-t18z' is listed. The 'Tanzu Kubernetes clusters' button is highlighted with a red box.

Click **Compute**

Click **Tanzu Kubernetes clusters**

you can see your new TKGs cluster being Created

## Deploy a TKG Workload Cluster using LCI

The screenshot shows the vSphere Client interface for the mgmt-cluster-01 cluster. The left sidebar lists various hosts and datastores under the mgmt-datacenter-01. The main pane displays the vSphere DRS section, which includes a large green circular progress bar indicating a Cluster DRS Score of 97%. Below the score, there are four horizontal bars representing VM DRS Scores for different VM ranges: 0-20%, 20-40%, 40-60%, and 60-100%, all showing 0 VMs. The DRS Recommendations and DRS Faults sections are both empty. A "VIEW DRS SETTINGS" and "VIEW ALL VMS" button are at the bottom. To the right, the vSphere HA section shows a "Protected" status with CPU and Memory reserved for failover at 25% each. Proactive HA is disabled, while Host Monitoring is enabled and VM Monitoring is disabled. A "Related Objects" panel on the far right lists the Datacenter (mgmt-datacenter-01), VSphere Zone (domain-c9), Supervisor (lf-tdd-supervisor), and a note about the supervisor being off.

Switch to **Inventory --> Hosts and Clusters** view and drill down into **Namespaces --> your namespace --> Your new tkg cluster**

# Deploy a TKG Workload Cluster using LCI

vSphere Client | mgmt-cluster-01 | ACTIONS

Summary Monitor Configure Permissions Hosts VMs Namespaces Datastores Networks Updates

vSphere DRS

Cluster DRS Score: 96% VM DRS Score: 96%

DRS Recommendations: 0 VMs DRS Faults: 0 VMs

VIEW DRS SETTINGS VIEW ALL VMs

vSphere HA

Protected

CPU: 0% - 50% - 100% Memory: 0% - 50% - 100%

CPU reserved for failover: 25% Memory reserved for failover: 25% Proactive HA: Disabled Host Monitoring: Enabled VM Monitoring: Disabled

Related Objects

- Datacenter: mgmt-datacenter-01
- vSphere Zone: domain-c9
- Supervisor: lf-tdd-supervisor

You can follow the deployment of the **controlplane** and **workload** cluster nodes

vSphere Client | If-tdd-namespace | ACTIONS

Namespaces If-tdd-namespace

Compute Storage Network Resources

Core Kubernetes Tanzu Kubernetes clusters

Name	Creation Time	Phase	Worker Count	Distribution Version	Control Plane
tkg-cluster-t18z	Aug 10, 2024, 7:17:43 PM	Running	3	v1.29.4---vmware.3-fips.1-tkg.1	10.80.0.4

VMware Resources Tanzu Kubernetes clusters Virtual Machines

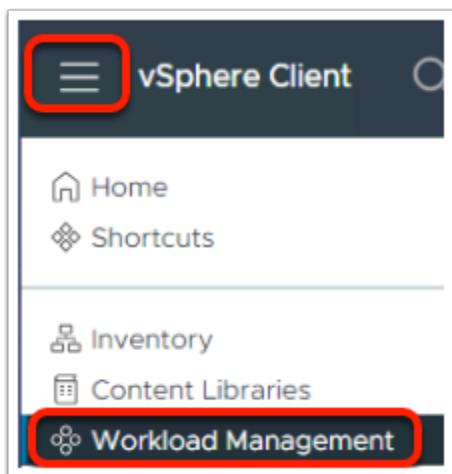
And in a few minutes (30 or so in this lab) you should have a TKGs cluster running

# K8s Tools for IaaS Control Pane Management - CLI

## 💡 Task description and objectives

Now that you have the Supervisor Cluster and a TKG Workload cluster deployed, it's time to deploy the tools needed to connect, manage, and make use of the eco system.

## From the vSphere client



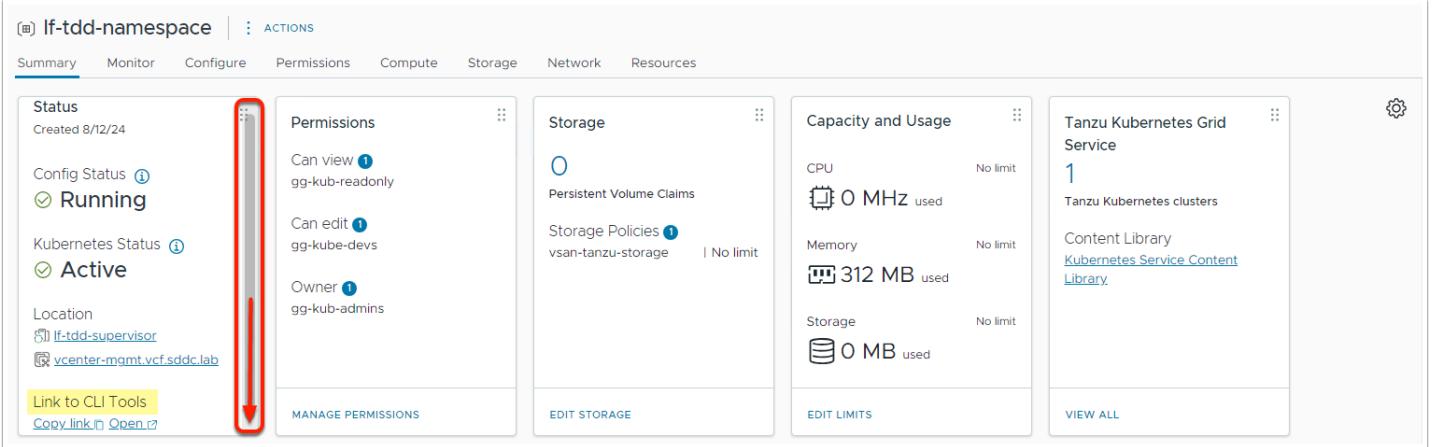
From the vSphere Client home menu, select **Workload Management**.

## Download the K8s Tools for IaaS Management Service

Workload Management									
Namespaces		Supervisors		Services		Updates			
NEW NAMESPACE		REMOVE							
Quick Filter	▼	Enter value							
	Namespaces	↑	Supervisor	Config Status	CPU (Used   Limit)	Memory (Used   Limit)	Storage (Used   Limit)	Description	vCenter
●	(i) If-tdd-namespace		If-tdd-supervisor	Running	0   No Limit	312 MB   No Limit	0   No Limit		vcenter-mgmt.vcf.sddc.lab
○	(i) svc-cci-service-domain-c9		If-tdd-supervisor	Running	209 MHz   No Limit	620 MB   No Limit	0   No Limit		vcenter-mgmt.vcf.sddc.lab
○	(i) svc-contour-domain-c9		If-tdd-supervisor	Running	0   No Limit	326 MB   No Limit	0   No Limit		vcenter-mgmt.vcf.sddc.lab
○	(i) svc-external-dns-domain-c9		If-tdd-supervisor	Running	0   No Limit	53 MB   No Limit	0   No Limit		vcenter-mgmt.vcf.sddc.lab
○	(i) svc-harbor-domain-c9		If-tdd-supervisor	Running	419 MHz   No Limit	1.16 GB   No Limit	18 GB   No Limit		vcenter-mgmt.vcf.sddc.lab
○	(i) svc-tkg-domain-c9		If-tdd-supervisor	Running	0   No Limit	0   No Limit	0   No Limit		vcenter-mgmt.vcf.sddc.lab
○	(i) svc-velero-domain-c9		If-tdd-supervisor	Running	0   No Limit	0   No Limit	0   No Limit		vcenter-mgmt.vcf.sddc.lab

In the Workload Management window click on the **If-tdd-namespace** namespace

# Download the K8s Tools for IaaS Management Service

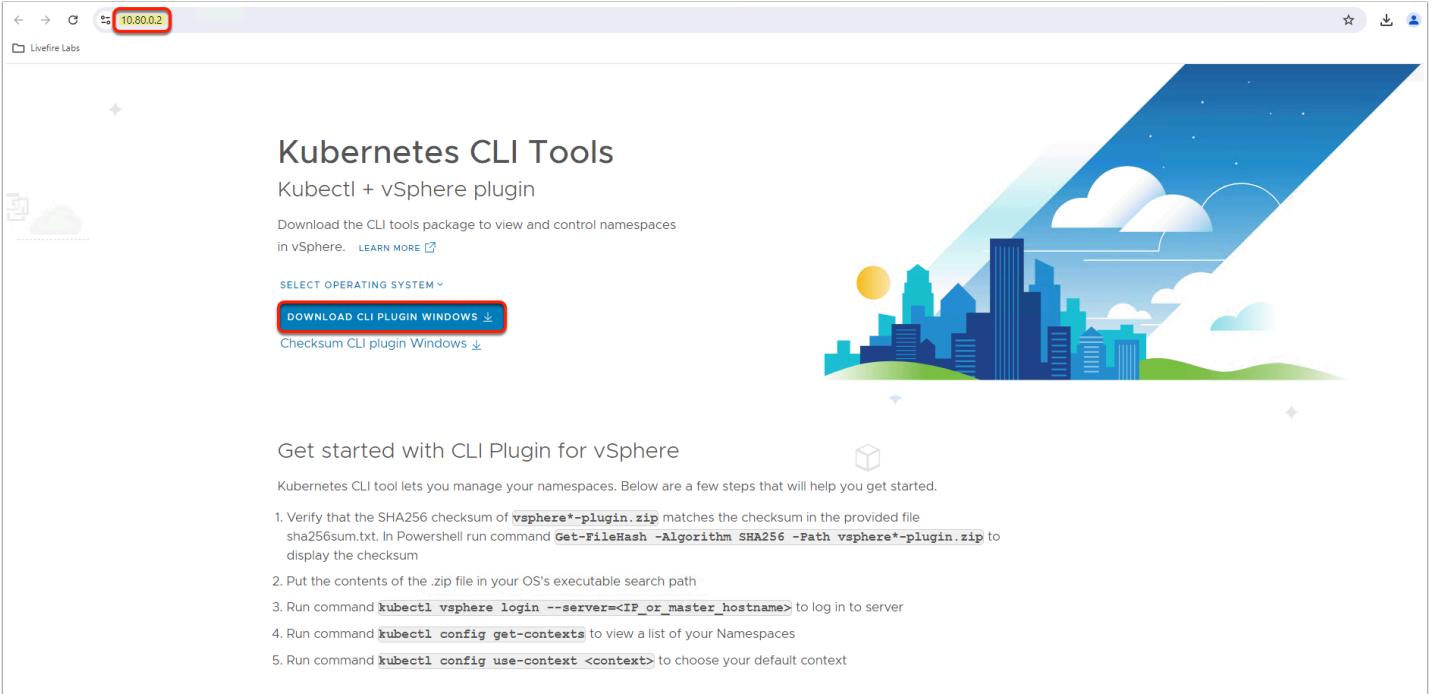


The screenshot shows the 'If-tdd-namespace' management interface. The 'Status' section is highlighted with a red arrow pointing down to the 'Link to CLI Tools' button. The 'Permissions', 'Storage', 'Capacity and Usage', and 'Tanzu Kubernetes Grid Service' sections are also visible.

In the namespace window make sure to scroll down to the bottom in the Status section.

Locate the **Link to CLI Tools** and click on **Open** (opens in a new tab)

# Download the K8s Tools for IaaS Management Service



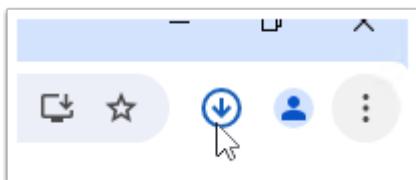
The screenshot shows the 'Kubernetes CLI Tools' download page for vSphere. The 'DOWNLOAD CLI PLUGIN WINDOWS' button is highlighted with a red box. The page includes instructions for getting started with the CLI plugin for vSphere.

*If the Your connection is not private message appears, click Advanced and then click Proceed to 10.80.0.2 (unsafe).*

The browser redirects to the Kubernetes CLI Tools for vSphere download page. The version of your OS is automatically recognized.

## Click on **DOWNLOAD CLI PLUGIN WINDOWS**

**⚠ NOTE:** A new browser tab may open. Or you may not notice any change to the browser at all. Look at the right top corner, for the download icon to show that the download has started.



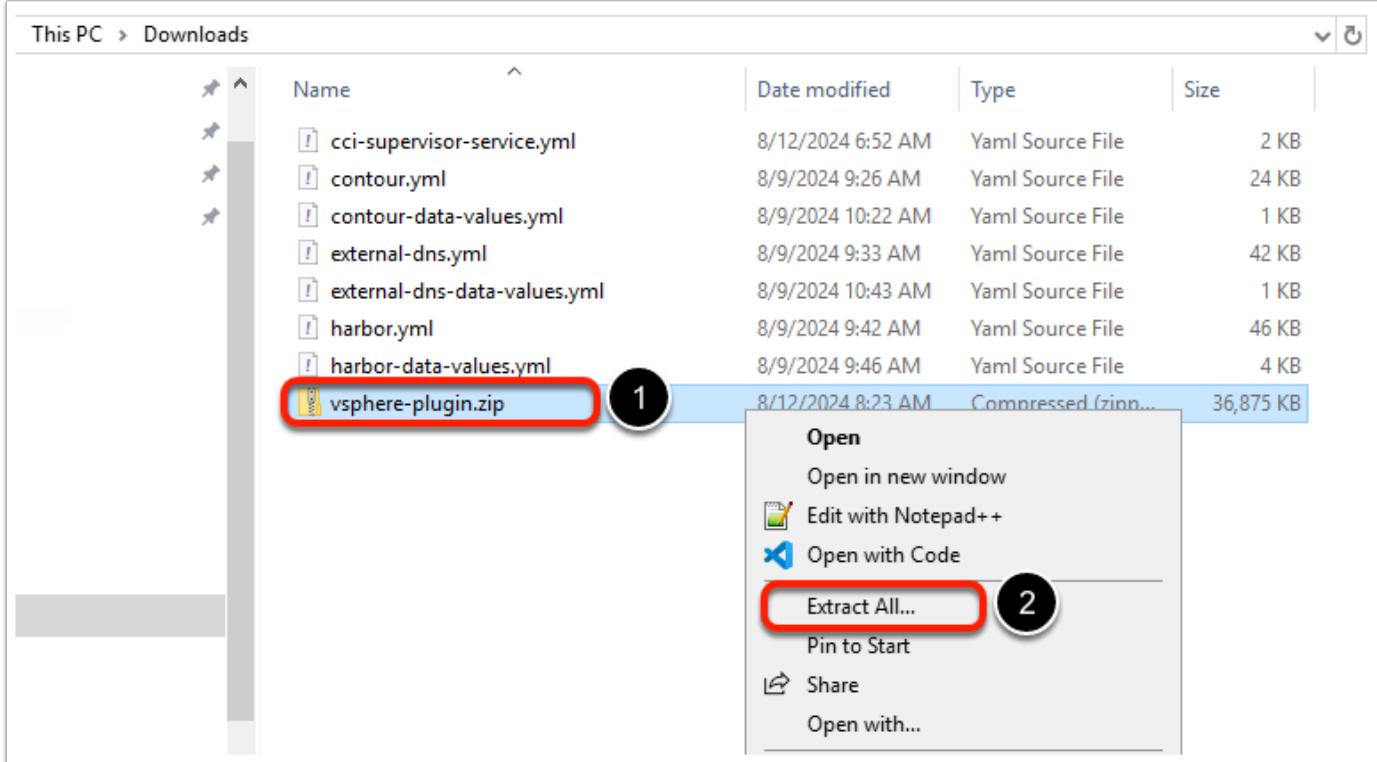
**💡 Please note the **IP address of your Supervisor**. You will use it shortly in the following activities!**

**<SUPERVISOR\_IP> = 10.80.0.2**

This PC > Downloads >				
	Name	Date modified	Type	Size
	cci-supervisor-service.yml	8/12/2024 6:52 AM	Yaml Source File	2 KB
	contour.yml	8/9/2024 9:26 AM	Yaml Source File	24 KB
	contour-data-values.yml	8/9/2024 10:22 AM	Yaml Source File	1 KB
	external-dns.yml	8/9/2024 9:33 AM	Yaml Source File	42 KB
	external-dns-data-values.yml	8/9/2024 10:43 AM	Yaml Source File	1 KB
	harbor.yml	8/9/2024 9:42 AM	Yaml Source File	46 KB
	harbor-data-values.vml	8/9/2024 9:46 AM	Yaml Source File	4 KB
	vsphere-plugin.zip	8/12/2024 8:23 AM	Compressed (zipp...)	36,875 KB

This will download the needed CLI tools in your Downloads folder.

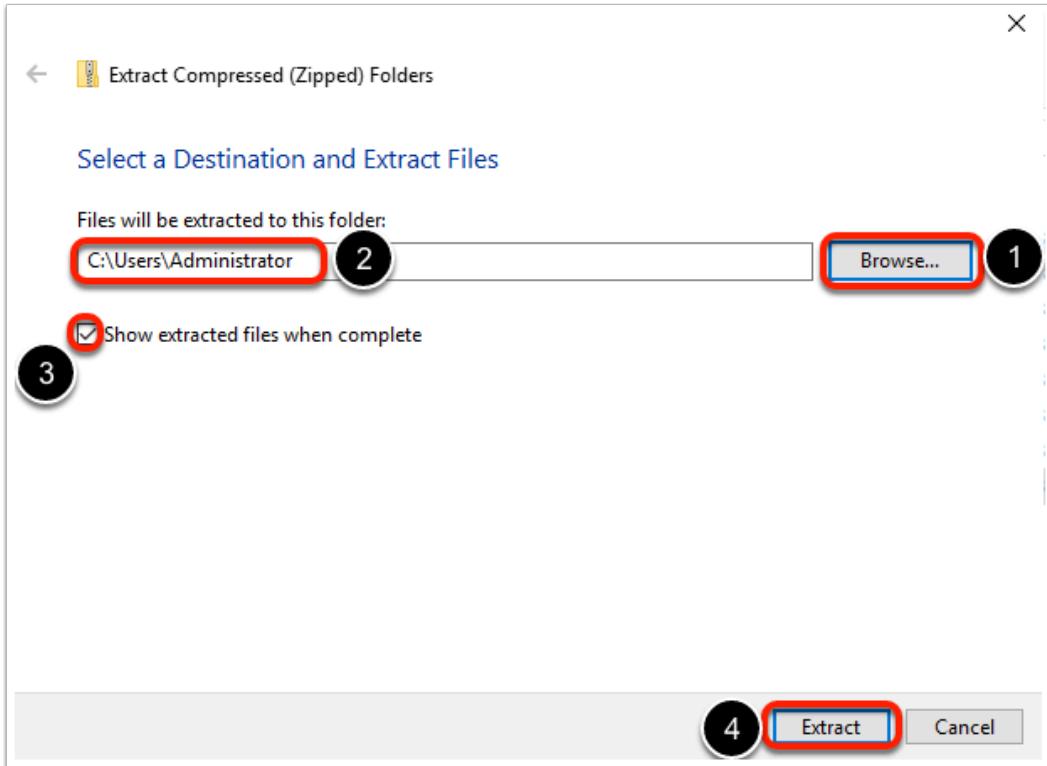
# Install the K8s Tools for IaaS Management Service



In your Downloads folder

1. Right-click on the **vsphere-plugin.zip** file
2. Choose the **Extract All** option

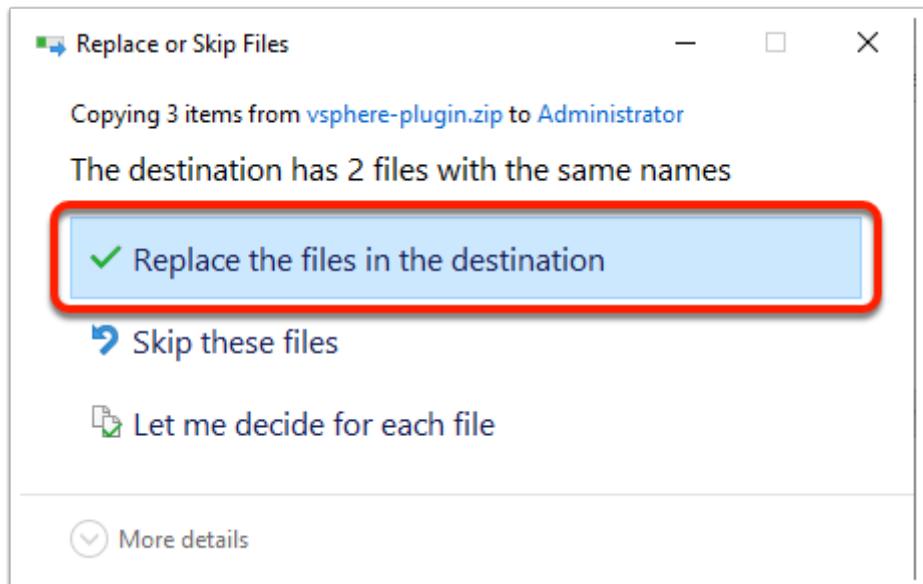
# Install the K8s Tools for IaaS Management Service



In the **Extract Compressed (Zipped) Folders** window

1. Click on Browse, select C:\Users\Administrator folder
2. Mark Show extracted files when complete
3. Click Extract

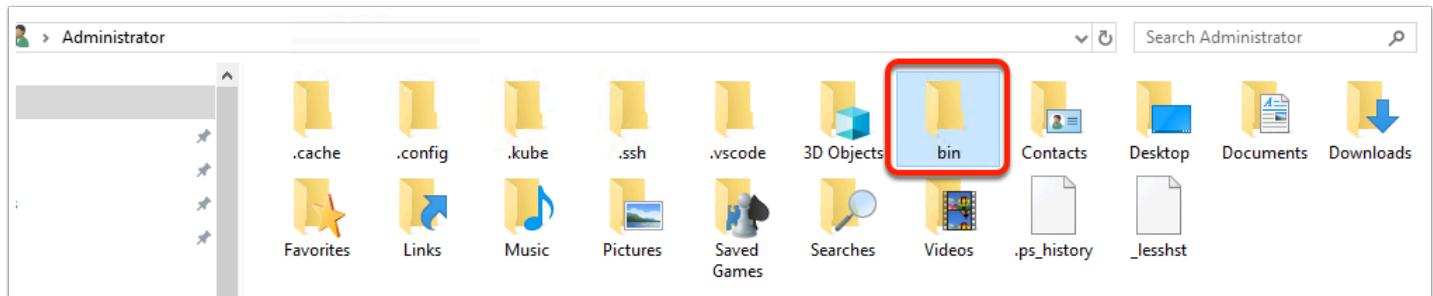
C:\Users\Administrator



Confirm Replace the files in the destination

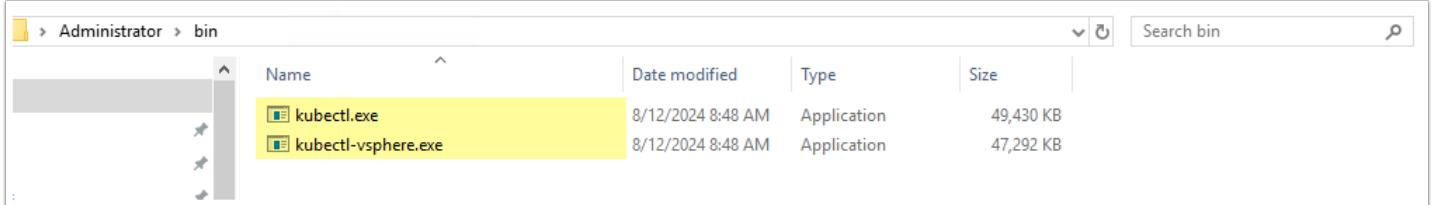
## Install the K8s Tools for IaaS Management Service

 Replacing will guarantee up-to date version of the cli binaries on par with your Supervisor



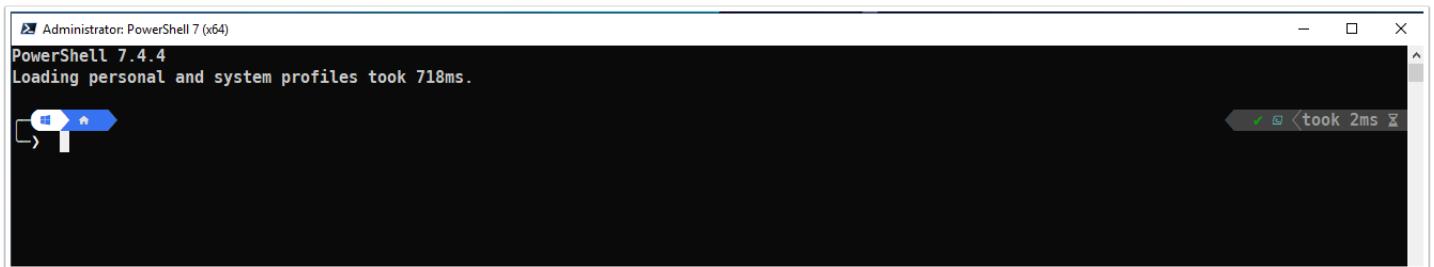
Double click on the **bin** folder

# Install the K8s Tools for IaaS Management Service

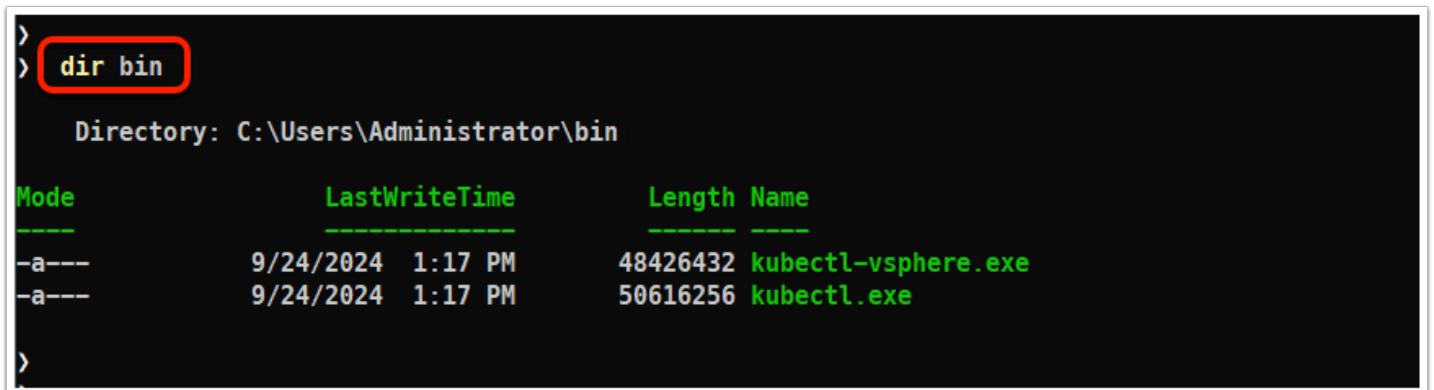


These are your Kubernetes CLI tools for vSphere

# Install the K8s Tools for IaaS Management Service



From your desktop taskbar, Open a new **PowerShell Prompt** and check the binaries



```
dir bin
```

```
> kubectl version --client=true
Client Version: v1.28.3+vmware.wcp.1
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
>
> kubectl vsphere version
kubectl-vsphere: version 0.1.9, build 23754142, change 13167650
>
```

Check the Kubernetes CLI version to ensure that the installation was successful.

```
kubectl version --client=true
```

```
kubectl vsphere version
```

Now you are ready to use the CLI and log in to your Supervisor.

## Install Tanzu CLI & Plugins

To install the Tanzu CLI for use with Tanzu Kubernetes, you install a compatible version of the **Tanzu Core CLI** and the **Tanzu CLI plugins**. Commands provided by these plugins enable cluster and package operations.

A Tanzu CLI plugin is an executable binary that packages a group of CLI commands. The core CLI has some command groups built in, and CLI plugins extend the CLI with additional command groups.

Specific plugins are relevant to different products and to different contexts that you connect the CLI to, based on the context type. Each product designates its relevant plugins as either standalone or context-scoped, as described in [CLI Core](#), [Plugins](#), [Plugin Groups](#), and [Products](#).

**Context-scoped plugins:** When you create a Tanzu CLI context to connect to a product, context-scoped plugins install automatically if they are not already installed. As a backup, if plugins do not automatically install, you can install them by running `tanzu plugin sync` with the CLI set to the desired context.

**Standalone plugins:** For some but not all products, you need to install standalone plugins by running `tanzu plugin install` as described below. To find out whether you need to install standalone plugins for your product, see [CLI Core](#), [Plugins](#), [Plugin Groups](#), and [Products](#).

After completing the steps in [Install the Tanzu CLI](#), follow the instructions to install standalone Tanzu CLI plugins. You can use one command to install all plugins in a plugin group or install each plugin individually.

# Install Tanzu Core CLI

As Tanzu CLI is a multiple solutions oriented CLI, you need to check the [interoperability matrix](#) and choose the correct version to start with:

		VMware Tanzu CLI			
		1.3.0	1.2.0	1.1.0	1.0.0
<b>Tanzu Application Platform</b>					
1.9.0		✓ <sup>0</sup>	✓	✓	—
1.8.0		✓ <sup>0</sup>	✓	✓	—
1.7.0		✓ <sup>0</sup>	✓	✓	—
<b>Tanzu Service Mesh</b>					
v3.2.2		—	—	—	✓
<b>Tanzu Mission Control Self-Managed</b>					
1.3.0		—	✓	—	—
1.2.0		—	—	—	✓
1.1.0		—	—	—	✓
<b>Tanzu Mission Control</b>					
Tanzu Mission Control		—	✓	✓	✓
<b>VMware Tanzu Kubernetes Grid</b>					
2.5.1		—	—	✓	✓
2.5.0		—	—	✓	✓
2.4.1		—	—	✓	✓
2.4.0		—	—	—	✓

Our goal is to become able to deploy and manage Workload clusters with CLI, hence **the correct version matching our requirement is 1.1.0**

# Install Tanzu Core CLI

vmware-tanzu / tanzu-cli Public

Code Issues 33 Pull requests 11 Actions Projects Security Insights

Releases / v1.1.0

v1.1.0

github-actions released this Nov 1, 2023 · 217 commits to main since this release · v1.1.0 · d0679f5

Compare

This release brings you:

- Support for the arm64 architecture for Darwin (Mac)
  - Note: Unlike for Darwin, the Linux CLI binary for arm64 is only available as part of the release to allow testing plugins built for arm64. This build is not yet meant for production and is therefore marked as "unstable".
- UX improvements including dynamic shell completion for core CLI commands arguments and flag values
- Bug fixes and optimizations made on top of the 1.0 release
- Improvements in CLI context management including support for a new variant of kubernetes-based context which integrates with Cloud Services, one-way syncing of kubeconfig on context switch

Get the Tanzu CLI core binary file from GitHub in Chrome open the URL <https://github.com/vmware-tanzu/tanzu-cli/releases/tag/v1.1.0>

## Install Tanzu Core CLI

Contributors  
marchkhouzam, chandrareddy, and 4 other contributors

▼ Assets 17

<a href="#">cltanzu.vmware.com_cliplugins.yaml</a>	4.4 KB	Nov 1, 2023	
<a href="#">tanzu-cli-binaries-checksums.txt</a>	494 Bytes	Nov 1, 2023	
<a href="#">tanzu-cli-binaries-checksums.txt.asc</a>	481 Bytes	Nov 1, 2023	
<a href="#">tanzu-cli-darwin-amd64.tar.gz</a>	38.2 MB	Nov 1, 2023	
<a href="#">tanzu-cli-darwin-arm64.tar.gz</a>	35.6 MB	Nov 1, 2023	
<a href="#">tanzu-cli-linux-amd64.tar.gz</a>	37.3 MB	Nov 1, 2023	
<a href="#">tanzu-cli-linux-arm64-unstable.tar.gz</a>	34.7 MB	Nov 1, 2023	
<a href="#">tanzu-cli-windows-amd64.zip</a>	37.7 MB	Nov 1, 2023	
<a href="#">tanzu-plugins-admin-darwin-amd64.tar.gz</a>	58.8 MB	Nov 1, 2023	
<a href="#">tanzu-plugins-admin-darwin-arm64.tar.gz</a>	54.9 MB	Nov 1, 2023	
<a href="#">tanzu-plugins-admin-linux-amd64.tar.gz</a>	57.8 MB	Nov 1, 2023	
<a href="#">tanzu-plugins-admin-linux-arm64.tar.gz</a>	53.7 MB	Nov 1, 2023	
<a href="#">tanzu-plugins-admin-windows-amd64.zip</a>	58.7 MB	Nov 1, 2023	
<a href="#">tanzu-plugins-checksums.txt</a>	526 Bytes	Nov 1, 2023	
<a href="#">tanzu-plugins-checksums.txt.asc</a>	481 Bytes	Nov 1, 2023	
<a href="#">Source code (zip)</a>		Nov 1, 2023	
<a href="#">Source code (tar.gz)</a>		Nov 1, 2023	

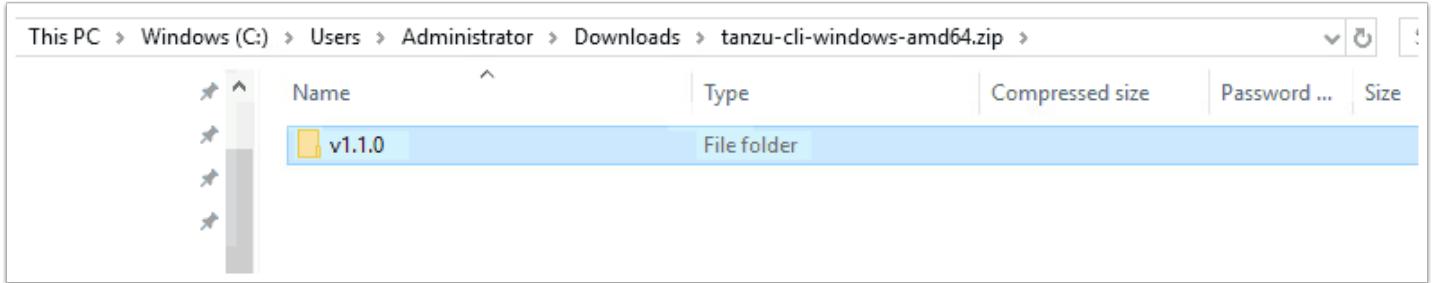
Scroll down to **Assets** and

Click on the the file: **tanzu-cli-windows-amd64.zip** to download the zip.



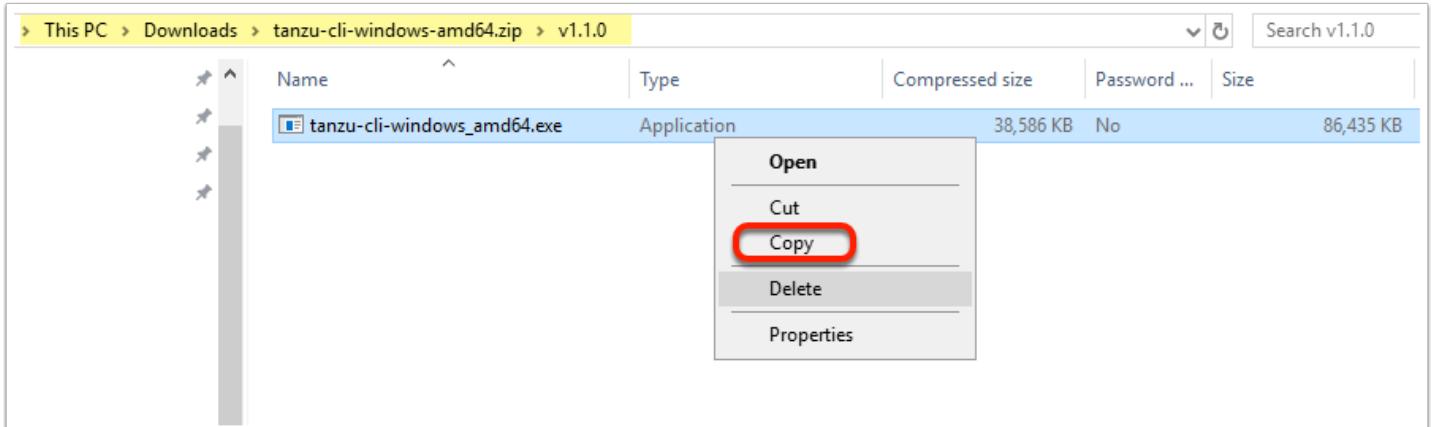
NOTE: A new browser tab may, open. Or you may not notice any change to the browser at all. Look at the right top corner, for the download icon to show the download icon to show the download has started.

# Install Tanzu Core CLI



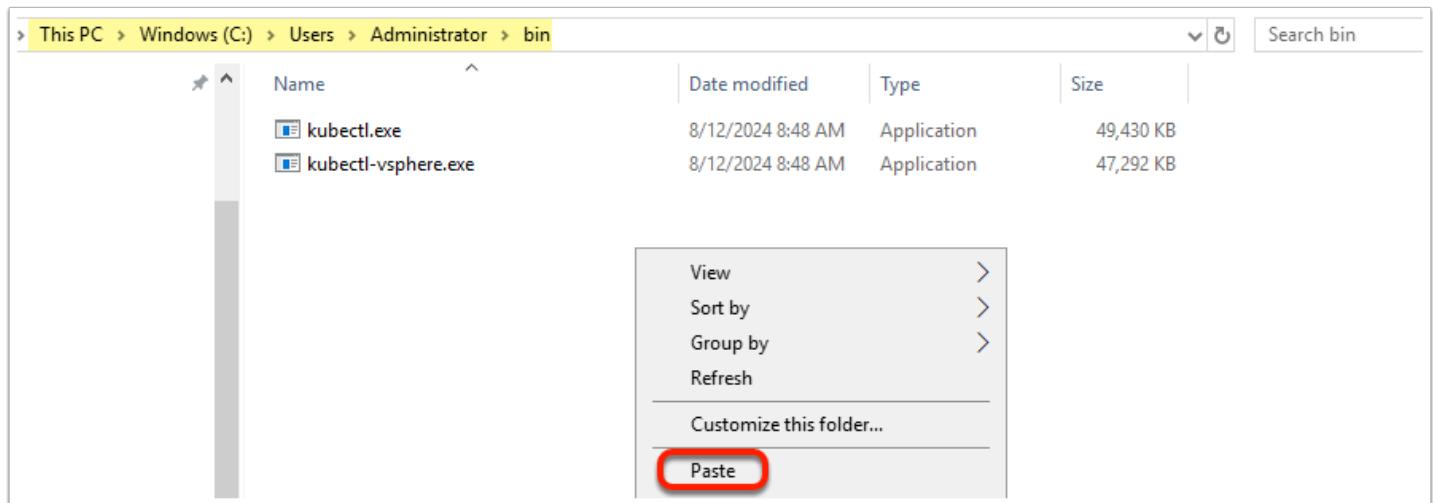
Find the zip archive in your Downloads folder. Double click the the archive, double click into the version subfolder and reach to **tanzu-cli-amd64.exe** binary.

# Install Tanzu Core CLI



Copy the tabzu-cli exe file

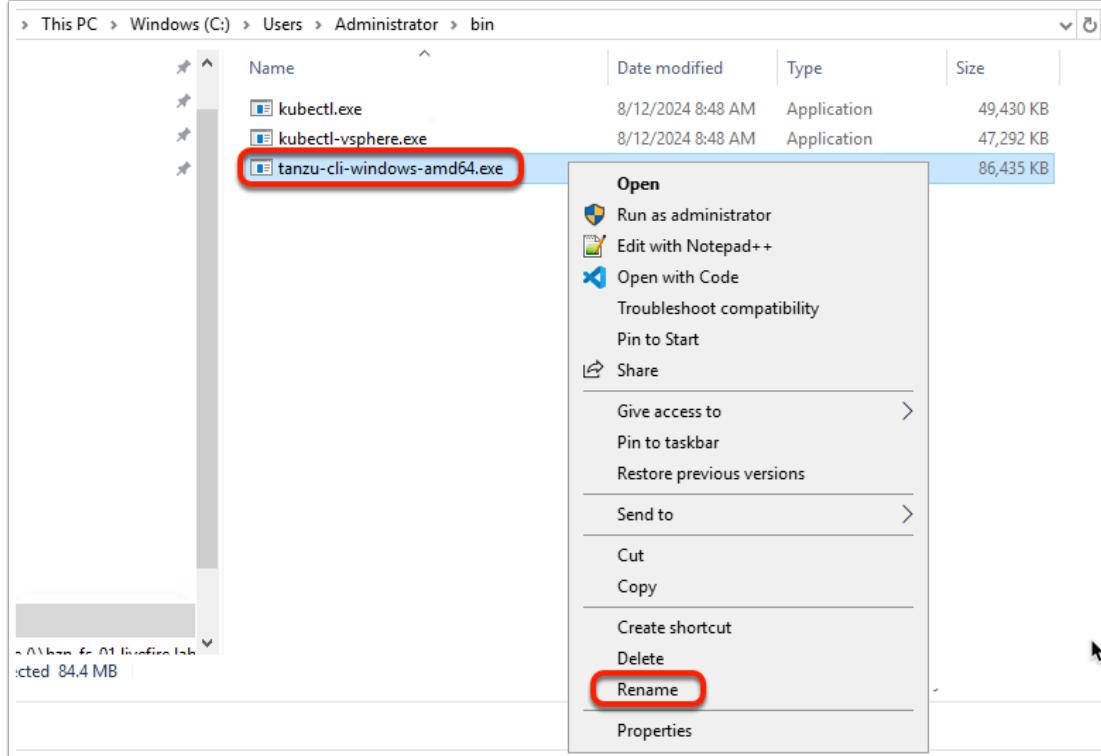
# Install Tanzu Core CLI



Paste it into the **C:\Users\Administrator\bin** folder.

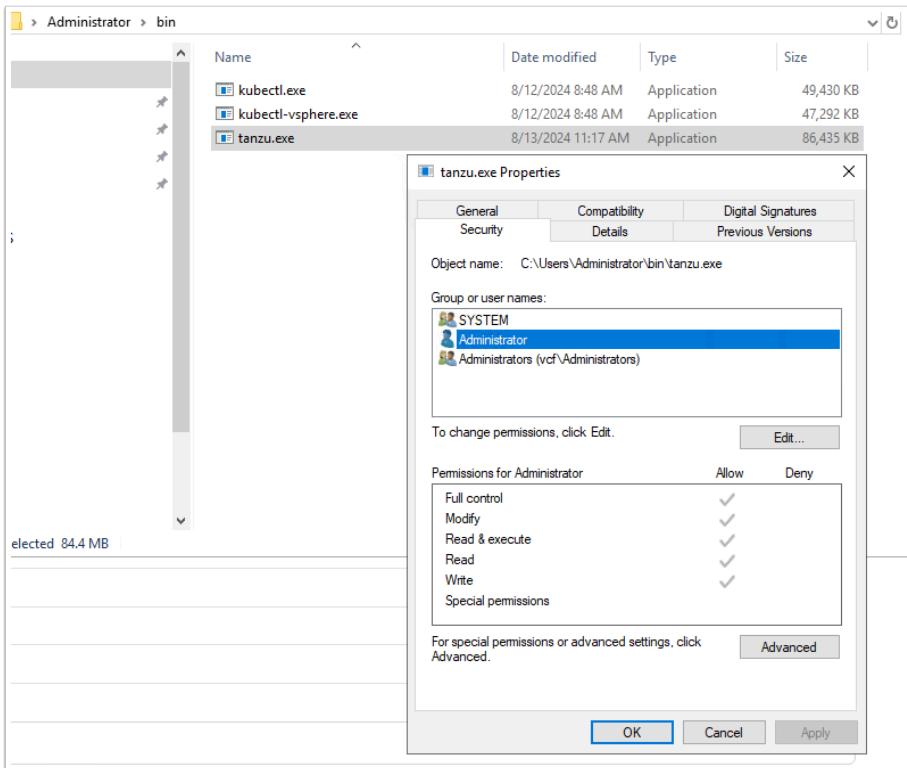
This folder is already in the System Path.

# Install Tanzu Core CLI



Rename the extracted binary to **tanzu.exe**

# Install Tanzu Core CLI



Right-click the **tanzu.exe** file, select **Properties -> Security**, and make sure that your user account has the **Full Control permission**

# Install Tanzu Core CLI

```
>
> tanzu version
version: v1.1.0 ←
buildDate: 2023-11-01
sha: d0679f5a
arch: amd64
>
```

Check that the correct version of the CLI is properly installed

```
tanzu version
```

 This is a manual install we are using, choco, homebrew and other package managers work as well.

## Install the Tanzu CLI plugins

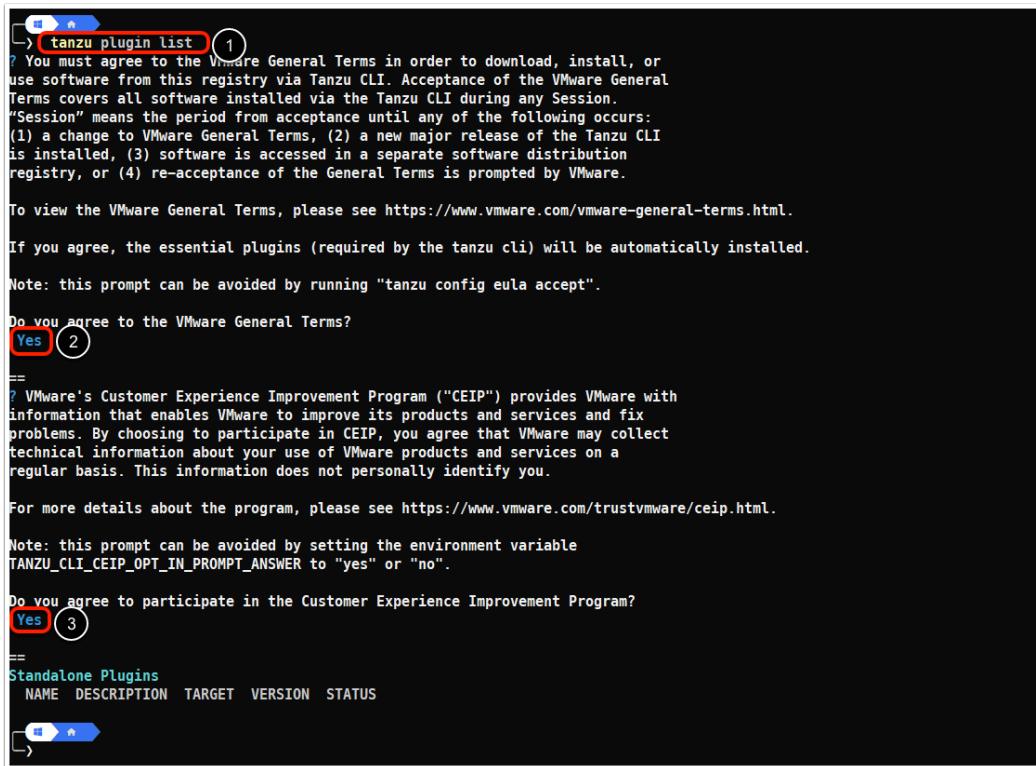
As stated at the beginning of the section, there are two flavors of plugins:

Context-scoped, that are automatically downloaded once you connect the core Tanzu CLI to the supported solution (for example Supervisor in vSphere)

Stand alone, that need to be downloaded additionally. The following section covers these.

The Tanzu CLI Standalone plugins are available through different **plugin groups**.

As a start, you will need to accept a few things:



The screenshot shows a terminal window with the following text:

```
tanzu plugin list 1
? You must agree to the VMware General Terms in order to download, install, or
use software from this registry via Tanzu CLI. Acceptance of the VMware General
Terms covers all software installed via the Tanzu CLI during any Session.
"Session" means the period from acceptance until any of the following occurs:
(1) a change to VMware General Terms, (2) a new major release of the Tanzu CLI
is installed, (3) software is accessed in a separate software distribution
registry, or (4) re-acceptance of the General Terms is prompted by VMware.

To view the VMware General Terms, please see https://www.vmware.com/vmware-general-terms.html.

If you agree, the essential plugins (required by the tanzu cli) will be automatically installed.

Note: this prompt can be avoided by running "tanzu config eula accept".

Do you agree to the VMware General Terms?
Yes 2

==

? VMware's Customer Experience Improvement Program ("CEIP") provides VMware with
information that enables VMware to improve its products and services and fix
problems. By choosing to participate in CEIP, you agree that VMware may collect
technical information about your use of VMware products and services on a
regular basis. This information does not personally identify you.

For more details about the program, please see https://www.vmware.com/trustvmware/ceip.html.

Note: this prompt can be avoided by setting the environment variable
TANZU_CLI_CEIP_OPT_IN_PROMPT_ANSWER to "yes" or "no".

Do you agree to participate in the Customer Experience Improvement Program?
Yes 3

==

Standalone Plugins
NAME DESCRIPTION TARGET VERSION STATUS

→
```

### 1. List the available plugins

```
tanzu plugin list
```

Agree to:

2. General Terms
3. Participate in CEIP

You can find all the plugin groups:

## Install the Tanzu CLI plugins

```
> tanzu plugin group search
[i] Reading plugin inventory for "projects.registry.vmware.com/tanzu_cli/plugins/plugin-inventory:latest", this will take a few seconds. ←
  GROUP          DESCRIPTION           LATEST
  vmware-tanzu/app-developer   Plugins for Application Developer for Tanzu Platform v0.1.7
  vmware-tanzu/platform-engineer Plugins for Platform Engineer for Tanzu Platform v0.1.7
  vmware-tanzu/essentials      Essential plugins for the Tanzu CLI           v1.0.0
  vmware-tap/default          Plugins for TAP                           v1.12.1
  vmware-tkg/default          Plugins for TKG                           v2.5.2
  vmware-tmc/default          Plugins for TMC                           v1.0.0
  vmware-vsphere/default      Plugins for vSphere                         v8.0.3
>
```

To find the list of groups available run

```
tanzu plugin group search
```

These are all the available Plugin groups, containing different **Stand Alone Tanzu CLI plugins**.

By default, on this iteration the **telemetry** plugin from the essentials group is getting installed.

Check it out:

## Install the Tanzu CLI plugins

```
>
> tanzu plugin list
[i] The tanzu cli essential plugins have not been installed and are being installed now. The install may take a few seconds. ←
[i] Installing plugins from plugin group 'vmware-tanzucli/essentials:v1.0.0'
[i] Installing plugin 'telemetry:v1.1.0' with target 'global'

Standalone Plugins
  NAME      DESCRIPTION           TARGET  VERSION  STATUS
  telemetry  configure cluster-wide settings for vmware tanzu telemetry  global  v1.1.0  installed
>
```

Now to install the essential plugin group run:

```
tanzu plugin list
```

This starts a check process for all the plugins available locally.

After a quick message you will see now that the **essential telemetry** plugin is installed

# Install the Tanzu CLI plugins

```
> tanzu plugin group get vmware-tkg/default
Plugins in Group: vmware-tkg/default:v2.5.2 ←
NAME      TARGET  VERSION
isolated-cluster  global  v0.32.3
management-cluster kubernetes  v0.32.3
package      kubernetes  v0.32.1
pinniped-auth   global  v0.32.3
secret       kubernetes  v0.32.0
telemetry    kubernetes  v0.32.3
>
```

You'll need the TKG Plugin shortly, so go ahead and install them. First get the list confirming versions.

Now you can see what is the current available **vmware-tkg** group version.

```
tanzu plugin group get vmware-tkg/default
```

# Install the Tanzu CLI plugins

```
> tanzu plugin install --group vmware-tkg/default
[i] The following plugins will be installed from plugin group 'vmware-tkg/default:v2.5.2' ←
NAME      TARGET  VERSION
isolated-cluster  global  v0.32.3
management-cluster kubernetes  v0.32.3
package      kubernetes  v0.32.1
pinniped-auth   global  v0.32.3
secret       kubernetes  v0.32.0
telemetry    kubernetes  v0.32.3
[i] Installing plugin 'isolated-cluster:v0.32.3' with target 'global'
[i] Installing plugin 'management-cluster:v0.32.3' with target 'kubernetes'
[i] Installing plugin 'package:v0.32.1' with target 'kubernetes'
[i] Installing plugin 'pinniped-auth:v0.32.3' with target 'global'
[i] Installing plugin 'secret:v0.32.0' with target 'kubernetes'
[i] Installing plugin 'telemetry:v0.32.3' with target 'kubernetes'
[ok] successfully installed all plugins from group 'vmware-tkg/default:v2.5.2'
>
>
```

Now run the install

```
tanzu plugin install --group vmware-tkg/default
```

Verify the plugins installed successfully!

# Install the Tanzu CLI plugins

Standalone Plugins				
NAME	DESCRIPTION	TARGET	VERSION	STATUS
isolated-cluster	Prepopulating images/bundle for internet-restricted environments	global	v0.32.3	installed
pinniped-auth	Pinniped authentication operations (usually not directly invoked)	global	v0.32.3	installed
telemetry	configure cluster-wide settings for vmware tanzu telemetry	global	v1.1.0	installed
management-cluster	Kubernetes management cluster operations	kubernetes	v0.32.3	installed
package	tanzu package management	kubernetes	v0.32.1	installed
secret	Tanzu secret management	kubernetes	v0.32.0	installed
telemetry	configure cluster-wide settings for vmware tanzu telemetry	kubernetes	v0.32.3	installed

```
tanzu plugin list
```

 Please, note: These are all **Stand-Alone** type of **Tanzu CLI Plugins!**

After you have installed the Tanzu CLI core and standalone plugins for Tanzu Kubernetes Grid but before you have used it to connect to a management cluster, all context-specific CLI command groups, such as *tanzu cluster* and *tanzu kubernetes-release*, are unavailable and not included in Tanzu CLI --help output.

The Tanzu CLI installs context-scoped plugins automatically when you connect to a management cluster!

# Log in to Supervisor via CLI

## Task description and objectives

As part of your activities, you have to log in to the Supervisor using CLI and the local administrator account.

## Log in to Supervisor using CLI `kubectl`

1. You can continue working in your **PowerShell prompt** and use it as a Terminal window.
2. Create a local **kubeconfig** file by logging in to the Supervisor with the **vSphere plug-in** for kubectl.

```
> kubectl vsphere login --server=10.80.0.2 --insecure-skip-tls-verify
Username: administrator@vsphere.local
Password: [REDACTED]
Logged in successfully.

You have access to the following contexts:
  10.80.0.2
    lf-tdd-namespace
    svc-cci-service-domain-c9
    svc-contour-domain-c9
    svc-external-dns-domain-c9
    svc-harbor-domain-c9
    svc-tkg-domain-c9
    svc-velero-domain-c9

If the context you wish to use is not in this list, you may need to try
logging in again later, or contact your cluster administrator.

To change context, use `kubectl config use-context <workload name>`
```

### Connect to the Tanzu supervisor

```
kubectl vsphere login --server=10.80.0.2 -u administrator@vsphere.local --insecure-skip-tls-verify
```

In this command, and in all commands in later activities in which interaction with the Supervisor is required, you must use the correct Supervisor IP address - **10.80.0.2** - that you recorded in an earlier task.

As the vCenter root certificate is not installed on your windows desktop machine, the **--insecure-skip-tls-verify** flag is required when you log in to the Supervisor

password:

```
VMware123!VMware123!
```

You are creating a local **kubeconfig** file by logging in to the Supervisor with the vSphere plug-in for **kubectl**.

## Log in to Supervisor using CLI kubectl

```
> dir .kube
> dir .kube

  Directory: C:\Users\Administrator\.kube

Mode                 LastWriteTime         Length Name
-->                8/14/2024  1:17 PM          2893 config
>
```

```
dir .kube
```

## Work with contexts in your Supervisor

Once you've logged in your Supervisor, you are informed which contexts you have access to.

But you can check this anytime and get a bit more details:

```
> kubectl config get-contexts
CURRENT  NAME           CLUSTER      AUTHINFO             NAMESPACES
*  10.80.0.2            10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
  lf-tdd-namespace       10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
  svc-cci-service-domain-c9  10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
  svc-contour-domain-c9   10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
  svc-external-dns-domain-c9  10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
  svc-harbor-domain-c9    10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
  svc-tkg-domain-c9      10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
  svc-velero-domain-c9    10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
>
```

```
kubectl config get-contexts
```

The one that is active is marked as current with (\*) in the CLI

You can switch between contexts

## Work with contexts in your Supervisor

```
> kubectl config use-context lf-tdd-namespace
Switched to context "lf-tdd-namespace".
>
> kubectl config get-contexts
CURRENT   NAME          CLUSTER      AUTHINFO           NAMESPACES
*  10.80.0.2    10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local  lf-tdd-namespace
  lf-tdd-namespace  10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local  svc-cci-service-domain-c9
  svc-cci-service-domain-c9  10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local  svc-contour-domain-c9
  svc-contour-domain-c9  10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local  svc-external-dns-domain-c9
  svc-external-dns-domain-c9  10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local  svc-harbor-domain-c9
  svc-harbor-domain-c9  10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local  svc-tkg-domain-c9
  svc-tkg-domain-c9  10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local  svc-velero-domain-c9
  svc-velero-domain-c9  10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local  svc-velero-domain-c9
>
```

```
kubectl config use-context lf-tdd-namespace
```

You can also check the clusters in the current context, if any

## Work with contexts in your Supervisor

```
> kubectl get clusters
NAME          CLUSTERCLASS      PHASE      AGE      VERSION
tkg-cluster-co3d  tanzukubernetescluster  Provisioned  83m    v1.29.4+vmware.3-fips.1
>
> kubectl get nodes -A
NAME          STATUS      ROLES      AGE      VERSION
4200b7c56b369021b669a236712a0913  Ready      control-plane,master  4h38m    v1.28.3+vmware.wcp.1
4200b85def74d6a8daf04a914a4e884  Ready      control-plane,master  4h52m    v1.28.3+vmware.wcp.1
4200e737786185a120ffb8b9d0b4305b  Ready      control-plane,master  4h38m    v1.28.3+vmware.wcp.1
esxi-1.vcf.sddc.lab  Ready      agent      4h30m    v1.28.2-sph-5111a65
esxi-2.vcf.sddc.lab  Ready      agent      4h34m    v1.28.2-sph-5111a65
esxi-3.vcf.sddc.lab  Ready      agent      4h27m    v1.28.2-sph-5111a65
esxi-4.vcf.sddc.lab  Ready      agent      4h27m    v1.28.2-sph-5111a65
>
```

```
kubectl get clusters
```

```
kubectl get nodes -A
```

## Connect the Tanzu CLI to the Supervisor

Make sure your current context points back to the Supervisor

```

> kubectl config use-context 10.80.0.2
Switched to context "10.80.0.2".
> kubectl config get-contexts
CURRENT NAME CLUSTER AUTHINFO NAMESPACE
* 10.80.0.2 10.80.0.2 wcp:10.80.0.2:administrator@vsphere.local lf-tdd-namespace
  lf-tdd-namespace 10.80.0.2 wcp:10.80.0.2:administrator@vsphere.local svc-cci-service-domain-c9
  svc-cci-service-domain-c9 10.80.0.2 wcp:10.80.0.2:administrator@vsphere.local svc-contour-domain-c9
  svc-contour-domain-c9 10.80.0.2 wcp:10.80.0.2:administrator@vsphere.local svc-external-dns-domain-c9
  svc-external-dns-domain-c9 10.80.0.2 wcp:10.80.0.2:administrator@vsphere.local svc-harbor-domain-c9
  svc-harbor-domain-c9 10.80.0.2 wcp:10.80.0.2:administrator@vsphere.local svc-tkg-domain-c9
  svc-tkg-domain-c9 10.80.0.2 wcp:10.80.0.2:administrator@vsphere.local svc-velero-domain-c9
  svc-velero-domain-c9 10.80.0.2 wcp:10.80.0.2:administrator@vsphere.local

```

`kubectl config use-context 10.80.0.2`

`kubectl config get-contexts`

**⚠️ The Tanzu CLI installs **context-scoped plugins** automatically when you connect to a management cluster!**

## Connect the Tanzu CLI to the Supervisor

```

> tanzu context create lf-tdd-supervisor --kubeconfig ./kube/config --kubecontext 10.80.0.2
[ok] successfully created a kubernetes context using the Kubeconfig ./kube/config
[i] Checking for required plugins for context 'lf-tdd-supervisor'...
[i] The following plugins will be installed for context 'lf-tdd-supervisor' of contextType 'kubernetes':
  NAME      TARGET      VERSION
  cluster   kubernetes  v0.33.1
  feature   kubernetes  v0.33.1
  kubernetes-release  kubernetes  v0.33.1
  namespaces  kubernetes  v1.2.0
[i] Installing plugin 'cluster:v0.33.1' with target 'kubernetes'
[i] Installing plugin 'feature:v0.33.1' with target 'kubernetes'
[i] Installing plugin 'kubernetes-release:v0.33.1' with target 'kubernetes'
[i] Installing plugin 'namespaces:v1.2.0' with target 'kubernetes'
[i] Successfully installed all required plugins
>

```

`tanzu context create lf-tdd-supervisor --kubeconfig ./kube/config --kubecontext 10.80.0.2`

Let's check what happened - Tanzu CLI should now have a configured context!

## Connect the Tanzu CLI to the Supervisor

```
> tanzu context list
NAME      ISACTIVE  TYPE      ENDPOINT  KUBECONFIGPATH  KUBECONTEXT
lf-tdd-supervisor  true    kubernetes  ./kube/config  10.80.0.2  ←
```

tanzu context list

From now on, you can refer to your Supervisor in the way you registered it within the context:

## Connect the Tanzu CLI to the Supervisor

```
> tanzu context use
? Select a context lf-tdd-supervisor (./kube/config:10.80.0.2)
[i] Successfully activated context 'lf-tdd-supervisor'
[i] Checking for required plugins for context 'lf-tdd-supervisor'...
[i] All required plugins are already installed and up-to-date
>
```

tanzu context use

Please select your Supervisor **lf-tdd-supervisor** when prompted

Let's inspect the overall plugins in place:

## Connect the Tanzu CLI to the Supervisor

```
> tanzu plugin list
Standalone Plugins
NAME      DESCRIPTION          TARGET  VERSION  STATUS
isolated-cluster  Prepopulating images/bundle for internet-restricted environments  global   v0.32.3  installed
pinniped-auth    Pinniped authentication operations (usually not directly invoked)  global   v0.32.3  installed
telemetry       configure cluster-wide settings for vmware tanzu telemetry     global   v1.1.0   installed
management-cluster Kubernetes management cluster operations                 kubernetes  v0.32.3  installed
package         tanzu package management                                kubernetes  v0.32.1  installed
secret          Tanzu secret management                                kubernetes  v0.32.0  installed
telemetry       configure cluster-wide settings for vmware tanzu telemetry     kubernetes  v0.32.3  installed

Plugins from Context: lf-tdd-supervisor ←
NAME      DESCRIPTION          TARGET  VERSION  STATUS
cluster   Kubernetes cluster operations        kubernetes  v0.33.1  installed
feature    Operate on features and featuregates  kubernetes  v0.33.1  installed
kubernetes-release Kubernetes release operations  kubernetes  v0.33.1  installed
namespaces Discover vSphere Supervisor namespaces you have access to  kubernetes  v1.2.0   installed
>
```

tanzu plugin list

Now you should see the list of:

- The standalone CLI plugins (installed as standalone in the previous task)
- Supervisor specific plugins (just installed as context-scoped) as well as

Be aware, you can always check if newer versions of your installed plugins are available

## Connect the Tanzu CLI to the Supervisor

```
>
> tanzu plugin sync
[i] Plugin sync will be performed for context: 'lf-tdd-supervisor'
[i] Checking for required plugins for context 'lf-tdd-supervisor'...
[i] All required plugins are already installed and up-to-date
[ok] Done
>
```

```
tanzu plugin sync
```

Make a final check if you need to update any of the installed plugins.

You are now ready to create Workload clusters with CLI

# Deploy a TKG Workload Cluster via CLI

## Task description and objectives

With Tanzu Kubernetes Grid on Supervisor, you can provision different styles of workload clusters, including Tanzu Kubernetes clusters with well-defined defaults and Cluster Class clusters with broad definition options.

With Tanzu Kubernetes Grid on Supervisor, you can provision different styles of workload clusters, including Tanzu Kubernetes clusters with well-defined defaults and Cluster Class clusters with broad definition options.

With this lab you will provision and manage the life cycle of TKG clusters on using the Supervisor in the Sphere IaaS control plane via the Tanzu CLI.

TKG Services come with a default ClusterClass definition named `tanzukubernetescluster`. It provides the template for cluster creation using the `v1beta` API.

The `tanzukubernetescluster` ClusterClass is available in all user namespaces. To create a cluster based on this ClusterClass, you need to reference it in the Cluster specification.

**IMPORTANT:** This exercise is being done with administrator@vsphere.local permissions set over the `lf-tdd-namespace`

The default `tanzukubernetescluster` ClusterClass is immutable. It may be updated with each release of the TKG Service.

To view the default `tanzukubernetescluster` ClusterClass that ships with your TKG Service instance, complete the following steps:

## Deploy a TKG Workload Cluster

Aligned with the previous section, Make sure you are logged in your **Supervisor**

Navigate to your working Namespace **lf-tdd-namespace** where you will provision an additional **workload cluster**

```

> kubectl config get-contexts
CURRENT   NAME          CLUSTER      AUTHINFO
*  10.80.0.2    10.80.0.2  wcp:10.80.0.2:administrator@vsphere.local
  lf-tdd-namespace  10.80.0.2  wcp:10.80.0.2:administrator@vsphere.local
  svc-cci-service-domain-c9  10.80.0.2  wcp:10.80.0.2:administrator@vsphere.local
  svc-contour-domain-c9   10.80.0.2  wcp:10.80.0.2:administrator@vsphere.local
  svc-external-dns-domain-c9  10.80.0.2  wcp:10.80.0.2:administrator@vsphere.local
  svc-harbor-domain-c9    10.80.0.2  wcp:10.80.0.2:administrator@vsphere.local
  svc-tkg-domain-c9      10.80.0.2  wcp:10.80.0.2:administrator@vsphere.local
  svc-velero-domain-c9    10.80.0.2  wcp:10.80.0.2:administrator@vsphere.local
> kubectl config use-context lf-tdd-namespace
Switched to context "lf-tdd-namespace".
>

```

kubectl config get-contexts

kubectl config use-context lf-tdd-namespace

Get the default **ClusterClass** and output this to a file named **tkc-dcc.yaml**.

You can **Review** the file.

```

> kubectl get clusterclass tanzukubernetescluster -o yaml > tkc-dcc.yaml
>
> code .\tkc-dcc.yaml
>

```

C: > Users > Administrator > tkc-dcc.yaml > apiVersion

```

io.x-k8s.cluster.v1beta1.ClusterClass (v1beta1@clusterclass.json)
1  apiVersion: cluster.x-k8s.io/v1beta1
2  kind: ClusterClass
3  metadata:
4    annotations:
5      run.tanzu.vmware.com/resolve-tkr: ""
6    creationTimestamp: "2024-08-14T11:21:29Z"
7    generation: 1
8    name: tanzukubernetescluster
9    namespace: lf-tdd-namespace
10   resourceVersion: "152043"
11   uid: 3bb14c04-5230-4add-a34a-04389b9d17ef

```

Note: The screenshot of the **tkc-tdd.yaml** file is truncated, showing only the starting ten rows)

kubectl get clusterclass tanzukubernetescluster -o yaml > tkc-dcc.yaml

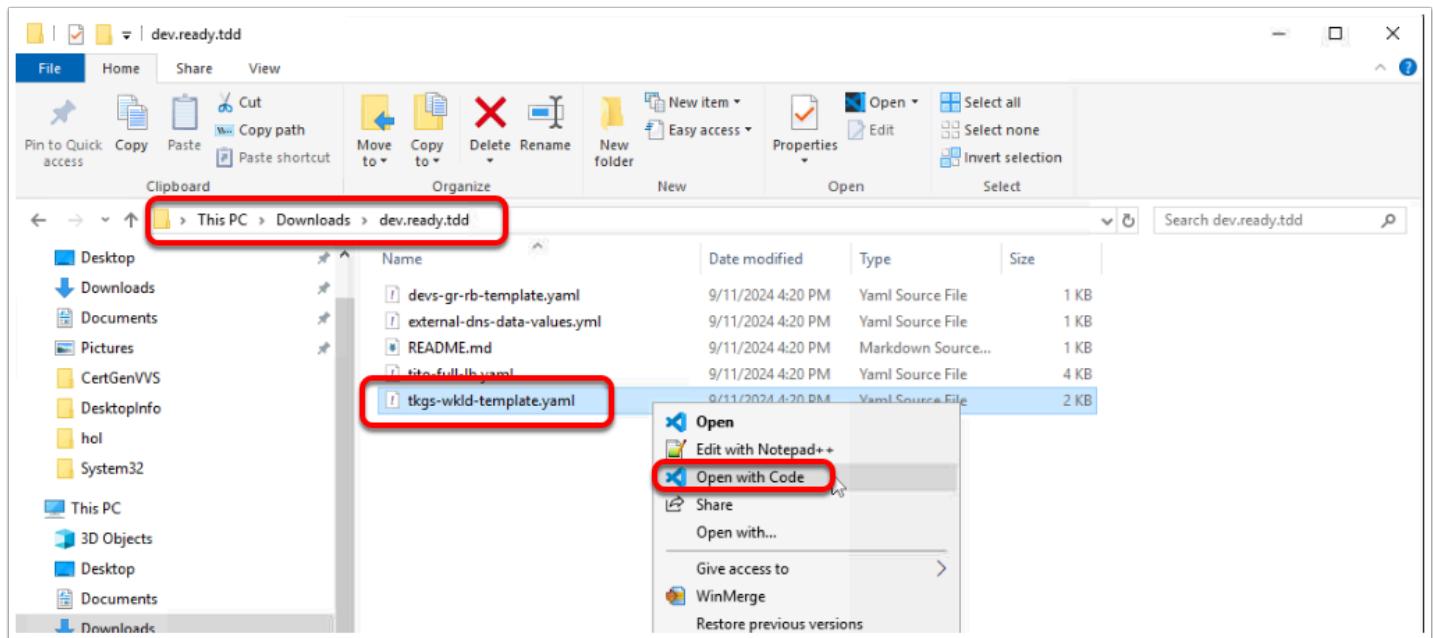
code tkc-dcc.yaml

# TKG Workload Cluster configuration file

You can create a Tanzu Kubernetes Grid **workload cluster configuration file** using the management cluster configuration file as **template**. We have provided a github repository with the files you need. You just need to clone the repo locally if you did not do it earlier. Assuming you did the git clone earlier then ignore this next step

In a terminal run the command

```
git clone https://github.com/Livefire-Labs/dev.ready.tdd.git C:\Users\Administrator\Downloads\dev.ready.tdd
```



Open the file *C:\Users\Administrator\Downloads\dev.ready.tdd\tkgs-wkld-template.yaml* in **VSCode**

This template references a **Custom Cluster Configuration Based on the Default ClusterClass**

```

C:\> Users > Administrator > Downloads > dev.ready.tdd > tkgs-wkld-template.yaml ...
io-x-k8s-cluster:v1beta1.Cluster/v1beta1@cluster.json)
-----
1 2 3 4 5 6 7 8 9
----- Livefire TDD DevReady 2024
You must have enabled your Supervisor in vSphere
in order to provision this TKG workload cluster
by Konstantin Nikolov, Livefire Team
-----
10 apiVersion: cluster.x-k8s.io/v1beta1
11 kind: Cluster
12 metadata:
13   name: CHANGE_IT # tkg-cluster-c001
14   namespace: CHANGE_IT # lf-tdd-namespace
15   labels:
16     | tkg-cluster-selector: CHANGE_IT # tkg-cluster-c001
17 spec:
18   clusterNetwork:
19     pods:
20       cidrBlocks:
21         | - CHANGE_IT # 192.168.156.0/20
22     services:
23       cidrBlocks:
24         | - CHANGE_IT # 10.96.0.0/12
25   serviceDomain: CHANGE_IT # cluster.local
26 topology:
27   class: CHANGE_IT # tanzukubernetescluster
28   version: CHANGE_IT # v1.29.4---vmware.3-fips.1-tkg.1
29 variables:
30   - name: storageClasses
31     value:
32       | - CHANGE_IT # vsan-tanzu-storage
33     - defaultStorageClass
34       value: CHANGE_IT # vsan-tanzu-storage
35     - vmClass
36       value: CHANGE_IT # best-effort-small
37     - storageClass
38       value: CHANGE_IT # vsan-tanzu-storage
39 controlPlane:
40   replicas: CHANGE_IT # 1
41   metadata:
42     annotations:
43       | run.tanzu.vmware.com/resolve-os-image: CHANGE_IT # os-name=ubuntu
44 workers:
45   machineDeployments:
46     - class: node-pool
47       name: CHANGE_IT # tkg-cluster-nodepool-c001
48       replicas: CHANGE_IT # 1
49       metadata:
50         annotations:
51           | run.tanzu.vmware.com/resolve-os-image: CHANGE_IT # os-name=ubuntu
52

```

Use the **SAVE AS** in VSCode and give the template manifest file a custom name: **C:\Downloads\tkgs-wkld-c001.yaml**. Livefire did the work for you, if you look at the tkgs-wkld-template.yaml file, or the copy you saved, we have marked the lines you will need to modify to successfully deploy a TKGs cluster to vSphere. Each line that needs to be changed has the Value "**CHANGE\_IT**" followed by a comment with the value to use for the lab.

Carefully remove all the **CHANGE\_IT #** text on each appropriate line by hand

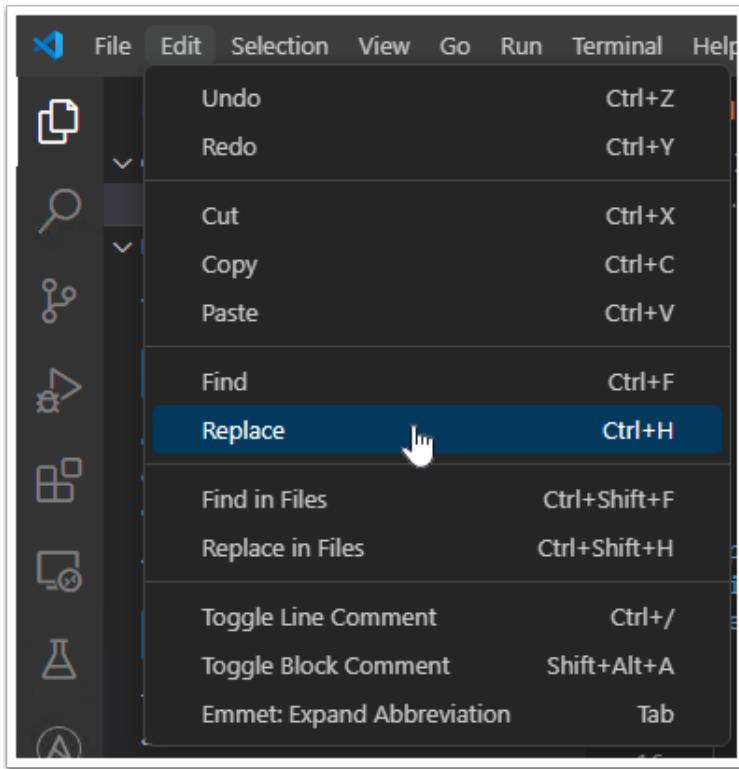
OR

```

10  apiVersion: cluster.x-k8s.io/v1beta1
11  kind: Cluster
12  metadata:
13    name: CHANGE_IT # tkg-cluster-c001
14    namespace: CHANGE_IT # lf-tdd-namespace
15    labels:

```

In VSCode highlight one of the **CHANGE\_IT #** entries. You will notice that VSCode recognizes that there are more text strings that match what you have selected.



With the **CHANGE\_IT #** string still highlighted in VSCode, select **Edit --> Replace**

A screenshot of the VSCode interface showing a 'Replace' dialog box open. The dialog has a search field containing 'CHANGE\_IT #' and a 'Replace' button. Below the dialog, the code editor shows multiple instances of the 'CHANGE\_IT #' placeholder string highlighted in orange. A red box highlights the search field in the dialog, and a red arrow points from the search field to one of the highlighted instances in the code editor.

```

C:\> Users > Administrator > Downloads > dev-ready.tdd > tkgs-wld-template.yaml > {} metadata
  io.x-k8s.cluster.v1beta1.Cluster /v1beta1@cluster.json)
1  # -----
2  #           Livefire TDD DevReady 2024
3  #
4  #       You must have enabled your Supervisor in vSphere
5  #           in order to provision this TKG workload cluster
6  #
7  #           by Konstantin Nikolov, Livefire Team
8  # -----
9
10 apiVersion: cluster.x-k8s.io/v1beta1
11 kind: Cluster
12 metadata:
13   name: CHANGE_IT # tkg-cluster-c001
14   namespace: CHANGE_IT # lf-tdd-namespace
15   labels:
16     | tkg-cluster-selector: CHANGE_IT # tkg-cluster-c001
17 spec:
18   clusterNetwork:
19     pods:
20       cidrBlocks:
21         | - CHANGE_IT # 192.168.156.0/20
22     services:
23       cidrBlocks:
24         | - CHANGE_IT # 10.96.0.0/12
25     serviceDomain: CHANGE_IT # cluster.local
26 topology:
27   class: CHANGE_IT # tanzukubernetescluster
28   version: CHANGE_IT # v1.29.4---vmware.3-fips.1-tkg.1
29   variables:
30     - name: storageClasses
31       value:
32         | - CHANGE_IT # vsan-tanzu-storage
33     - name: defaultStorageClass
34       value: CHANGE_IT # vsan-tanzu-storage
35     - name: vmClass
36       value: CHANGE_IT # best-effort-small
37     - name: storageClass
38       value: CHANGE_IT # vsan-tanzu-storage
39   controlPlane:
40     replicas: CHANGE_IT # 1
41     metadata:
42       annotations:
43         run.tanzu.vmware.com/resolve-os-image: CHANGE_IT # os-name=ubuntu
44   workers:
45     machineDeployments:
46       - class: node-pool
47         name: CHANGE_IT # tkg-cluster-nodepool-c001
48         replicas: CHANGE_IT # 1
49         metadata:
50           annotations:
51             run.tanzu.vmware.com/resolve-os-image: CHANGE_IT # os-name=ubuntu
52

```

VSCode should then highlight ALL of the strings to replace. Leave the bottom value empty

Click the **Replace All** button

```

C:\> Users > Administrator > Downloads > tkgs-wkld-c001.yaml > ...
3  #
4  #       You must have enabled your Supervisor in vSphere
5  #           in order to provision this TKG workload cluster
6  #
7  #           by Konstantin Nikolov, Livefire Team
8  #
9  #
10 apiVersion: cluster.x-k8s.io/v1beta1
11 kind: Cluster
12 metadata:
13   name: tkg-cluster-c001
14   namespace: lf-tdd-namespace
15   labels:
16     tkg-cluster-selector: tkg-cluster-c001
17 spec:
18   clusterNetwork:
19     pods:
20       cidrBlocks:
21         - 192.168.156.0/20
22     services:
23       cidrBlocks:
24         - 10.96.0.0/12
25   serviceDomain: cluster.local
26 topology:
27   class: tanzukubernetescluster
28   version: v1.29.4--vmware.3-fips.1-tkg.1
29 variables:
30   - name: storageClasses
31     value:
32       - vsan-tanzu-storage
33   - name: defaultStorageClass
34     value: vsan-tanzu-storage
35   - name: vmClass
36     value: best-effort-small
37   - name: storageClass
38     value: vsan-tanzu-storage
39 controlPlane:
40   replicas: 1
41   metadata:
42     annotations:
43       run.tanzu.vmware.com/resolve-os-image: os-name=ubuntu
44 workers:
45   machineDeployments:
46     - class: node-pool
47       name: tkg-cluster-nodepool-c001
48       replicas: 1
49       metadata:
50         annotations:
51           run.tanzu.vmware.com/resolve-os-image: os-name=ubuntu
52

```

VSCode should make your configuration file look exactly as the screenshot above. If not you can always undo and try again.

Remember to Save the file!

You are now ready to start provisioning your new workload cluster with CLI.

## Create a TKG Workload Cluster

The Kubernetes [Cluster API](#) is a suite of tools which provide for the declarative provisioning, upgrading, and operating of Kubernetes clusters. [ClusterClass](#) is an evolution of the Cluster API that lets you define templates for managing the life cycle of sets of clusters. TKG Service supports ClusterClass using the [v1beta1](#) API.

ClusterClass API is the newest Cluster API that should be used to deploy all new Kubernetes clusters using built in ClusterClass definitions.

TanzuKubernetesCluster API is provided for backward compatibility and a legacy offering, which will be deprecated in a future release.

Next step is to provision a workload cluster, using the above described ClusterClass, but also leverage the updated ClusterClass API. This is already configured in your configuration file from the previous step.

Let's check you have the correct file in place:

```

> ls .\Downloads\tkgs-wkld-c001.yaml
Directory: C:\Users\Administrator\Downloads
Mode                LastWriteTime         Length Name
----              -----          -----   Name
-a---      9/24/2024  2:06 PM           1767 tkgs-wkld-c001.yaml
>

```

```
ls .\Downloads\tkgs-wkld-c001.yaml
```

A good idea is to perform an initial dry run before the actual cluster creation, which queries vCenter to ensure that the vSphere resources exist.

## Create a TKG Workload Cluster

```

> tanzu cluster create -f .\Downloads\tkgs-wkld-c001.yaml --dry-run
Downloading the TKG Bill of Materials (BOM) file from 'projects.registry.vmware.com/tkg/tkg-bom:v2.5.0'
Downloading the TKG Bill of Materials (BOM) file from 'projects.registry.vmware.com/tkg/tkr-bom:v1.28.4_vmware.1-tkg.1'
the old providers folder C:\Users\Administrator\.config\tanzu\tkg\providers is backed up to C:\Users\Administrator\.config\tanzu\tkg\providers-20240924141049-yv2x3ca0

```

```

# ====== Livefire TDD DevReady 2024 ======
#
# You must have enabled your Supervisor in vSphere
# in order to provision this TKG workload cluster
#
# by Konstantin Nikolov, Livefire Team
# ======

apiVersion: cluster.x-k8s.io/v1beta1
kind: Cluster
metadata:
  name: tkg-cluster-c001
  namespace: lf-tdd-namespace
  labels:
    tkg-cluster-selector: tkg-cluster-c001
spec:
  clusterNetwork:
    pods:
      cidrBlocks:
        - 192.168.156.0/20
    services:
      cidrBlocks:
        - 10.96.0.0/12
    serviceDomain: cluster.local
  topology:
    class: tanzukubernetescluster
    version: v1.29.4---vmware.3-ips.1-tkg.1
  variables:
    - name: storageClasses
      value:
        - vsan-tanzu-storage
    - name: defaultStorageClass
      value: vsan-tanzu-storage
    - name: vmclass
      value: best-effort-small
    - name: storageClass
      value: vsan-tanzu-storage
  controlPlane:
    replicas: 1
    metadata:
      annotations:
        run.tanzu.vmware.com/resolve-os-image: os-name=ubuntu
  workers:
    machineDeployments:
      - class: node-pool
        name: tkg-cluster-nodepool-c001
        replicas: 1
        metadata:
          annotations:
            run.tanzu.vmware.com/resolve-os-image: os-name=ubuntu

```

Run the command

```
tanzu cluster create -f .\Downloads\tkgs-wkld-c001.yaml --dry-run
```

If the output shows **exit status 1**, you must review the configuration parameters in the **tkgs-wkld-c001.yaml** configuration file.

If dry run finished okay, go ahead and create the cluster:

## Create a TKG Workload Cluster

```
> tanzu cluster create -f .\Downloads\tkgs-wkld-c001.yaml
You are trying to create a cluster with kubernetes version '' on vSphere with Tanzu, Please make sure virtual machine image for the same is available in the cluster content library.
Do you want to continue? [y/N] y
Validating configuration...
waiting for cluster to be initialized...
[zero or multiple KCP objects found for the given cluster, 0 tkg-cluster-c001 lf-tdd-namespace, no MachineDeployment objects found for the given cluster]
[cluster control plane is still being initialized: ScalingUp, cluster infrastructure is still being provisioned: ScalingUp]
cluster control plane is still being initialized: ScalingUp
cluster control plane is still being initialized: WaitingForKubeadmInit
waiting for cluster nodes to be available...
unable to get the autoscaler deployment, maybe it is not exist
waiting for addons core packages installation...

Workload cluster 'tkg-cluster-c001' created
>
```

Run the same command as before, but without the --dry-run

```
tanzu cluster create -f .\Downloads\tkgs-wkld-c001.yaml
```

When prompted type **y** to continue!

The Tanzu Kubernetes Grid cluster takes **approximately 25 minutes** to deploy.

Wait for the **Workload cluster 'tkg-cluster-c001' created** message to appear.

While your new workload cluster is being provisioned, you can track some visuals in the vSphere client UI

In your Workload Management --> Namespaces --> If-tdd-namespace pane, you will see the new cluster changing phases:

# Create a TKG Workload Cluster

The screenshot shows the Tanzu Kubernetes Grid service interface for the namespace 'If-tdd-namespace'. The 'Resources' tab is selected. A red box highlights the 'If-tdd-namespace' namespace name at the top left. Another red box highlights the 'Resources' tab at the top right. A red arrow points from the 'Creating' status of the first cluster row to the 'Creating' text in the Phase column of the same row.

Name	Phase	Tanzu Kubernetes Release	Age
<a href="#">tkg-cluster-c001</a>	... Creating	v1.29.4---vmware.3-fips.1-tkg.1	1 minute
<a href="#">tkg-cluster-c03d</a>	Running	v1.29.4---vmware.3-fips.1-tkg.1	21 hours

Manage Columns Clusters per page 25 1 - 2 of 2 Clusters

- from ... Creating

# Create a TKG Workload Cluster

If-tdd-namespace | ACTIONS

Summary Monitor Configure Permissions Compute Storage Network Resources

If-tdd-namespace > Tanzu Kubernetes Grid service

Tanzu Kubernetes Grid service

Tanzu Kubernetes Grid provides a self-service way to create and manage the lifecycle of Kubernetes clusters.

Below is the list of Kubernetes clusters that exist on the namespace 'If-tdd-namespace'.

+ CREATE Filter

Name	Phase	Tanzu Kubernetes Release	Age
<a href="#">tkg-cluster-c001</a>	Running	v1.29.4---vmware.3-fips.1-tkg.1	24 minutes
<a href="#">tkg-cluster-c03d</a>	Running	v1.29.4---vmware.3-fips.1-tkg.1	21 hours

Manage Columns Clusters per page: 25 1 - 2 of 2 Clusters

... to Running within 25 mins or so.

# Create a TKG Workload Cluster

The screenshot shows the Tanzu Kubernetes Grid service interface for the namespace 'lf-tdd-namespace'. The 'Resources' tab is selected. A cluster named 'tkg-cluster-c001' is listed, with options to 'VIEW YAML' or 'DOWNLOAD KUBECONFIG FILE'. Below the cluster name, there is a section titled 'Cluster Details' containing the following information:

Phase	Running
Cluster Class	tanzukubernetescluster
Tanzu Kubernetes Release	v1.29.4---vmware.3-fips.1-tkg.1

Below 'Cluster Details' is a section titled 'Persistent Volume Storage' with the following settings:

Allowed Storage Classes	vsan-tanzu-storage
Default Storage Class	vsan-tanzu-storage

Finally, there is a section titled 'Networking' with the following configuration:

Pods CIDR	192.168.156.0/20
Services CIDR	10.96.0.0/12
Service Domain	cluster.local
CNI	Antrea

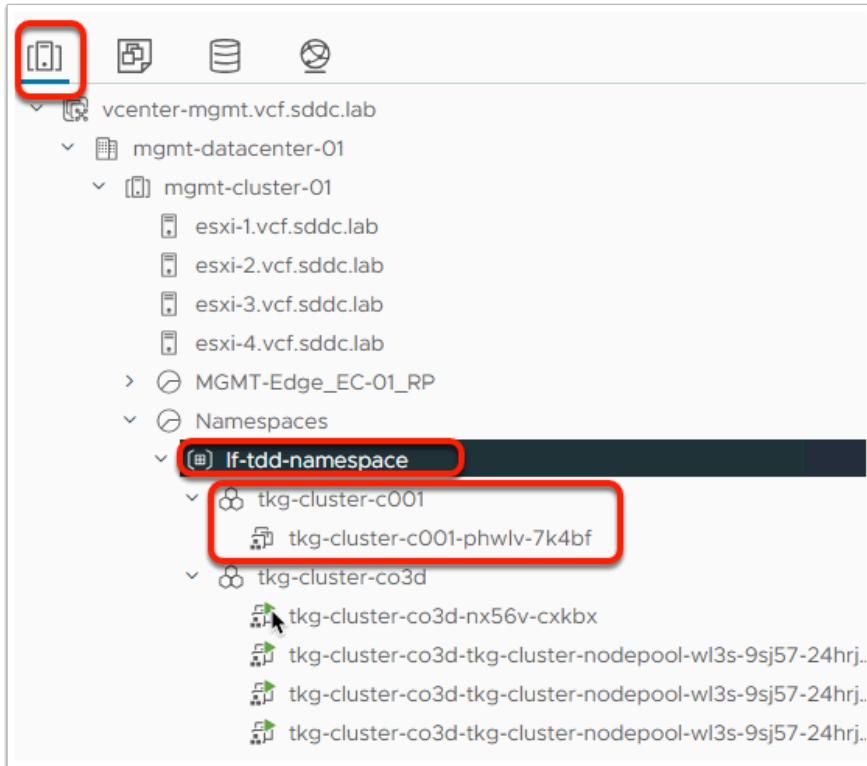
A red arrow points downwards along the right side of the slide bar.

If you click on the **tkg-cluster-c001** name, you will get visual of all the **Cluster Details**

Scroll down the slide bar in the UI

Also, you can track the new cluster appearance from **Inventory --> VMs and Clusters** --> under **lf-tdd-namespace**

# Create a TKG Workload Cluster



... starting with the Control Plane provisioning

## Create a TKG Workload Cluster

The screenshot shows the vSphere Web Client interface. In the navigation pane, under 'vcenter-mgmt.vcf.sddc.lab / mgmt-datacenter-01 / mgmt-cluster-01', several ESXi hosts are listed: esxi-1.vcf.sddc.lab, esxi-2.vcf.sddc.lab, esxi-3.vcf.sddc.lab, and esxi-4.vcf.sddc.lab. Below these, there are two network resources: MGMT-Edge\_EC-01\_RP and Namespaces. The 'Namespaces' section is expanded, showing a folder named 'If-tdd-namespace'. Inside this folder, a new cluster named 'tkg-cluster-c001' is being created, indicated by a red box around the cluster name. Underneath 'tkg-cluster-c001', two sub-clusters are listed: tkg-cluster-c001-phwlv-7k4bf and tkg-cluster-c001-tkg-cluster-nodepool-c001-grql9-58l8c.

... and ending with the Workload nodes added

All seems good, so let's get back in the PS Terminal and check the cluster(s) residing in the If-tdd-namespace

## Create a TKG Workload Cluster

```
> kubectl get clusters
NAME          CLUSTERCLASS      PHASE      AGE        VERSION
tkg-cluster-c001  tanzukubernetescluster  Provisioned  3h22m    v1.29.4+vmware.3-fips.1
tkg-cluster-co3d  tanzukubernetescluster  Provisioned  24h       v1.29.4+vmware.3-fips.1
>
```

```
kubectl get clusters
```

You can do the same, but getting a bit more info with the tanzu cli

# Create a TKG Workload Cluster

```
> tanzu cluster list --namespace lf-tdd-namespace
NAME      NAMESPACE      STATUS      CONTROLPLANE      WORKERS      KUBERNETES      ROLES      PLAN      TKR
tkg-cluster-c001  lf-tdd-namespace  running  1/1      1/1      v1.29.4+vmware.3-fips.1  <none>      v1.29.4---vmware.3-fips.1-tkg.1
tkg-cluster-c03d  lf-tdd-namespace  running  1/1      3/3      v1.29.4+vmware.3-fips.1  <none>      v1.29.4---vmware.3-fips.1-tkg.1
>
```

```
tanzu cluster list --namespace lf-tdd-namespace
```

Okay, you are now ready to go to the next section!

# Manage a TKG Workload Cluster

## Task description and objectives

Your task is to inspect the health of the deployed Tanzu Kubernetes Grid workload cluster you recently created.

The workload cluster currently has one worker node, which is not sufficient for the purpose of the cluster. You must also increase the number of worker nodes so that new applications can be deployed. The number of worker nodes must be increased to two.

As a part of this lab, you must allow privileged workload deployment in the Tanzu Kubernetes Grid workload cluster

 **IMPORTANT:** This exercise is being done with administrator@vsphere.local permissions set over the lf-tdd-namespace

Make sure you are logged in your **Supervisor**

## Manage a TKG Workload Cluster

Aligned with the previous section, make sure you are connected to your Supervisor!

```
> kubectl config get-contexts
CURRENT  NAME          CLUSTER      AUTHINFO                                     NAMESPACE
*        10.80.0.2       10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
          lf-tdd-namespace   10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
          svc-cci-service-domain-c9  10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
          svc-contour-domain-c9   10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
          svc-external-dns-domain-c9  10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
          svc-harbor-domain-c9    10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
          svc-tkg-domain-c9      10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
          svc-velero-domain-c9    10.80.0.2    wcp:10.80.0.2:administrator@vsphere.local
> kubectl config use-context lf-tdd-namespace
Switched to context "lf-tdd-namespace".
>
```

Navigate to your working Namespace **lf-tdd-namespace** where you will provision an additional **workload cluster**

```
kubectl config get-contexts
```

```
kubectl config use-context lf-tdd-namespace
```

Make sure to log in to your Supervisor context with Tanzu CLI as well

## Manage a TKG Workload Cluster

```
> tanzu context use lf-tdd-supervisor
[i] Successfully activated context 'lf-tdd-supervisor'
[i] Checking for required plugins for context 'lf-tdd-supervisor'...
[i] All required plugins are already installed and up-to-date
>
```

```
tanzu context use lf-tdd-supervisor
```

## Manage a TKG Workload Cluster

Once again, assure your workload clusters are in place:

```
> tanzu cluster list --namespace lf-tdd-namespace
NAME      NAMESPACE   STATUS    CONTROLPLANE WORKERS KUBERNETES          ROLES   PLAN   TKR
tkg-cluster-c001  lf-tdd-namespace  running  1/1        1/1    v1.29.4+vmware.3-fips.1 <none> v1.29.4---vmware.3-fips.1-tkg.1
tkg-cluster-c03d  lf-tdd-namespace  running  1/1        3/3    v1.29.4+vmware.3-fips.1 <none> v1.29.4---vmware.3-fips.1-tkg.1
>
```

```
tanzu cluster list --namespace lf-tdd-namespace
```

## Manage a TKG Workload Cluster

Retrieve the administrator **kubeconfig** file for the **tkg-cluster-c001** workload cluster.

```
> tanzu cluster kubeconfig get tkg-cluster-c001 --namespace=lf-tdd-namespace --admin
Credentials of cluster 'tkg-cluster-c001' have been saved
You can now access the cluster by running 'kubectl config use-context tkg-cluster-c001-admin@tkg-cluster-c001'
>
```

```
tanzu cluster kubeconfig get tkg-cluster-c001 --namespace=lf-tdd-namespace --admin
```

Now set the **kubectl** context to point to your new workload cluster. You can do this following Option A or Option B below:

**Option A:** Use the **kubectl** tool

```
>  
> kubectl config use-context tkg-cluster-c001-admin@tkg-cluster-c001  
Switched to context "tkg-cluster-c001-admin@tkg-cluster-c001".  
>
```

```
kubectl config use-context tkg-cluster-c001-admin@tkg-cluster-c001
```

### Option B: Use the **kubectx** tool

Pro tip: Alternatively, for changing the default kubectl context, you can use the **kubectx** tool, available on your PS Terminal.

It can list the available contexts for you

## Manage a TKG Workload Cluster

```
>  
> kubectx  
10.80.0.2  
lf-tdd-namespace  
svc-cci-service-domain-c9  
svc-contour-domain-c9  
svc-external-dns-domain-c9  
svc-harbor-domain-c9  
svc-tkg-domain-c9  
svc-velero-domain-c9  
tkg-cluster-c001-admin@tkg-cluster-c001  
>
```

```
kubectx
```

The active/default context is shown in green.

And you can easily switch to what's needed for you

# Manage a TKG Workload Cluster

```
> kubectl tkg-cluster-c001-admin@tkg-cluster-c001
> Switched to context "tkg-cluster-c001-admin@tkg-cluster-c001".
>
> Kubectl
10.80.0.2
lf-tdd-namespace
svc-cci-service-domain-c9
svc-contour-domain-c9
svc-external-dns-domain-c9
svc-harbor-domain-c9
svc-tkg-domain-c9
svc-velero-domain-c9
tkg-cluster-c001-admin@tkg-cluster-c001
>
```

```
kubectl tkg-cluster-c001-admin@tkg-cluster-c001
```

In your workload cluster context:

Get info on the nodes

# Manage a TKG Workload Cluster

```
> kubectl get nodes -o wide
NAME           STATUS   ROLES      AGE    VERSION      INTERNAL-IP     EXTERNAL-IP   OS-IMAGE
tkg-cluster-c001-phwlv-7k4bf   Ready   control-plane   3h46m   v1.29.4+vmware.3-fips.1   10.244.0.114   <none>       Ubuntu 22.04.4 LTS
tkg-cluster-c001-tkg-cluster-nodepool-c001-grql9-5818c-h628r   Ready   <none>      3h37m   v1.29.4+vmware.3-fips.1   10.244.0.115   <none>       Ubuntu 22.04.4 LTS
>
```

```
kubectl get nodes -o wide
```

Please, note the VM name of your workload TKG cluster node!

In the example above its name ends with **-h628r** Yours should be different. Just note down whatever your end-of-name is.

Get info on the pods

# Manage a TKG Workload Cluster

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
Kube-system	antrea-agent-47mbm	2/2	Running	0	3h41m
Kube-system	antrea-agent-qc9jr	2/2	Running	0	3h35m
Kube-system	antrea-controller-7d47cc598b-dspsq	1/1	Running	0	3h41m
Kube-system	coredns-66d8b77bb7-6z2kj	1/1	Running	0	3h44m
Kube-system	coredns-66d8b77bb7-7xr19	1/1	Running	0	3h44m
Kube-system	docker-registry-tkg-cluster-c001-phwlv-7k4bf	1/1	Running	0	3h44m
Kube-system	docker-registry-tkg-cluster-c001-tkg-cluster-nodepool-c001-grql9-58l8c-h628r	1/1	Running	0	3h35m
Kube-system	etcd-tkg-cluster-c001-phwlv-7k4bf	1/1	Running	0	3h44m
Kube-system	kube-apiserver-tkg-cluster-c001-phwlv-7k4bf	1/1	Running	0	3h44m
Kube-system	kube-controller-manager-tkg-cluster-c001-phwlv-7k4bf	1/1	Running	0	3h44m
Kube-system	kube-proxy-8wkbd	1/1	Running	0	3h44m
Kube-system	kube-proxy-x8m8f	1/1	Running	0	3h35m
Kube-system	kube-scheduler-tkg-cluster-c001-phwlv-7k4bf	1/1	Running	0	3h44m
Kube-system	metrics-server-74c854cc9d-tw6pb	1/1	Running	0	3h41m
Kube-system	snapshot-controller-775c78dff7-ph5l4	1/1	Running	0	3h41m
secretgen-controller	secretgen-controller-57dc66946-drrns	1/1	Running	0	3h41m
tkg-system	kapp-controller-75496d6bb5-ltkdb	2/2	Running	0	3h41m
tkg-system	tanzu-capabilities-controller-manager-64bfc9dd4-zskmf	1/1	Running	0	3h40m
vmware-system-antrea	register-placeholder-xrkcb	0/1	Completed	0	3h41m
vmware-system-auth	guest-cluster-auth-svc-r7zmk	1/1	Running	0	3h40m
vmware-system-cloud-provider	guest-cluster-cloud-provider-6565747f55-nmf98	1/1	Running	0	3h41m
vmware-system-csi	vsphere-csi-controller-9c545d675-c97js	7/7	Running	0	3h41m
vmware-system-csi	vsphere-csi-node-8pdmw	3/3	Running	3 (3h39m ago)	3h41m
vmware-system-csi	vsphere-csi-node-ns5cq	3/3	Running	0	3h35m

```
kubectl get pods -A
```

## Scale a TKG Workload Cluster

You add capacity to the Tanzu Kubernetes Grid workload cluster by scaling the number of worker nodes.

NAME	NAMESPACE	STATUS	CONTROLPLANE	WORKERS	KUBERNETES	ROLES	PLAN	TKR
tkg-cluster-c001	lf-tdd-namespace	running	1/1	1/1	v1.29.4+vmware.3-fips.1	<none>	v1.29.4---vmware.3-fips.1-tkg.1	
tkg-cluster-co3d	lf-tdd-namespace	running	1/1	3/3	v1.29.4+vmware.3-fips.1	<none>	v1.29.4---vmware.3-fips.1-tkg.1	

```
tanzu cluster list --namespace lf-tdd-namespace
```

You have just one worker in the workload cluster.

Let's add another one via Terminal:

## Scale a TKG Workload Cluster

```
> tanzu cluster scale tkg-cluster-c001 --namespace lf-tdd-namespace --worker-machine-count 2
Workload cluster 'tkg-cluster-c001' is being scaled
>
```

```
tanzu cluster scale tkg-cluster-c001 --namespace lf-tdd-namespace --worker-machine-count 2
```

The scale operation takes place in the background.

Check the status of the workload cluster, being expanded with a workload node:

## Scale a TKG Workload Cluster

```
> tanzu cluster list --namespace lf-tdd-namespace
NAME          NAMESPACE      STATUS    CONTROLPLANE WORKERS KUBERNETES      ROLES   PLAN   TCR
tkg-cluster-c001  lf-tdd-namespace  updating  1/1        1/2     v1.29.4+vmware.3-fips.1  <none>  v1.29.4---vmware.3-fips.1-tkg.1
tkg-cluster-co3d  lf-tdd-namespace  running   1/1        3/3     v1.29.4+vmware.3-fips.1  <none>  v1.29.4---vmware.3-fips.1-tkg.1
>
```

```
tanzu cluster list --namespace lf-tdd-namespace
```

It takes a couple of minutes to provision the second workload node and add it to the cluster.

Actually, you can get a bit more wider picture about it:

## Scale a TKG Workload Cluster

```
> kubectl 10.80.0.2
Switched to context "10.80.0.2".
> kubectl get machines -n lf-tdd-namespace --watch
NAME          CLUSTER      NODENAME      PROVIDERID
tkg-cluster-c001-phwlv-7k4bf  tkg-cluster-c001  tkg-cluster-c001-phwlv-7k4bf  vsphere://42001289-94dd-
7051-59c4-14587047382c  Running   4h53m  v1.29.4+vmware.3-fips.1  3
tkg-cluster-c001-tkg-cluster-nodepool-c001-grql9-58l8c-h628r  tkg-cluster-c001  tkg-cluster-c001-tkg-cluster-nodepool-c001-grql9-58l8c-h628r  vsphere://42001e11-98aa-
1926-1f98-4b4fa4c13be0  Running   4h36m  v1.29.4+vmware.3-fips.1
tkg-cluster-c001-tkg-cluster-nodepool-c001-grql9-58l8c-kvbx5  tkg-cluster-c001  tkg-cluster-c001-tkg-cluster-nodepool-c001-grql9-58l8c-kvbx5  vsphere://42007ff1-8c17-
4f97-0bd5-96279dd3e88d  Running   6m58s  v1.29.4+vmware.3-fips.1  4
tkg-cluster-co3d-nx56v-cxkbx  tkg-cluster-co3d  tkg-cluster-co3d-nx56v-cxkbx  vsphere://42001069-612c-
d124-52a3-ecd6ef319ce9  Running   26h    v1.29.4+vmware.3-fips.1  5
tkg-cluster-co3d-tkg-cluster-nodepool-wl3s-9sj57-24hrj-68htc  tkg-cluster-co3d  tkg-cluster-co3d-tkg-cluster-nodepool-wl3s-9sj57-24hrj-68htc  vsphere://420079bb-4654-
dd45-f483-34cle0b6e173  Running   25h    v1.29.4+vmware.3-fips.1
tkg-cluster-co3d-tkg-cluster-nodepool-wl3s-9sj57-24hrj-qdmm9  tkg-cluster-co3d  tkg-cluster-co3d-tkg-cluster-nodepool-wl3s-9sj57-24hrj-qdmm9  vsphere://420078c3-5c95-
74ef-f01e-176526da00e7  Running   25h    v1.29.4+vmware.3-fips.1
tkg-cluster-co3d-tkg-cluster-nodepool-wl3s-9sj57-24hrj-zvd8l  tkg-cluster-co3d  tkg-cluster-co3d-tkg-cluster-nodepool-wl3s-9sj57-24hrj-zvd8l  vsphere://4200789b-c221-
9e90-778e-1ba363636373 Running   25h    v1.29.4+vmware.3-fips.1
>
```

1. Change your kubectl context to the Supervisor and
2. Monitor the cluster API resources.

```
kubectx 10.80.0.2
```

```
kubectl get machines -n lf-tdd-namespace --watch
```

3. At the beginning of the section, you were asked to note the name of your VM representing the initial workload node. It ended with **-h628r**.

- Now you can distinguish the other node, being added to the cluster - a different VM, which name ends on `-kvb5`. Just as an example.
- Please wait for the new VM phase to change from **Provisioning** to **Running** before continuing.
- Press **Ctrl+C** to exit the **watch** command.

Once you get the new workload node ready, please change your kubeconfig context back to your workload cluster and inspect it:

## Scale a TKG Workload Cluster

```
> kubectx tkg-cluster-c001-admin@tkg-cluster-c001
✓ Switched to context "tkg-cluster-c001-admin@tkg-cluster-c001".
> tanzu cluster list --namespace lf-tdd-namespace
NAME          NAMESPACE      STATUS   CONTROLPLANE WORKERS   KUBERNETES      ROLES    PLAN   TKR
tka-cluster-c001 lf-tdd-namespace running  1/1        2/2       v1.29.4+vmware.3-fips.1 <none>    v1.29.4---vmware.3-fips.1-tkg.1
tkg-cluster-c03d lf-tdd-namespace running  1/1        3/3       v1.29.4+vmware.3-fips.1 <none>    v1.29.4---vmware.3-fips.1-tkg.1
>
```

```
kubectx tkg-cluster-c001-admin@tkg-cluster-c001
```

```
tanzu cluster list --namespace lf-tdd-namespace
```

The output shows STATUS running and WORKERS 2/2.

You can also double-check this in the vSphere UI:

In your **Workload Management --> Namespaces --> If-tdd-namespace** pane, you will see the new cluster Toplogy update. Make sure you scroll the slide bar to the bottom to reveal the info:

# Scale a TKG Workload Cluster

If-tdd-namespace > Tanzu Kubernetes Grid service > tkg-cluster-c001

tkg-cluster-c001 [VIEW YAML](#) [DOWNLOAD KUBECONFIG FILE](#)

Cluster health status

Last Transition Time	Type	Status	Severity	Reason	Message
Aug 15, 2024, 9:17:45 AM	ControlPlaneReady	True			
Aug 15, 2024, 9:17:45 AM	ControlPlaneInitialized	True			
Aug 15, 2024, 9:17:45 AM	Ready	True			
Aug 15, 2024, 9:03:23 AM	InfrastructureReady	True			
Aug 15, 2024, 9:03:14 AM	TopologyReconciled	True			
Aug 15, 2024, 9:03:07 AM	UpdatesAvailable	False	Info		AlreadyUpToDate

Items per page: 10 6 items

Events

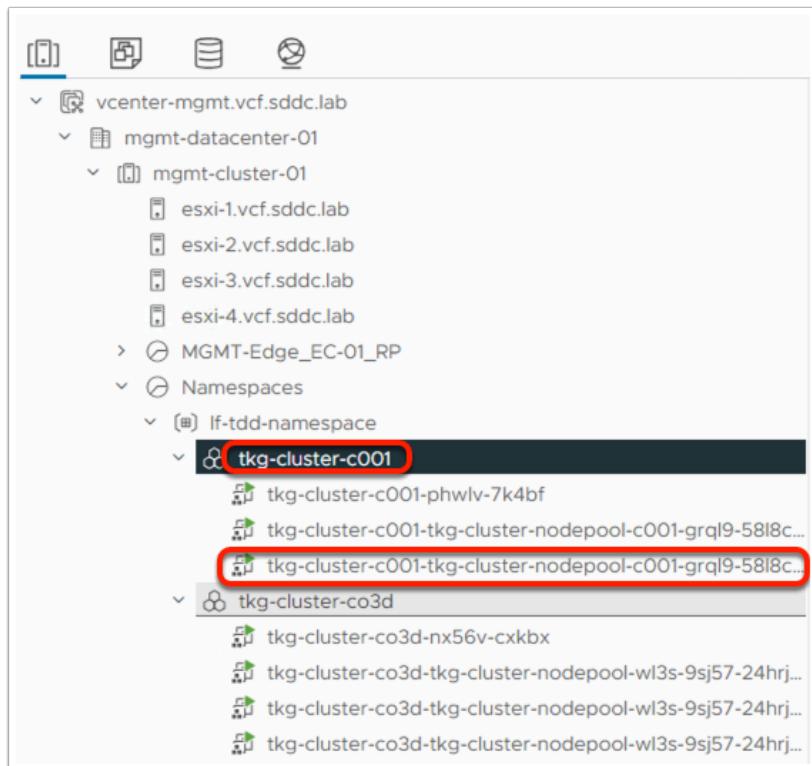
Type	Reason	Age	From	Message
Normal	TopologyUpdate	34 minutes	topology/cluster	Updated "MachineDeployment/tkg-cluster-c001-tkg-cluster-nodepool-c001-grql9"

Filter

Check the:

- Cluster health status
- Events

Additionally, while in the vSphere client:



From **Inventory** --> **VMs and Clusters** --> under **If-tdd-namespace**

You will see the second workload node added to your recent cluster.

# Packaging in a TKG Workload Cluster

## Task description and objectives

Your task is to examine the functionality of Tanzu CLI and its Package plug-in to manage repositories and packages in a TKG workload cluster. You will get the experience of working with Tanzu Package CLI Plug-In and repositories. To make sure hosting modern Applications on top of your workload TKG cluster will go smooth, a bunch of packages must be installed in addition.

### **IMPORTANT:**

This exercise is being done with `administrator@vsphere.local` permissions set over the **If-tdd-namespace**. Make sure you are logged in your **Supervisor**

Your **kubeconfig** context must be set to the **tkg-cluster-c001** cluster!

## Working context

Make sure you are working on the **tkg-cluster-c001** workload cluster you created and managed in previous steps

```
> kubectx ①  
10.80.0.2  
lf-tdd-namespace  
svc-cci-service-domain-c9  
svc-contour-domain-c9  
svc-external-dns-domain-c9  
svc-harbor-domain-c9  
svc-tkg-domain-c9  
svc-velero-domain-c9  
tkg-cluster-c001-admin@tkg-cluster-c001  
> kubectx tkg-cluster-c001-admin@tkg-cluster-c001 ②  
✓ Switched to context "tkg-cluster-c001-admin@tkg-cluster-c001". ←
```

kubectx

kubectx tkg-cluster-c001-admin@tkg-cluster-c001

# Packaging

In your workload cluster context: Get info on the package repositories, if any available at this point

```
> tanzu package repository list -A
```

NAMESPACE	NAME	SOURCE	STATUS

```
>
```

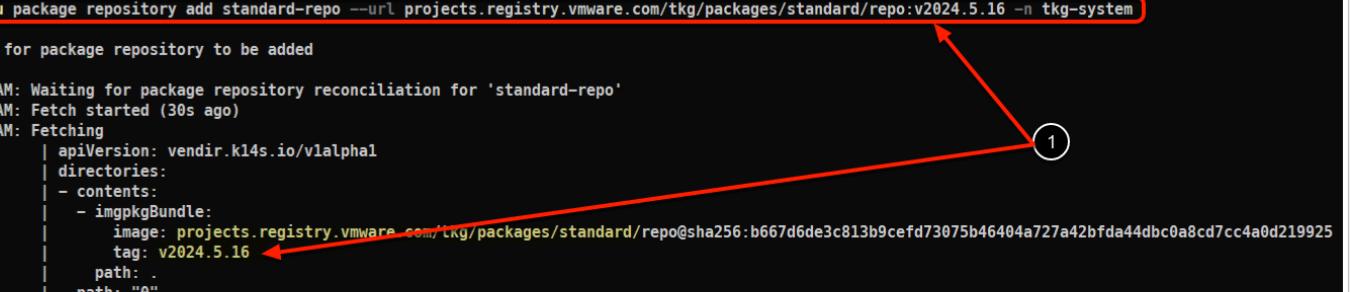
```
tanzu package repository list -A
```

No repos are configured yet. Let's add some:

# Packaging

```
> tanzu package repository add standard-repo --url projects.registry.vmware.com/tkg/packages/standard/repo:v2024.5.16 -n tkg-system
```

Waiting for package repository to be added

7:36:53AM: Waiting for package repository reconciliation for 'standard-repo'  
7:36:54AM: Fetch started (30s ago)  
7:37:24AM: Fetching  
| apiVersion: vendir.k14s.io/v1alpha1  
| directories:  
| - contents:  
| - imgpkgBundle:  
| image: projects.registry.vmware.com/tkg/packages/standard/repo@sha256:b667d6de3c813b9cefd73075b46404a727a42bfda44dbc0a8cd7cc4a0d219925  
| tag: v2024.5.16  
| path: .  
| path: "0"  
| kind: LockConfig  
7:37:24AM: Fetch succeeded  
7:37:37AM: Template succeeded  
7:37:37AM: Deploy started (12s ago)  
7:37:49AM: Deploying

The screenshot is truncated!

```
| 7:37:48AM: ---- applying complete [171/171 done] ----  
| 7:37:48AM: ---- waiting complete [171/171 done] ----  
| Succeeded  
7:37:49AM Deploy succeeded
```

```
tanzu package repository add tanzu-standard --url projects.registry.vmware.com/tkg/packages/standard/repo:v2024.5.16 -n tkg-system
```

Deploy succeeded:

After adding this package repo a reconciliation process kicks-in, and you can see it as **tanzu-standard** under namespace **tkg-system** in your workload cluster, in about of couple of minutes.

# Packaging

The **tanzu-standard** package repository appears in the **tkg-system** namespace with STATUS **Reconcile succeeded**

NAMESPACE	NAME	SOURCE	STATUS
tkg-system	standard-repo	(imgpkg) projects.registry.vmware.com/tkg/packages/standard/repo:v2024.5.16	Reconcile succeeded

```
tanzu package repository list -A
```

The **tanzu-standard** package repository appears in the **tkg-system** namespace with **version v2024.5.16** and **STATUS Reconcile succeeded**

# Packaging

Okay, after adding a **tanzu-standard** repo, let's inspect what's available from it as packages

NAME	DISPLAY-NAME
cert-manager.tanzu.vmware.com	cert-manager
cluster-autoscaler.tanzu.vmware.com	autoscaler
contour.tanzu.vmware.com	contour
external-csi-snapshot-webhook.tanzu.vmware.com	external-csi-snapshot-webhook
external-dns.tanzu.vmware.com	external-dns
fluent-bit.tanzu.vmware.com	fluent-bit
fluxcd-helm-controller.tanzu.vmware.com	Flux Helm Controller
fluxcd-kustomize-controller.tanzu.vmware.com	Flux Kustomize Controller
fluxcd-source-controller.tanzu.vmware.com	Flux Source Controller
grafana.tanzu.vmware.com	grafana
harbor.tanzu.vmware.com	harbor
multus-cni.tanzu.vmware.com	multus-cni
prometheus.tanzu.vmware.com	prometheus
vsphere-pv-csi-webhook.tanzu.vmware.com	vsphere-pv-csi-webhook
whereabouts.tanzu.vmware.com	whereabouts

```
tanzu package available list
```

A list of all available packages and their latest versions is seen.

If you focus on any of those, you can retrieve a bunch of additional information, for example **cert-manager**:

# Packaging

```
> tanzu package available list cert-manager.tanzu.vmware.com
NAME          VERSION      RELEASED-AT
cert-manager.tanzu.vmware.com 1.1.0+vmware.1-tkg.2 2020-11-24 18:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.1.0+vmware.2-tkg.1 2020-11-24 18:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.10.1+vmware.1-tkg.1 2021-10-29 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.10.1+vmware.1-tkg.2 2021-10-29 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.10.2+vmware.1-tkg.1 2023-01-11 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.11.1+vmware.1-tkg.1 2023-01-11 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.12.10+vmware.1-tkg.1 2023-06-15 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.12.2+vmware.1-tkg.1 2023-06-15 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.12.2+vmware.2-tkg.2 2023-06-15 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.5.3+vmware.2-tkg.1 2021-08-23 17:22:51 +0000 UTC
cert-manager.tanzu.vmware.com 1.5.3+vmware.4-tkg.1 2021-08-23 17:22:51 +0000 UTC
cert-manager.tanzu.vmware.com 1.5.3+vmware.6-tkg.1 2021-08-23 17:22:51 +0000 UTC
cert-manager.tanzu.vmware.com 1.5.3+vmware.7-tkg.1 2021-08-23 17:22:51 +0000 UTC
cert-manager.tanzu.vmware.com 1.5.3+vmware.7-tkg.2 2021-08-23 17:22:51 +0000 UTC
cert-manager.tanzu.vmware.com 1.5.3+vmware.7-tkg.3 2021-08-23 17:22:51 +0000 UTC
cert-manager.tanzu.vmware.com 1.7.2+vmware.1-tkg.1 2021-10-29 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.7.2+vmware.3-tkg.1 2021-10-29 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.7.2+vmware.3-tkg.2 2021-10-29 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.7.2+vmware.3-tkg.3 2021-10-29 12:00:00 +0000 UTC
>
```

A list of all available versions for the **cert-manager** package is seen

```
tanzu package available list cert-manager.tanzu.vmware.com
```

But you can retrieve some more and get details about the **cert-manager** package:

# Packaging

```
> tanzu package available get cert-manager.tanzu.vmware.com
NAME: cert-manager.tanzu.vmware.com
DISPLAY-NAME: cert-manager
CATEGORIES: - certificate management
SHORT-DESCRIPTION: Certificate management
LONG-DESCRIPTION: Provides certificate management provisioning within the cluster
PROVIDER: VMware
MAINTAINERS: - name: Nicholas Seemiller
SUPPORT-DESCRIPTION: Support provided by VMware for deployment on Tanzu clusters. Best-effort support
for deployment on any conformant Kubernetes cluster. Contact support by opening
a support request via VMware Cloud Services or my.vmware.com.

VERSION RELEASED-AT
1.1.0+vmware.1-tkg.2 2020-11-24 18:00:00 +0000 UTC
1.1.0+vmware.2-tkg.1 2020-11-24 18:00:00 +0000 UTC
1.10.1+vmware.1-tkg.1 2021-10-29 12:00:00 +0000 UTC
1.10.1+vmware.1-tkg.2 2021-10-29 12:00:00 +0000 UTC
1.10.2+vmware.1-tkg.1 2023-01-11 12:00:00 +0000 UTC
1.11.1+vmware.1-tkg.1 2023-01-11 12:00:00 +0000 UTC
1.12.10+vmware.1-tkg.1 2023-06-15 12:00:00 +0000 UTC
1.12.2+vmware.1-tkg.1 2023-06-15 12:00:00 +0000 UTC
1.12.2+vmware.2-tkg.2 2023-06-15 12:00:00 +0000 UTC
1.5.3+vmware.2-tkg.1 2021-08-23 17:22:51 +0000 UTC
1.5.3+vmware.4-tkg.1 2021-08-23 17:22:51 +0000 UTC
1.5.3+vmware.6-tkg.1 2021-08-23 17:22:51 +0000 UTC
1.5.3+vmware.7-tkg.1 2021-08-23 17:22:51 +0000 UTC
1.5.3+vmware.7-tkg.2 2021-08-23 17:22:51 +0000 UTC
1.5.3+vmware.7-tkg.3 2021-08-23 17:22:51 +0000 UTC
1.7.2+vmware.1-tkg.1 2021-10-29 12:00:00 +0000 UTC
1.7.2+vmware.3-tkg.1 2021-10-29 12:00:00 +0000 UTC
1.7.2+vmware.3-tkg.2 2021-10-29 12:00:00 +0000 UTC
1.7.2+vmware.3-tkg.3 2021-10-29 12:00:00 +0000 UTC
>
```

```
tanzu package available get cert-manager.tanzu.vmware.com
```

Information about the cert-manager package appears, including:

- package name
- categories
- description
- support details
- versions

You can List all the installed packages in the cluster that were installed automatically during the cluster creation appear.

# Packaging

tanzu package installed list -A				
NAMESPACE	NAME	PACKAGE-NAME	PACKAGE-VERSION	STATUS
vmware-system-tkg	tkg-cluster-c001-antra	antrea.tanzu.vmware.com	1.13.3+vmware.3-tkg.3-vmware	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-capabilities	capabilities.tanzu.vmware.com	0.33.1+vmware.1	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-gateway-api	gateway-api.tanzu.vmware.com	1.0.0+vmware.1-tkg.2-vmware	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-guest-cluster-auth-service	guest-cluster-auth-service.tanzu.vmware.com	1.3.3+tkg.2-vmware	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-metrics-server	metrics-server.tanzu.vmware.com	0.6.2+vmware.3-tkg.6-vmware	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-pinniped	pinniped.tanzu.vmware.com	0.25.0+vmware.2-tkg.2-vmware	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-secretgen-controller	secretgen-controller.tanzu.vmware.com	0.16.1+vmware.1-tkg.1-vmware	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-vsphere-cpi	vsphere-cpi.tanzu.vmware.com	1.29.0+vmware.1-tkg.1-vmware	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-vsphere-pv-csi	vsphere-pv-csi.tanzu.vmware.com	3.2.0+vmware.1-tkg.2-vmware	Reconcile succeeded

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

# Packaging

So obviously, **cert-manager** is not installed in your workload cluster. Although **cert-manager** is installed automatically in management clusters, you must install it using the `tanzu package` CLI plug-in in workload clusters.

The Installation process is quite neat:

- you create a namespace to fetch the package files into your cluster
- then install the package itself from there, through a reconciliation process

Let's start by creating a cert-manager namespace, where the package needs to be installed on top of your cluster:

# Packaging

```
> kubectl create ns cert-manager
namespace/cert-manager created
>
```

```
kubectl create ns cert-manager
```

# Packaging

Typically, we install the most recent version.

As a refresher, let's get the versions available:

# Packaging

```
> tanzu package available list cert-manager.tanzu.vmware.com 1
NAME          VERSION      RELEASED-AT
cert-manager.tanzu.vmware.com 1.1.0+vmware.1-tkg.2 2020-11-24 18:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.1.0+vmware.2-tkg.1 2020-11-24 18:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.10.1+vmware.1-tkg.1 2021-10-29 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.10.1+vmware.1-tkg.2 2021-10-29 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.10.2+vmware.1-tkg.1 2023-01-11 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.11.1+vmware.1-tkg.1 2023-01-11 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.12.10+vmware.1-tkg.1 2023-06-15 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.12.2+vmware.1-tkg.1 2023-06-15 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.12.2+vmware.2-tkg.2 2023-06-15 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.5.3+vmware.2-tkg.1 2021-08-23 17:22:51 +0000 UTC
cert-manager.tanzu.vmware.com 1.5.3+vmware.4-tkg.1 2021-08-23 17:22:51 +0000 UTC
cert-manager.tanzu.vmware.com 1.5.3+vmware.6-tkg.1 2021-08-23 17:22:51 +0000 UTC
cert-manager.tanzu.vmware.com 1.5.3+vmware.7-tkg.1 2021-08-23 17:22:51 +0000 UTC
cert-manager.tanzu.vmware.com 1.5.3+vmware.7-tkg.2 2021-08-23 17:22:51 +0000 UTC
cert-manager.tanzu.vmware.com 1.5.3+vmware.7-tkg.3 2021-08-23 17:22:51 +0000 UTC
cert-manager.tanzu.vmware.com 1.7.2+vmware.1-tkg.1 2021-10-29 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.7.2+vmware.3-tkg.1 2021-10-29 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.7.2+vmware.3-tkg.2 2021-10-29 12:00:00 +0000 UTC
cert-manager.tanzu.vmware.com 1.7.2+vmware.3-tkg.3 2021-10-29 12:00:00 +0000 UTC
>
```

```
tanzu package available list cert-manager.tanzu.vmware.com
```

So from this list, you can pick-up a version for install.

Let's get the most recent one: `1.12.2+vmware.2-tkg.2`

# Packaging

```
> tanzu package install cert-manager -p cert-manager.tanzu.vmware.com -n cert-manager -v 1.12.2+vmware.2-tkg.2
8:15:46AM: Creating service account 'cert-manager-cert-manager-sa'
8:15:46AM: Creating cluster admin role 'cert-manager-cert-manager-cluster-role'
8:15:46AM: Creating cluster role binding 'cert-manager-cert-manager-cluster-rolebinding'
8:15:46AM: Creating overlay secrets
8:15:46AM: Creating package install resource
8:15:46AM: Waiting for PackageInstall reconciliation for 'cert-manager'
8:15:48AM: Fetch started (17s ago)
8:16:06AM: Fetching
| apiVersion: vendir.k14s.io/v1alpha1
| directories:
| - contents:
|   - imgpkgBundle:
|     image: projects.registry.vmware.com/tkg/packages/standard/cert-manager@sha256:15563d5759851b6cf59a2e0891ddc2d6d0b204e68130914adf8e62bfb4762a24
|     path: .
|     path: "0"
|     kind: LockConfig
8:16:06AM: Fetch succeeded
8:16:07AM: Template succeeded
8:16:07AM: Deploy started (1s ago)
8:16:09AM: Deploying
```

The output on the screenshot is truncated!

```

| 8:16:42AM: L ok: waiting on replicaset/cert-manager-webhook-6855846455 (apps/v1) namespace: cert-manager
| 8:16:42AM: L ongoing: waiting on pod/cert-manager-webhook-6855846455-qqfk4 (v1) namespace: cert-manager
| 8:16:42AM:     ^ Condition Ready is not True (False)
8:16:45AM: Deploy succeeded
>

```

```
tanzu package install cert-manager -p cert-manager.tanzu.vmware.com -n cert-manager -v 1.12.2+vmware.2-tkg.2
```

Installation takes up to 5min.

The process involves kapp controller at the backend, implementing the following:

- Creating a dedicated service account
- Creating a cluster admin role
- Creating cluster role binding
- Creating overlay secrets
- Creating package install resource
- Reconciling changes until package gets installed

You can check the changes and get info update, after the cert-manager gets successfully installed

From list of installed packages perspective:

## Packaging

NAMESPACE	NAME	PACKAGE-NAME	PACKAGE-VERSION	STATUS
cert-manager	cert-manager	cert-manager.tanzu.vmware.com	1.12.2+vmware.2-tkg.2	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-antrea	antrea.tanzu.vmware.com	1.13.3+vmware.3-tkg.3-vmware	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-capabilities	capabilities.tanzu.vmware.com	0.33.1+vmware.1	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-gateway-api	gateway-api.tanzu.vmware.com	1.0.0+vmware.1-tkg.2-vmware	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-guest-cluster-auth-service	guest-cluster-auth-service.tanzu.vmware.com	1.3.3+tkg.2-vmware	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-metrics-server	metrics-server.tanzu.vmware.com	0.6.2+vmware.3-tkg.6-vmware	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-pinniped	pinniped.tanzu.vmware.com	0.25.0+vmware.2-tkg.2-vmware	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-secretgen-controller	secretgen-controller.tanzu.vmware.com	0.16.1+vmware.1-tkg.1-vmware	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-vsphere-cpi	vsphere-cpi.tanzu.vmware.com	1.29.0+vmware.1-tkg.1-vmware	Reconcile succeeded
vmware-system-tkg	tkg-cluster-c001-vsphere-pv-csi	vsphere-pv-csi.tanzu.vmware.com	3.2.0+vmware.1-tkg.2-vmware	Reconcile succeeded

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

From installation namespace perspective:

# Packaging

```
> tanzu package installed list cert-manager -n cert-manager ①
NAME          PACKAGE-NAME          PACKAGE-VERSION      STATUS
cert-manager   cert-manager.tanzu.vmware.com 1.12.2+vmware.2-tkg.2 Reconcile succeeded
> tanzu package installed get cert-manager -n cert-manager ②
NAMESPACE:    cert-manager
NAME:         cert-manager
PACKAGE-NAME: cert-manager.tanzu.vmware.com
PACKAGE-VERSION: 1.12.2+vmware.2-tkg.2
STATUS:       Reconcile succeeded
CONDITIONS:  - status: "True"
              - type: ReconcileSucceeded
>
```

```
tanzu package installed list -n cert-manager
```

```
tanzu package installed get cert-manager -n cert-manager
```

Upon successful installation of **cert-manager** package, you can check:

- pods
- services
- deployments
- replicaset

# Packaging

```
> kubectl get all -n cert-manager
NAME                                         READY   STATUS    RESTARTS   AGE
pod/cert-manager-7f545657f4-tnplt           1/1     Running   0          2d1h
pod/cert-manager-cainjector-56564f68dd-stmvn 1/1     Running   0          2d1h
pod/cert-manager-webhook-6855846455-qqfk4   1/1     Running   0          2d1h

NAME          TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
service/cert-manager   ClusterIP   10.107.183.131  <none>        9402/TCP   2d1h
service/cert-manager-webhook   ClusterIP   10.96.194.148  <none>        443/TCP    2d1h

NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/cert-manager  1/1     1           1           2d1h
deployment.apps/cert-manager-cainjector  1/1     1           1           2d1h
deployment.apps/cert-manager-webhook   1/1     1           1           2d1h

NAME          DESIRED   CURRENT   READY   AGE
replicaset.apps/cert-manager-7f545657f4  1         1         1         2d1h
replicaset.apps/cert-manager-cainjector-56564f68dd  1         1         1         2d1h
replicaset.apps/cert-manager-webhook-6855846455  1         1         1         2d1h
>
```

```
kubectl get all -n cert-manager
```

In case it is needed, the best approach for troubleshooting is from description perspective

```
> kubectl describe pkgi cert-manager -n cert-manager
Name: cert-manager
Namespace: cert-manager
Labels: <none>
Annotations: packaging.carvel.dev/package-ClusterRole: cert-manager-cert-manager-cluster-role
             packaging.carvel.dev/package-ClusterRoleBinding: cert-manager-cert-manager-cluster-rolebinding
             packaging.carvel.dev/package-ServiceAccount: cert-manager-cert-manager-sa
             tkg.tanzu.vmware.com/tanzu-package-ClusterRole: cert-manager-cert-manager-cluster-role
             tkg.tanzu.vmware.com/tanzu-package-ClusterRoleBinding: cert-manager-cert-manager-cluster-rolebinding
             tkg.tanzu.vmware.com/tanzu-package-ServiceAccount: cert-manager-cert-manager-sa
API Version: packaging.carvel.dev/v1alpha1
Kind: PackageInstall
Metadata:
  Creation Timestamp: 2024-09-14T08:15:46Z
  Finalizers:
    finalizers.packageinstall.packaging.carvel.dev/delete
  Generation: 1
  Resource Version: 5131
  UID: 72c40e38-e175-46ee-85ef-30af26fle27d
Spec:
  Package Ref:
    Ref Name: cert-manager.tanzu.vmware.com
    Version Selection:
      Constraints: 1.12.2+vmware.2-tkg.2
      Prereleases:
        Service Account Name: cert-manager-cert-manager-sa
Status:
  Conditions:
    Status: True
    Type: ReconcileSucceeded
    Friendly Description: Reconcile succeeded
  Last Attempted Version: 1.12.2+vmware.2-tkg.2
  Observed Generation: 1
  Version: 1.12.2+vmware.2-tkg.2
Events:
  <none>
}
```

```
kubectl describe pkgi cert-manager -n cert-manager
```



You are now ready to continue with the next chapter!

# Set the Image Registry

## 1. Task description and objectives

Your task is to log in to Harbor, create a project, and obtain the certificate for login to the registry.

### ! IMPORTANT:

This exercise is being done with administrator@vsphere.local permissions set over the **If-tdd-namespace**. Make sure you are logged in your **Supervisor**

Your **kubeconfig** context must be set to the **tkg-cluster-c001** cluster!

## Working context

Make sure you are working on the **tkg-cluster-c001** workload cluster you created and managed in previous steps

```
> kubectx ①  
10.80.9.2  
lf-tdd-namespace  
svc-cci-service-domain-c9  
svc-contour-domain-c9  
svc-external-dns-domain-c9  
svc-harbor-domain-c9  
svc-tkg-domain-c9  
svc-velero-domain-c9  
tkg-cluster-c001-admin@tkg-cluster-c001  
> kubectx tkg-cluster-c001-admin@tkg-cluster-c001 ②  
✓ Switched to context "tkg-cluster-c001-admin@tkg-cluster-c001". ←
```

kubectx

kubectx tkg-cluster-c001-admin@tkg-cluster-c001

## 2. Work with Harbor

### 2.1. Access and Prepare Harbor

Access Harbor using its FQDN

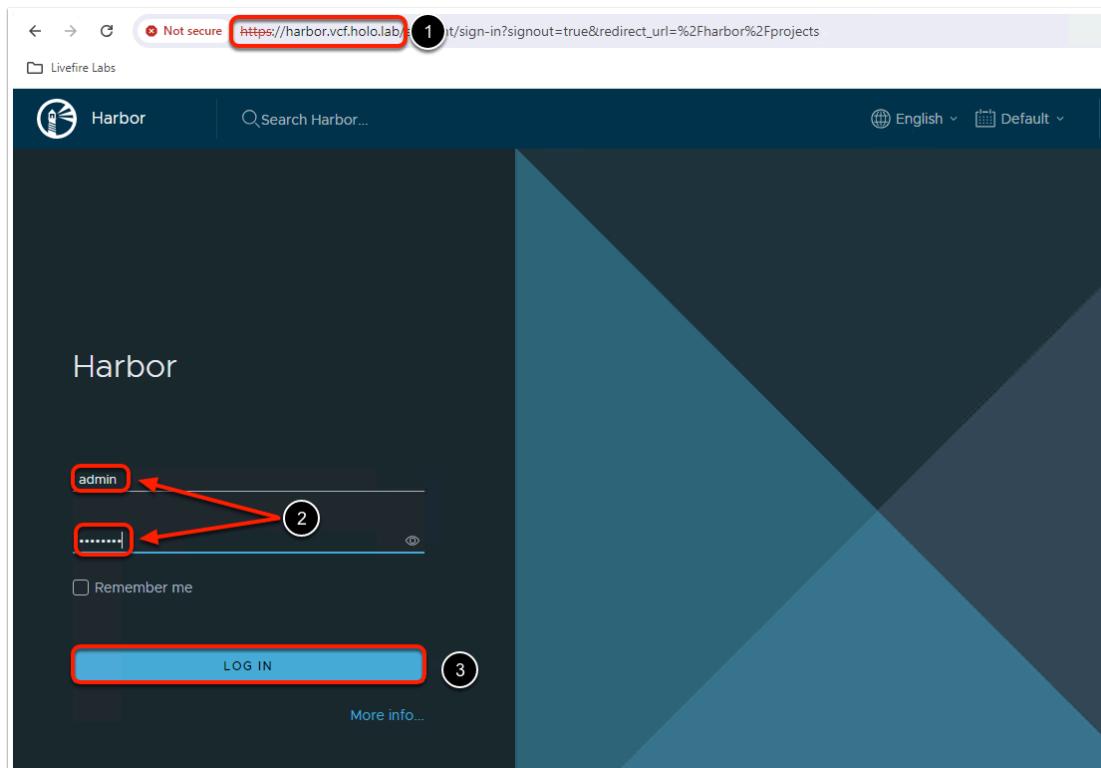
The Harbor FQDN is provided by ExternalDNS, and it is set in the **harbor-data-values.yml** file.

```
>  
> cat .\Downloads\harbor-data-values.yml | grep hostname  
hostname: harbor.vcf.holo.lab  
>
```

You can review this value from **harbor-data-values.yml** file by running a command in Terminal.

```
cat .\Downloads\harbor-data-values.yml | grep hostname
```

Now let's prepare your environment for Harbor



1. In Chrome, Go to your harbor URL: <https://harbor.vcf.holo.lab>
2. Provide the credentials
3. Confirm with LOG IN

Username:

```
admin
```

Password:

```
VMware1!
```

Let's create a Project

The screenshot shows the Harbor Project management interface. On the left, a sidebar menu includes options like Projects, Logs, Administration, Users, Robot Accounts, Registries, Replications, Distributions, Labels, Project Quotas, Interrogation Services, Clean Up, Job Service Dashboard, Configuration, LIGHT, and Harbor API V2.0. The main area is titled "Projects" and displays a summary table with counts for Private, Public, and Total projects and repositories. Below this is a table listing existing projects, with one entry for "library". A red box highlights the "+ NEW PROJECT" button at the top left of the project list area.

Click on NEW PROJECT

The screenshot shows the "New Project" dialog box. It contains fields for "Project Name" (containing "lf-tdd-project"), "Access Level" (set to "Public" with a checked checkbox), "Project quota limits" (set to "-1 GiB"), and "Proxy Cache" (disabled). The "OK" button at the bottom right is highlighted with a red box. Three numbered circles (1, 2, 3) point to the "Project Name" field, the "Access Level" checkbox, and the "OK" button respectively.

Project Name:

lf-tdd-project

Access Level: mark Public Leave the other options unchanged and click **OK**

The screenshot shows the Harbor interface with the 'Projects' section selected. On the left sidebar, under 'Administration', 'Access Level' is set to 'Public'. The main area displays two projects: 'If-tdd-project' and 'library', both listed as 'Public' with 'Project Admin' roles assigned. The 'Access Level' column in the table is highlighted with a red box.

On the Projects page go to your If-tdd-project

The screenshot shows the detailed view of the 'If-tdd-project'. The 'Access Level' is set to 'Public'. The 'Repositories' tab is selected, showing a message: 'We couldn't find any repositories!'. The 'REGISTRY CERTIFICATE' button is highlighted with a red box.

Click on REGISTRY CERTIFICATE

The **ca.cer** file is downloaded in your Downloads folder

Name	Date modified	Type	Size
ca.crt	8/16/2024 11:08 AM	Security Certificate	2 KB
allow-privileged.yaml	8/15/2024 1:33 PM	Yaml Source File	1 KB
tkgs-wkld-c001.yaml	8/15/2024 11:37 AM	Yaml Source File	2 KB
tkgs-wkld-template.yaml	8/15/2024 10:15 AM	Yaml Source File	1 KB
tkc-dcc.yaml	8/15/2024 8:28 AM	Yaml Source File	89 KB
tanzu-cli-windows-amd64.zip	8/14/2024 12:22 PM	Compressed (zipp...)	38,586 KB
vsphere-plugin.zip	8/14/2024 12:14 PM	Compressed (zipp...)	36,875 KB
cci-supervisor-service.yml	8/14/2024 11:07 AM	Yaml Source File	2 KB
harbor-data-values.yml	8/14/2024 10:59 AM	Yaml Source File	4 KB
harbor.yml	8/14/2024 10:58 AM	Yaml Source File	46 KB
external-dns-data-values.yml	8/14/2024 10:54 AM	Yaml Source File	1 KB
external-dns.yml	8/14/2024 10:53 AM	Yaml Source File	42 KB
contour-data-values.yml	8/14/2024 10:40 AM	Yaml Source File	1 KB
contour.yml	8/14/2024 10:40 AM	Yaml Source File	24 KB

Rename this file to more distinguished name:

```
> ls C:\Users\Administrator\Downloads\ca.crt          1
> Directory: C:\Users\Administrator\Downloads
Mode                LastWriteTime        Length Name
----              -----           ----- 
-a---    9/16/2024  9:35 AM           1155 ca.crt

> ren C:\Users\Administrator\Downloads\ca.crt C:\Users\Administrator\Downloads\harbor.vcf.holo.lab.crt      2
>
> ls C:\Users\Administrator\Downloads\harbor.vcf.holo.lab.crt          3
> Directory: C:\Users\Administrator\Downloads
Mode                LastWriteTime        Length Name
----              -----           ----- 
-a---    9/16/2024  9:35 AM           1155 harbor.vcf.holo.lab.crt

>
```

```
ls C:\Users\Administrator\Downloads\ca.crt
```

```
ren C:\Users\Administrator\Downloads\ca.crt C:\Users\Administrator\Downloads\harbor.vcf.holo.lab.crt
```

```
ls C:\Users\Administrator\Downloads\harbor.vcf.holo.lab.crt
```

## 2.2. Use the Harbor Repository

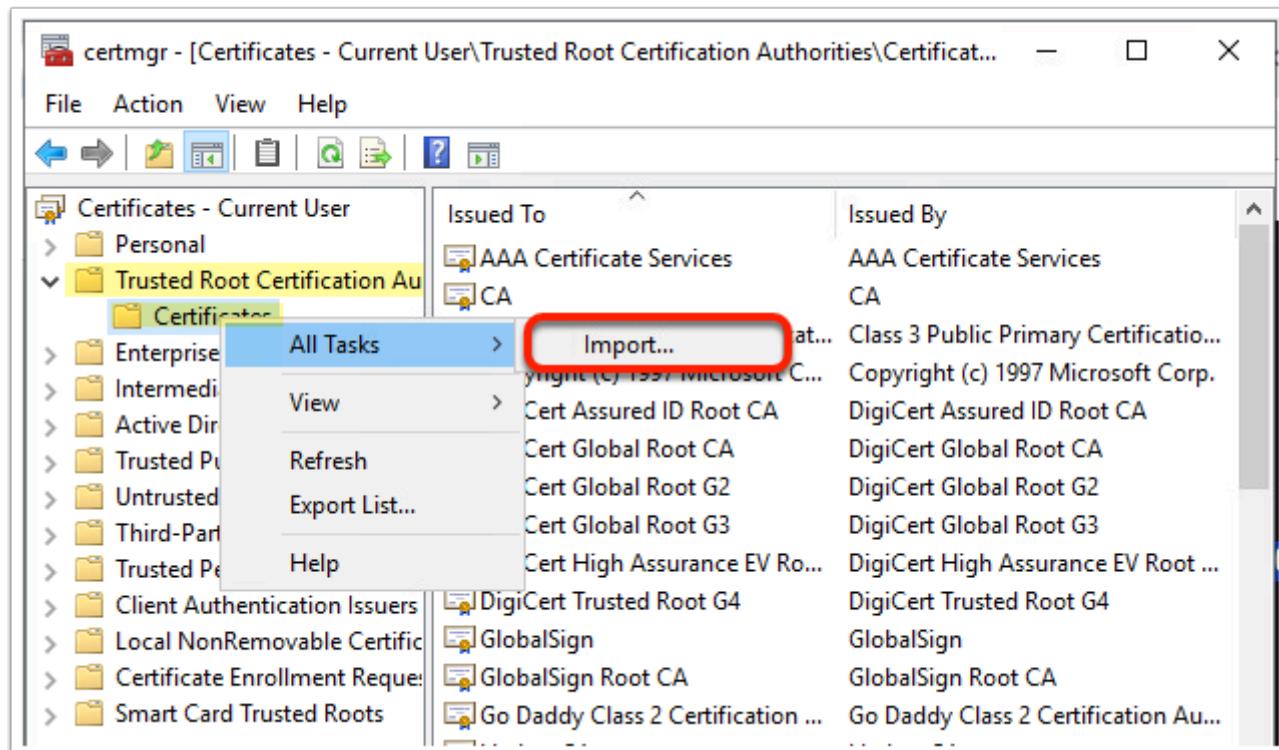
We need to import the **harbor.vcf.holo.lab** certificate file you got in the previous step.

You can do this via the Certificate snap-in for Microsoft Management Console on your Holo-Console machine:

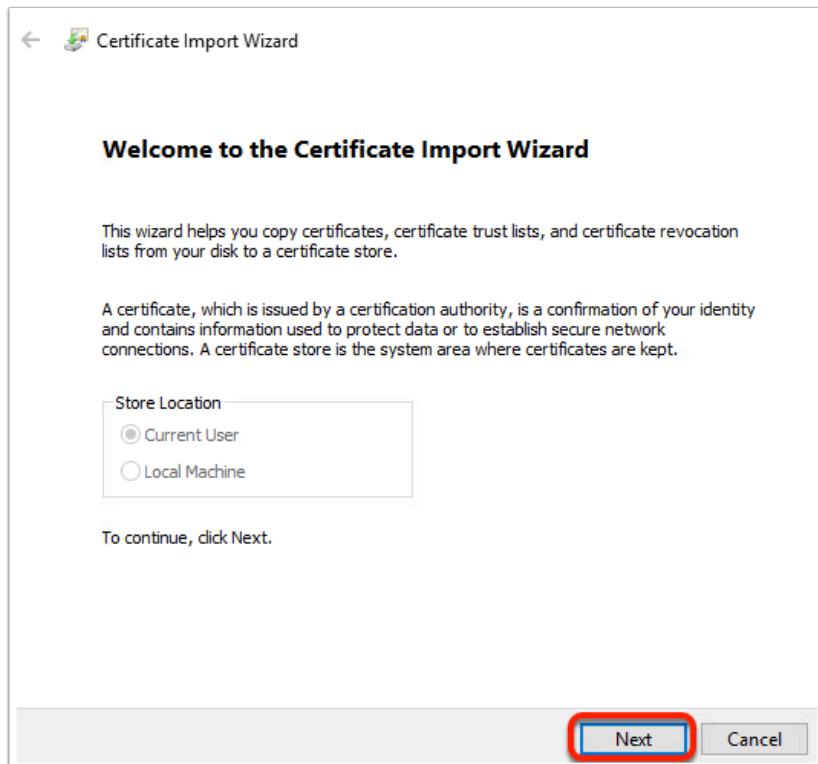


```
certmgr.msc
```

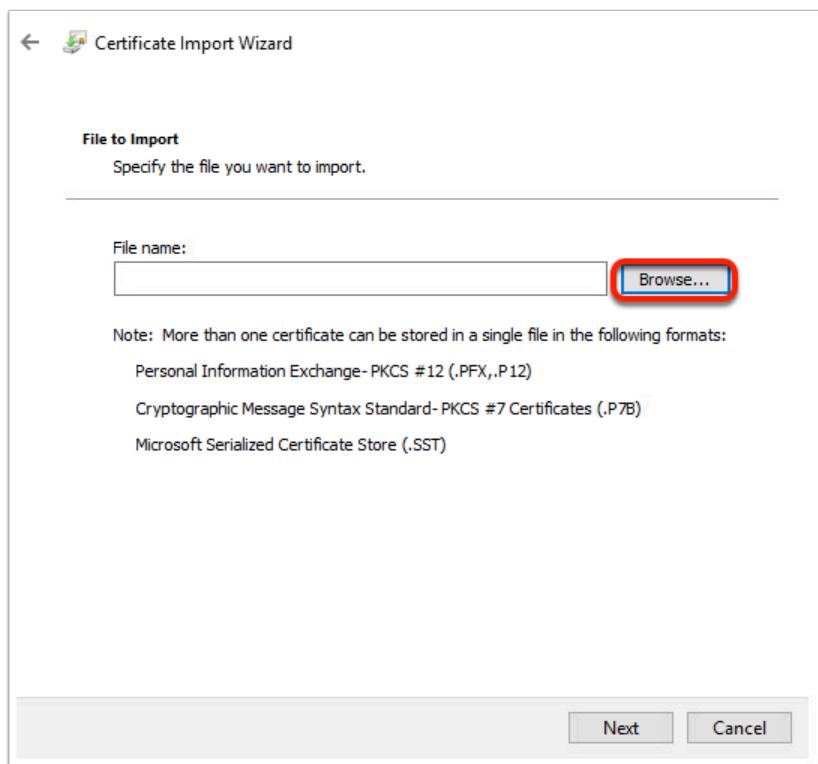
This invokes Microsoft Management Console, where you need to add the needed certificate



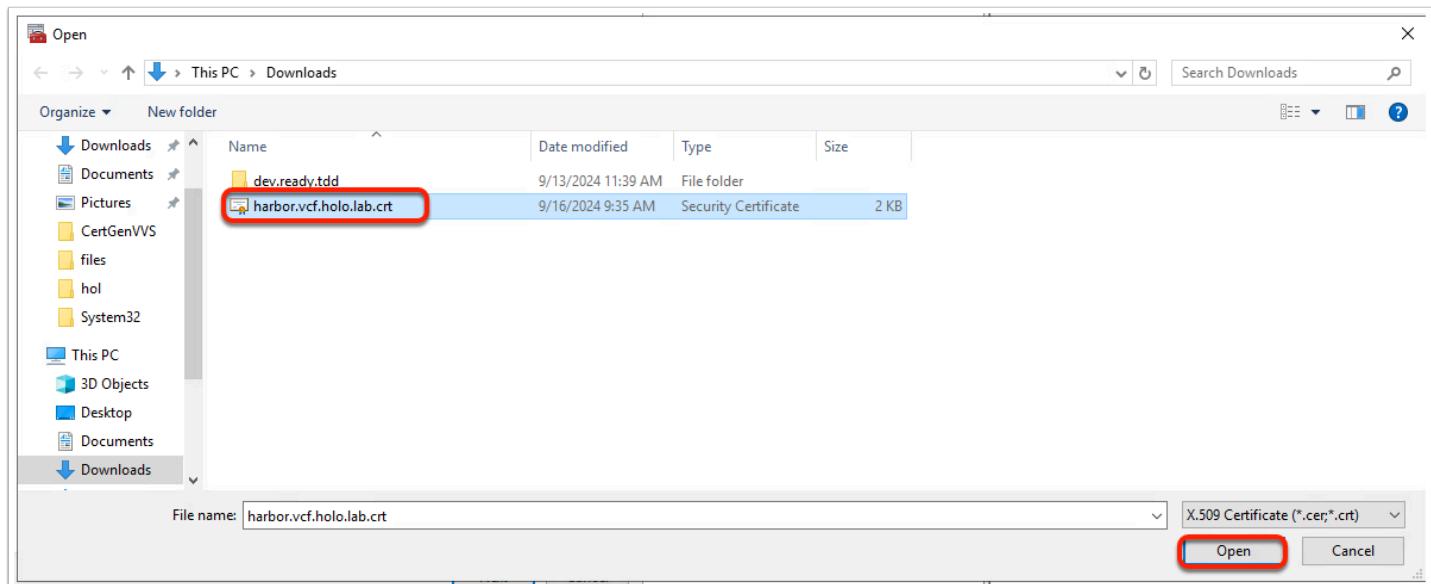
From **Certificates - Current User** -> reveal **Trusted Root Certification Authorities** -> right click on **Certificates** -> select **All Tasks** -> then **Import**



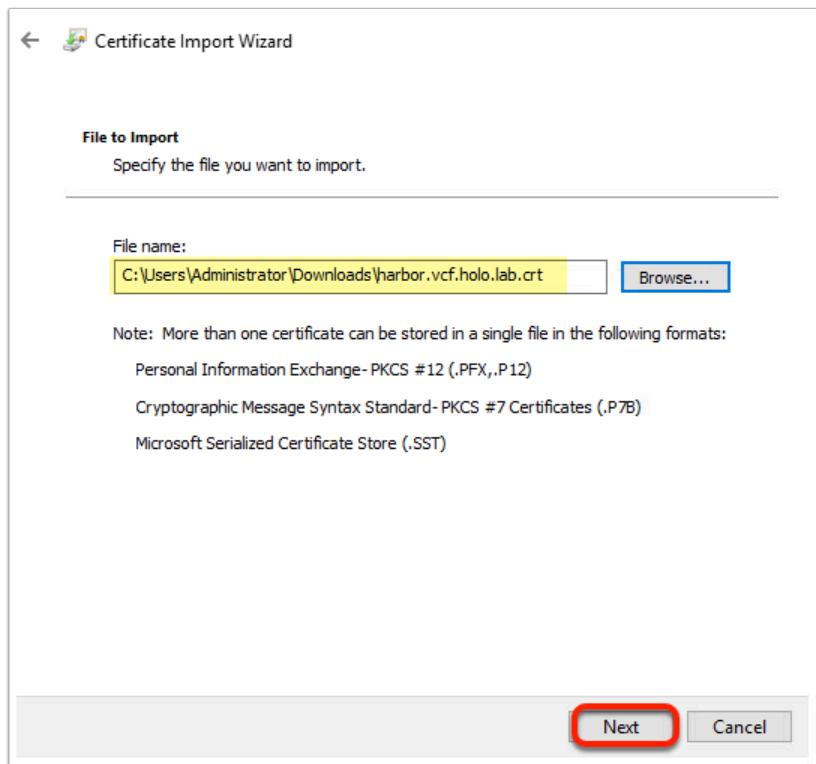
Select **Next**



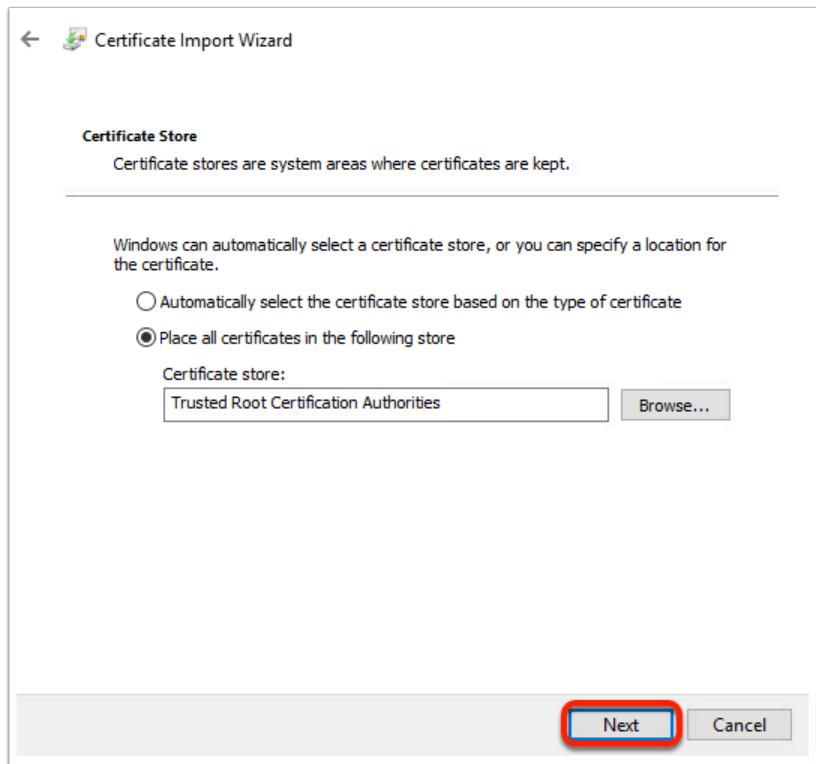
Select **Browse**



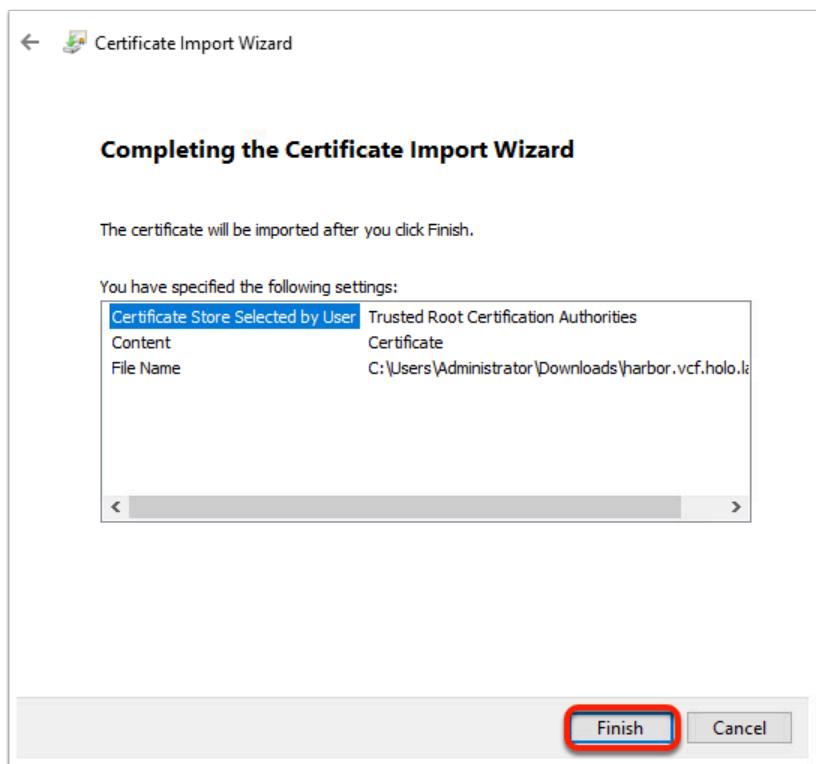
Point your **harbor.vcf.holo.lab** certificate and confirm with **Open**



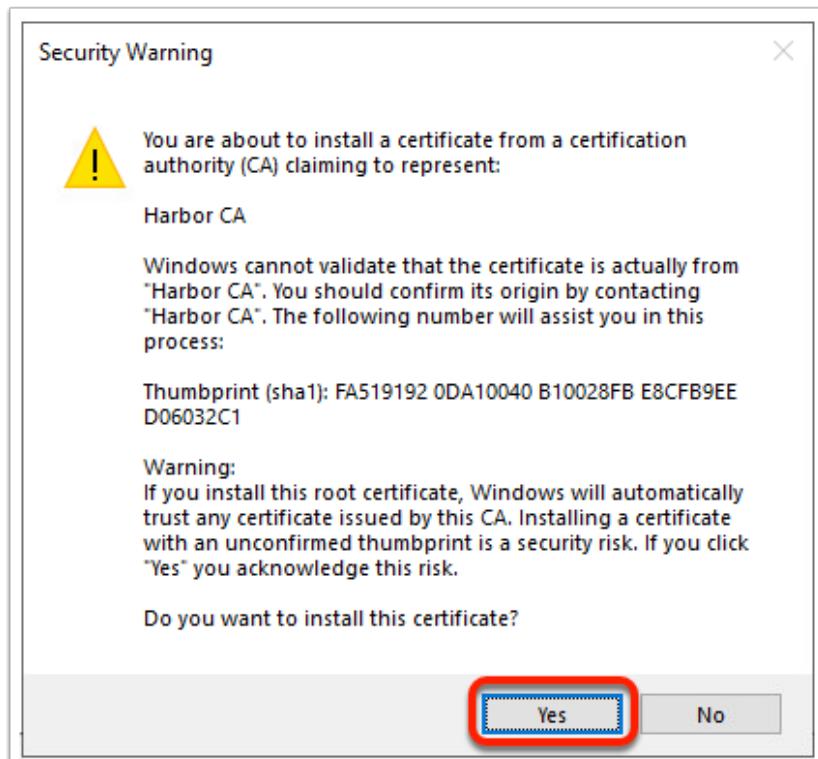
Select **Next**



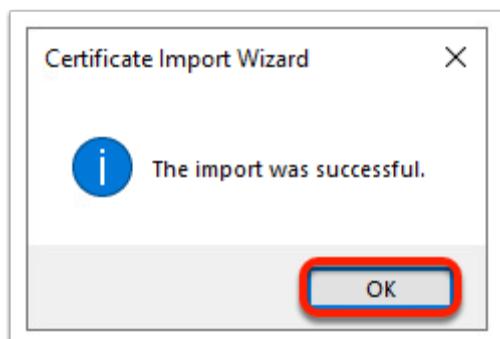
Select **Next**



Click on **Finish**



In the **Security Warning** dialog, select **Yes**



Click **OK**

Issued To	Issued By	Expiration Date	Intended Purposes	Friendly Name	Status
AAA Certificate Services	AAA Certificate Services	12/31/2028	Client Authentication	Sectigo (AAA)	<None>
CA	CA	7/25/2034	<All>	<None>	
Class 3 Public Primary Certificate...	Class 3 Public Primary Certificate...	8/1/2028	Client Authentication	VeriSign Class 3 Pu...	
Copyright (c) 1997 Microsoft C...	Copyright (c) 1997 Microsoft Corp.	12/30/1999	Time Stamping	Microsoft Timesta...	
DigiCert Assured ID Root CA	DigiCert Assured ID Root CA	11/10/2031	Client Authentication	DigiCert	
DigiCert Global Root CA	DigiCert Global Root CA	11/10/2031	Client Authentication	DigiCert	
DigiCert Global Root G2	DigiCert Global Root G2	1/15/2038	Client Authentication	DigiCert Global Roo...	
DigiCert Global Root G3	DigiCert Global Root G3	1/15/2038	Client Authentication	DigiCert Global Roo...	
DigiCert High Assurance EV Ro...	DigiCert High Assurance EV Root...	11/10/2031	Client Authentication	DigiCert	
DigiCert Trusted Root G4	DigiCert Trusted Root G4	1/15/2038	Client Authentication	DigiCert Trusted Ro...	
GlobalSign	GlobalSign	3/18/2029	Client Authentication	GlobalSign Root CA...	
GlobalSign Root CA	GlobalSign Root CA	1/28/2028	Client Authentication	GlobalSign Root CA...	
Go Daddy Class 2 Certification ...	Go Daddy Class 2 Certification Au...	6/29/2034	Client Authentication	Go Daddy Class 2 C...	
<b>Harbor CA</b>	<b>Harbor CA</b>	<b>9/11/2034</b>	<b>Server Authentication</b>	<b>&lt;None&gt;</b>	
ISRG Root X1	ISRG Root X1	6/4/2035	Client Authentication	ISRG Root X1	
Microsoft Authenticode(tm) Ro...	Microsoft Authenticode(tm) Root...	12/31/1999	Secure Email, Code ...	Microsoft Authenti...	
Microsoft ECC Product Root Ce...	Microsoft ECC Product Root Certi...	2/27/2043	<All>	Microsoft ECC Prod...	
Microsoft ECC TS Root Certifica...	Microsoft ECC TS Root Certificate...	2/27/2043	<All>	Microsoft ECC TS R...	
Microsoft Root Authority	Microsoft Root Authority	12/31/2020	<All>	Microsoft Root Aut...	
Microsoft Root Certificate Auth...	Microsoft Root Certificate Authori...	5/9/2021	<All>	Microsoft Root Cert...	
Microsoft Root Certificate Auth...	Microsoft Root Certificate Authori...	6/23/2035	<All>	Microsoft Root Cert...	
Microsoft Root Certificate Auth...	Microsoft Root Certificate Authori...	3/22/2036	<All>	Microsoft Root Cert...	
Microsoft RSA Root Certificate ...	Microsoft RSA Root Certificate Au...	7/18/2049	Client Authentication	Microsoft RSA Root...	

Now you have the harbor cert in place!

As a next step, we need to have Docker CLI in windows desktop / PS Terminal

```

>
> choco install docker-cli
Chocolatey v2.3.0
Installing the following packages:
docker-cli
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading docker-cli 27.3.1... 100%

docker-cli v27.3.1 [Approved]
docker-cli package files install completed. Performing other installation steps.
ShimGen has successfully created a shim for docker.exe
The install of docker-cli was successful.
Deployed to 'C:\ProgramData\chocolatey\lib\docker-cli'

Chocolatey installed 1/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
>

```

```
choco install docker-cli
```

Check it is working:

```
> docker login --help

Usage: docker login [OPTIONS] [SERVER]

Authenticate to a registry.
Defaults to Docker Hub if no server is specified.

Options:
  -p, --password string    Password
  --password-stdin          Take the password from stdin
  -u, --username string    Username
>
```

```
docker login --help
```

Okay, time to log in to our Harbor:

```
>
> docker login harbor.vcf.holo.lab
Username: admin
Password:
WARNING! Your password will be stored unencrypted in C:\Users\Administrator\.docker\config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credential-stores

Login Succeeded
>
```

```
docker login harbor.vcf.holo.lab
```

username:

password:

You are successfully connected to your Harbor.

You are now ready to move to the next section!

# Configure Access to Developers and Deploy a Workload

## Task description and objectives

As a cluster administrator you can grant cluster access to other users, such as developers.

Developers can deploy pods to clusters directly using their user accounts, or indirectly using service accounts.

Your task is to establish access to Developers and Deploy a Workload.

### ! IMPORTANT:

This exercise is being done with administrator@vsphere.local permissions set over the **If-tdd-namespace**. Make sure you are logged in your **Supervisor**

Your **kubeconfig** context must be set to the **tkg-cluster-c001** cluster!

## Working context

Make sure you are working on the **tkg-cluster-c001** workload cluster you created and managed in previous steps

```
> kubectx ①
10.80.0.2
lf-tdd-namespace
svc-cc1-service-domain-c9
svc-contour-domain-c9
svc-external-dns-domain-c9
svc-harbor-domain-c9
svc-tkg-domain-c9
svc-velero-domain-c9
tkg-cluster-c001-admin@tkg-cluster-c001
> kubectx tkg-cluster-c001-admin@tkg-cluster-c001 ②
> Switched to context "tkg-cluster-c001-admin@tkg-cluster-c001". ←
```

kubectx

kubectx tkg-cluster-c001-admin@tkg-cluster-c001

# Configure Access to Developers

For user account authentication, Tanzu Kubernetes clusters support vCenter Single Sign-On users and groups. The user or group can be local to the vCenter Server, or synchronized from a supported directory server.

For service account authentication, you can use service tokens

To grant cluster access to developers:

You have to:

- Define a Role or ClusterRole for the user or group and apply it to the cluster.
- Create a RoleBinding or ClusterRoleBinding for the user or group and apply it to the cluster
- Prepare a Namespace in the cluster where you configure the Developer's access.

## Create a Namespace for the Developer

 Another aspect you need to consider, is **Pod Security Admission**

 **IMPORTANT:** You need to consider Configuration of **PSA** for TKR 1.25 and Later

Tanzu Kubernetes releases v1.25 and later enable the **Pod Security Admission (PSA)** controller. With **PSA** you can uniformly enforce pod security using namespace labels.

The **Pod Security Admission (PSA)** controller is a Kubernetes controller that lets you apply security standards to pods running on TKG clusters. By default, Tanzu Kubernetes releases v1.25 and later enable the Pod Security Admission (PSA) controller. The PSA controller replaces the Pod Security Policy (PSP) controller, which is deprecated and removed.

### PSA Modes

Mode	Description
enforce	Security violations cause the pod to be rejected.
audit	Security violations trigger the addition of an audit annotation to the event recorded in the audit log, but are otherwise allowed
Warn	Security violations trigger a user-facing

Mode	Description
	warning, but are otherwise allowed

Tanzu Kubernetes release v1.25 is a transitional release with PSA configured to warn. Starting with Tanzu Kubernetes release v1.26, **PSA is enforced**. You should plan to migrate pod workloads from PSP to PSA in anticipation of upgrading TKG clusters or modify your PSA in accordance with your workloads.

## PSA Standards

Level	Description
privileged	Unrestricted control, providing the widest possible level of permissions. This security standard allows for known privilege escalations.
baseline	Minimally restrictive control which prevents known privilege escalations. This security standard allows the default (minimally specified) pod configuration
restricted	Heavily restricted control, following pod hardening best practices

### ! IMPORTANT!

By default, TKG clusters provisioned with Tanzu Kubernetes releases v1.26 and later have the **PSA mode enforce** set to **restricted** for **non-system namespaces**.

If a pod violates security, it is rejected. You must configure PSA on the namespace to run pods with less restrictive control.

The TKR of the **tkg-cluster-c001** you deployed is **v1.29**

To create objects such as namespaces, you must be logged in as a user that is assigned the cluster admin role.

```
> kubectl create ns dev-ns
namespace/dev-ns created
>
```

Create a namespace called **dev-ns** in **tkg-cluster-c001**

```
kubectl create ns dev-ns
```

Verify that the namespace was created successfully.

## Create a Namespace for the Developer

```
> kubectl get ns
NAME          STATUS  AGE
cert-manager   Active  2d2h
default        Active  2d2h
dev-ns ←      Active  12s
kube-node-lease Active  2d2h
kube-public    Active  2d2h
kube-system    Active  2d2h
secretgen-controller Active  2d2h
tkg-system     Active  2d2h
velero-vsphere-plugin-backupdriver Active  2d2h
vmware-system-antrea  Active  2d2h
vmware-system-auth   Active  2d2h
vmware-system-cloud-provider Active  2d2h
vmware-system-csi    Active  2d2h
vmware-system-tkg    Active  2d2h
>
```

```
kubectl get ns
```

However, you need to regulate the **PSA** accordingly so that the placeholder in your cluster, has the needed relaxed permissions:

```
> kubectl label --overwrite ns dev-ns pod-security.kubernetes.io/enforce=privileged
namespace/dev-ns labeled
>
```

```
kubectl label --overwrite ns dev-ns pod-security.kubernetes.io/enforce=privileged
```

Once the developer's namespace is ready, you can continue with establishment of role and permissions.

## Create a RBAC for the Developer's Account

To grant access to a vCenter Single Sign-On user or group, the subject in the RoleBinding must contain either of the following values for the `name` parameter

## Supported User and Group Fields

Field	Description
sso:USER-NAME@DOMAIN	For example, a local user name, such as sso: <a href="#">administrator@vsphere.local</a>
sso:GROUP-NAME@DOMAIN	For example, a group name from a directory server integrated with the vCenter Server, such as <a href="#">sso:gg-kube-devs@vcf.holo.lab</a>

Open the file `C:\Users\Administrator\Downloads\dev.ready.tdd\devs-gr-rb-template.yaml` in **VSCode**

```
> ls C:\Users\Administrator\Downloads\dev.ready.tdd\devs-gr-rb-template.yaml
Directory: C:\Users\Administrator\Downloads\dev.ready.tdd
Mode                LastWriteTime        Length Name
----                -----          ----  --
-a---       9/13/2024 11:39 AM         1015  devs-gr-rb-template.yaml

>
> code C:\Users\Administrator\Downloads\dev.ready.tdd\devs-gr-rb-template.yaml
```

```
ls C:\Users\Administrator\Downloads\dev.ready.tdd\devs-gr-rb-template.yaml

code C:\Users\Administrator\Downloads\dev.ready.tdd\devs-gr-rb-template.yaml
```

The following template **RoleBinding** binds the vCenter Single Sign-On SSO integrated AD group named **gg-kube-devs@vcf.holo.lab** to the default ClusterRole named `edit`.

This role permits read/write access to most objects in a namespace, in this case the `dev-ns` namespace..

## Create a RBAC for the Developer's Account

```
C: > Users > Administrator > Downloads > 📁 devs-gr-rb-template.yaml > ...
10 kind: RoleBinding
11 apiVersion: rbac.authorization.k8s.io/v1
12 metadata:
13   name: CHANGE_IT          #rolebind-cluster-group-devs
14   namespace: CHANGE_IT     #dev-ns
15 roleRef:
16   kind: ClusterRole
17   name: CHANGE_IT          #edit
18   apiGroup: rbac.authorization.k8s.io
19 subjects:
20 - kind: User
21   name: CHANGE_IT          #sso:gg-kube-devs
22   apiGroup: rbac.authorization.k8s.io
```

Use the **SAVE AS** in VSC and give the template manifest file a custom name: C:\Users\Administrator\Downloads\devs-gr-rb.yaml

Please note the "CHANGE\_IT" strings, you will need to substitute these and provide actual values

## Create a RBAC for the Developer's Account

```
C: > Users > Administrator > Downloads > 📄 devs-gr-rb.yaml > ...  
9 |  
10 kind: RoleBinding  
11 apiVersion: rbac.authorization.k8s.io/v1  
12 metadata:  
13   name: rolebind-cluster-group-devs  
14   namespace: dev-ns  
15 roleRef:  
16   kind: ClusterRole  
17   name: edit  
18   apiGroup: rbac.authorization.k8s.io  
19 subjects:  
20 - kind: User  
21   name: gg-kube-devs@vcf.holo.lab  
22   apiGroup: rbac.authorization.k8s.io
```

Your configuration file should look exactly as the screenshot above.

Save the file!

Double check you have the manifest in place:

## Create a RBAC for the Developer's Account

```
>  
> ls .\Downloads\devs-gr-rb.yaml  
  
Directory: C:\Users\Administrator\Downloads  
  
Mode                LastWriteTime        Length  Name  
----                -----          ----  --  
-a---       9/16/2024 11:02 AM         944  devs-gr-rb.yaml  
  
>
```

```
ls .\Downloads\devs-gr-rb.yaml
```

You are now ready to implement the role in Terminal:

# Create a RBAC for the Developer's Account

```
> kubectl apply -f .\Downloads\devs-gr-rb.yaml
rolebinding.rbac.authorization.k8s.io/rolebind-cluster-group-devs created
>
```

```
kubectl apply -f .\Downloads\devs-gr-rb.yaml
```

## Test Permissions for the Developer

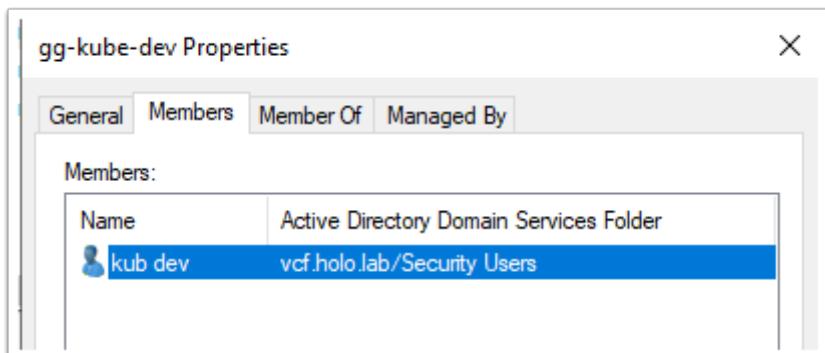
In Terminal, log out as **administrator@vsphere.local**

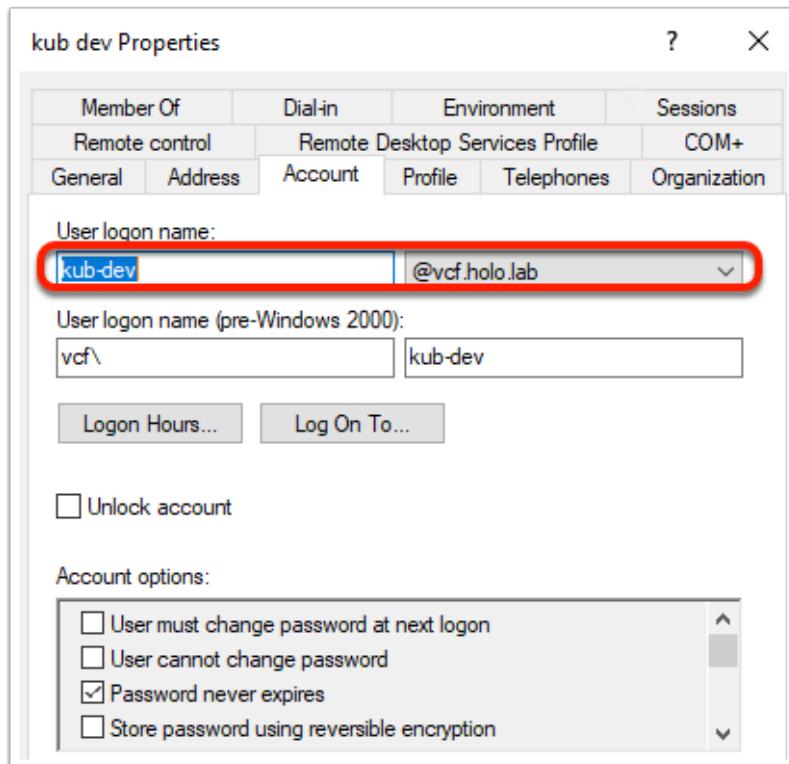
```
> kubectl vsphere logout
Logged out of all vSphere namespaces.
>
```

```
kubectl vsphere logout
```

Log in as an SSO user, member of the AD group you configured rolebinding earlier.

## Test Permissions for the Developer





```
> kubectl vsphere login --vsphere-username kub-dev@vcf.holo.lab --server=10.80.0.2 --tanzu-kubernetes-cluster-name tkg-cluster-c001 --tanzu-kubernetes-cluster-namespace lf-tdd-namespace --insecure-skip-tls-verify

Password:
Logged in successfully.

You have access to the following contexts:
  10.80.0.2
  lf-tdd-namespace
  tkg-cluster-c001 ←

If the context you wish to use is not in this list, you may need to try
logging in again later, or contact your cluster administrator.

To change context, use `kubectl config use-context <workload name>`
>
```

```
kubectl vsphere login --vsphere-username kub-dev@vcf.holo.lab --server=10.80.0.2 --
tanzu-kubernetes-cluster-name tkg-cluster-c001 --tanzu-kubernetes-cluster-namespace lf-
tdd-namespace --insecure-skip-tls-verify
```

username: `kub-dev@vcf.holo.lab`

password: `VMware123!`

Test the permissions of the objects in the cluster

Change working context to the workload TKG cluster

## Test Permissions for the Developer

```
> kubectx tkg-cluster-c001
> Switched to context "tkg-cluster-c001".
```

```
kubectx tkg-cluster-c001
```

Test permissions

## Test Permissions for the Developer

```
> kubectl get ns
NAME          STATUS  AGE
cert-manager   Active  27h
default        Active  28h
dev-ns         Active  49m ←
kube-node-lease Active  28h
kube-public    Active  28h
kube-system    Active  28h
secretgen-controller Active  28h
tito           Active  25h
tkg-system     Active  28h
velero-vsphere-plugin-backupdriver Active  28h
vmware-system-antrea  Active  28h
vmware-system-auth   Active  28h
vmware-system-cloud-provider Active  28h
vmware-system-csi    Active  28h
vmware-system-tkg    Active  28h
```

```
kubectl get ns
```

```
> kubectl get nodes -o wide
NAME                  STATUS  ROLES      AGE   VERSION  INTERNAL-IP  EXTERNAL-IP  OS-IMAGE       KERNEL-VERSION  CONTAINER-RUNTIME
tkg-cluster-c001-qbkjc-k2qc8  Ready   control-plane  2d3h  v1.29.4+vmware.3-fips.1  10.244.0.66  <none>        Ubuntu 22.04.4 LTS  5.15.0-112-generic  containerd://1.6.31
tkg-cluster-c001-tkg-cluster-nodepool-c001-jcvkd-c6ksj-n2vhq  Ready   <none>      2d3h  v1.29.4+vmware.3-fips.1  10.244.0.67  <none>        Ubuntu 22.04.4 LTS  5.15.0-112-generic  containerd://1.6.31

```

```
kubectl get nodes -o wide
```

```
> kubectl get pods -A
NAMESPACE          NAME               READY  STATUS    RESTARTS  AGE
cert-manager       cert-manager-7f545657f4-tnplt  1/1    Running  0          2d2h
cert-manager       cert-manager-cainjector-56564f68dd-stmvn  1/1    Running  0          2d2h
cert-manager       cert-manager-webhook-6855846455-qqfk4  1/1    Running  0          2d2h
kube-system        antrea-agent-2p8hw  2/2    Running  0          2d3h
kube-system        antrea-agent-4zgzs  2/2    Running  0          2d3h
kube-system        antrea-controller-69f8dc4689-58fn5  1/1    Running  0          2d3h
kube-system        coredns-66d8b77bb7-5v4r9  1/1    Running  0          2d3h
kube-system        coredns-66d8b77bb7-qpw8x  1/1    Running  0          2d3h
kube-system        docker-registry-tkg-cluster-c001-qbkjc-k2qc8  1/1    Running  0          2d3h
kube-system        docker-registry-tkg-cluster-c001-tkg-cluster-nodepool-c001-jcvkd-c6ksj-n2vhq  1/1    Running  0          2d3h
kube-system        etcd-tkg-cluster-c001-qbkjc-k2qc8  1/1    Running  0          2d3h
kube-system        kube-apiserver-tkg-cluster-c001-qbkjc-k2qc8  1/1    Running  0          2d3h
kube-system        kube-controller-manager-tkg-cluster-c001-qbkjc-k2qc8  1/1    Running  0          2d3h
kube-system        kube-proxy-2n74j  1/1    Running  0          2d3h
kube-system        kube-proxy-gg8zk  1/1    Running  0          2d3h
kube-system        kube-scheduler-tkg-cluster-c001-qbkjc-k2qc8  1/1    Running  0          2d3h
kube-system        metrics-server-5d7798ccbc-84p8s  1/1    Running  0          2d3h
kube-system        snapshot-controller-86d9b86fb4-4mkkj  1/1    Running  0          2d3h
secretgen-controller secretgen-controller-6f866c4756-6rrz5  1/1    Running  0          2d3h
tkg-system         kapp-controller-56bd5797bd-sjb42  2/2    Running  0          2d3h
tkg-system         tanzu-capabilities-controller-manager-6466f94878-7h757  1/1    Running  0          2d3h
vmware-system-antrea register-placeholder-skjvj  0/1    Completed 0          2d3h
vmware-system-auth  guest-cluster-auth-svc-nztjv  1/1    Running  0          2d3h
vmware-system-cloud-provider guest-cluster-cloud-provider-59758d68c9-q8x2p  1/1    Running  0          2d3h
vmware-system-csi   vsphere-csi-controller-fb55d5746-zxl7c  7/7    Running  0          2d3h
vmware-system-csi   vsphere-csi-node-l5xks  3/3    Running  0          2d3h
vmware-system-csi   vsphere-csi-node-ssc9v  3/3    Running  5 (2d3h ago)  2d3h

```

```
kubectl get pods -A
```

## Deploy a Workload as a Developer

Check you have access to the manifest

```
> ls .\Downloads\dev.ready.tdd\tito-full-lb.yaml  
Directory: C:\Users\Administrator\Downloads\dev.ready.tdd  


| Mode  | LastWriteTime      | Length | Name              |
|-------|--------------------|--------|-------------------|
| -a--- | 9/13/2024 11:39 AM | 3466   | tito-full-lb.yaml |

  
>
```

```
ls .\Downloads\dev.ready.tdd\tito-full-lb.yaml
```

Deploy the TITO app in the corresponding namespace `dev-ns` on the workload TKG cluster

## Deploy a Workload as a Developer

```
>  
> kubectl create -f .\Downloads\dev.ready.tdd\tito-full-lb.yaml -n dev-ns  
secret/mysql-pass created  
deployment.apps/titosql-m6dhr created  
service/tito-sql-service created  
deployment.apps/titofe created  
service/titofe-service created  
>
```

```
kubectl create -f .\Downloads\dev.ready.tdd\tito-full-lb.yaml -n dev-ns
```

In a couple of minutes you should have the app running

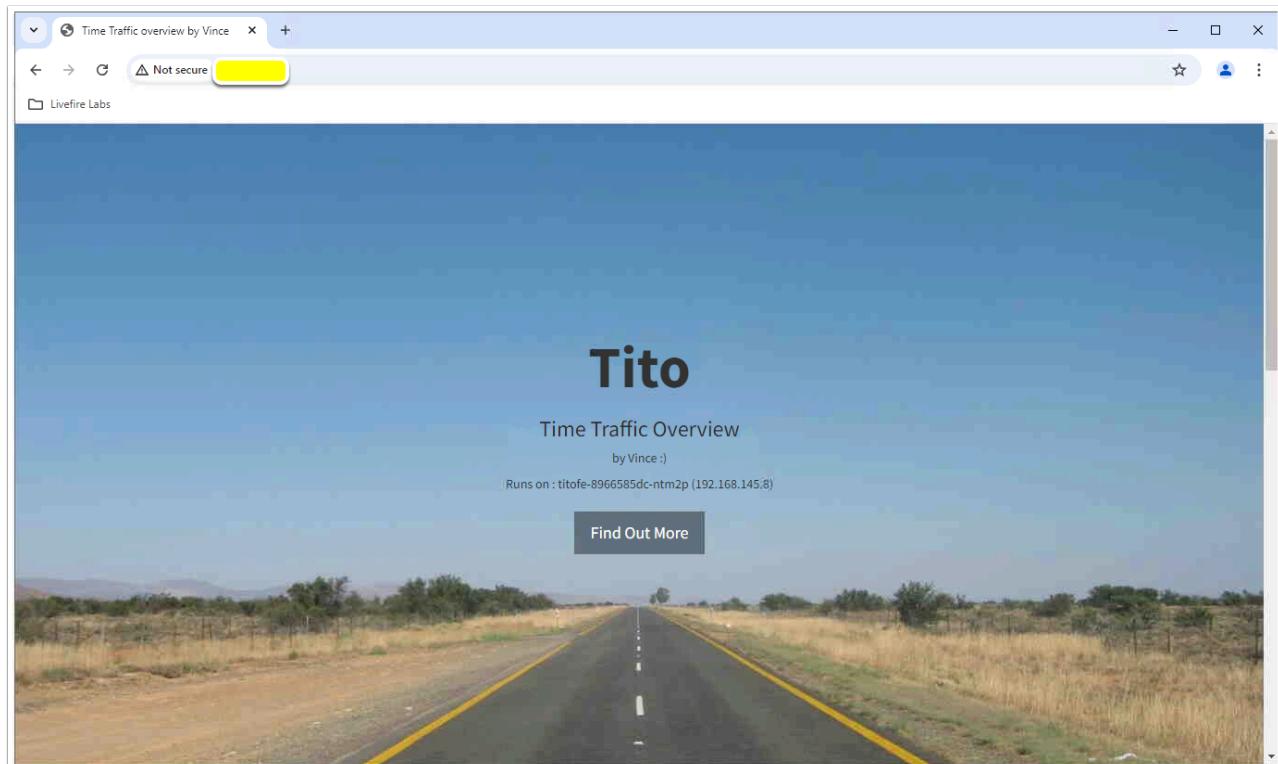
# Deploy a Workload as a Developer

```
> kubectl get all -n dev-ns
NAME                               READY   STATUS    RESTARTS   AGE
pod/titofe-8966585dc-7wp9g        1/1     Running   0          2m28s
pod/titofe-8966585dc-ntm2p        1/1     Running   0          2m28s
pod/titosql-m6dhr-55b4d79fc8-k5vr6 1/1     Running   0          2m28s

NAME                         TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
service/tito-sql-service   ClusterIP  10.98.175.205  <none>           3306/TCP    2m29s
service/titofe-service      LoadBalancer  10.104.134.0   10.80.0.5       80:30779/TCP  2m28s
                                         ↑
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/titofe        2/2     2           2           2m28s
deployment.apps/titosql-m6dhr 1/1     1           1           2m29s

NAME                     DESIRED   CURRENT   READY   AGE
replicaset.apps/titofe-8966585dc 2         2         2       2m28s
replicaset.apps/titosql-m6dhr-55b4d79fc8 1         1         1       2m29s
>
```

```
kubectl get all -n dev-ns
```



You can explore the deployed **Tito** application, via your browser at `http://<EXTERNAL IP FROM titofe-service LoadBanlancer>`

# **Post- Deployment: Troubleshooting**

# Post Deployment Basic Troubleshooting

## 1. Task description and objectives

As part of initial troubleshooting you are tasked to perform a thorough health-check of the environment components, to ensure that no issues exist

## 2. Post Deployment Initial Troubleshooting

You need to:

Collect and inspect Supervisor logs

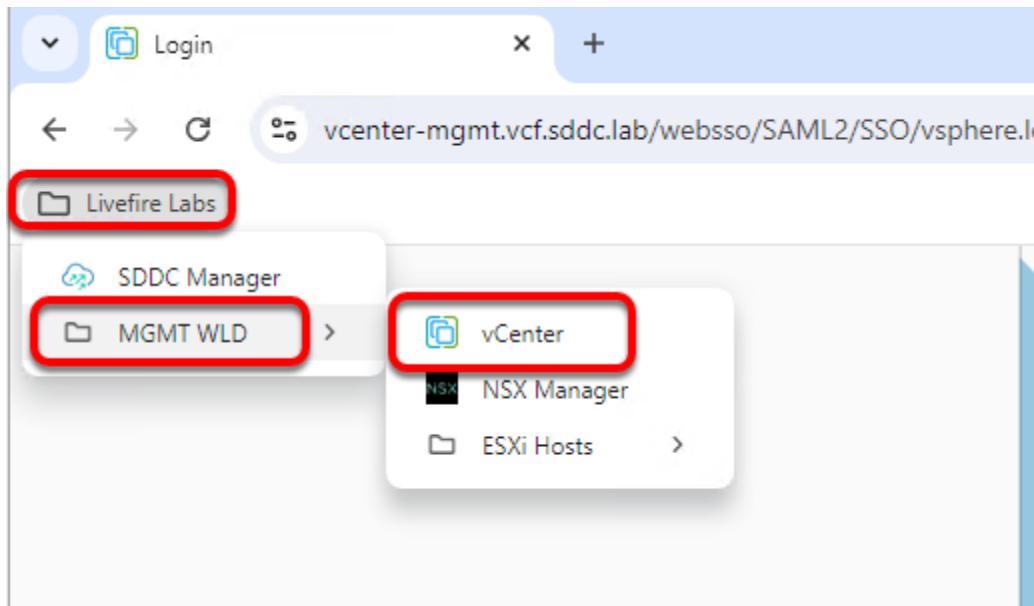
Collect and inspect ClusterAPI logs

Inspect the health of Supervisor's TKG components

Verify SSH access to Supervisor

### 2.1. Get the Supervisor Logs

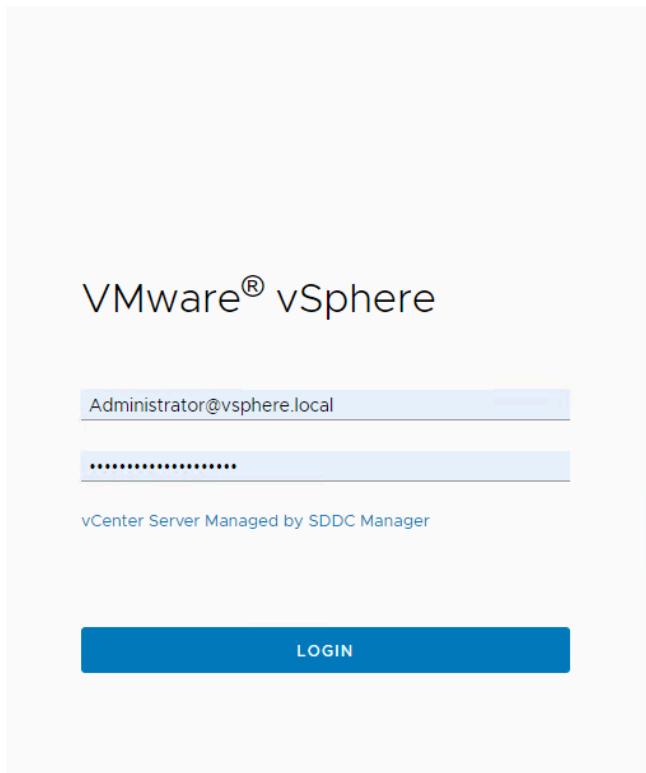
Add text



Open Chrome

Open the Livefire Labs Bookmark Folder

Choose --> **MGMT WLD** --> **vCenter**



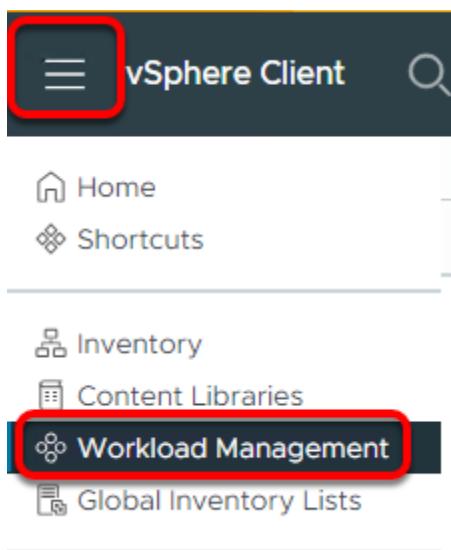
Login:

User:

Administrator@vsphere.local

Password:

VMware123!VMware123!

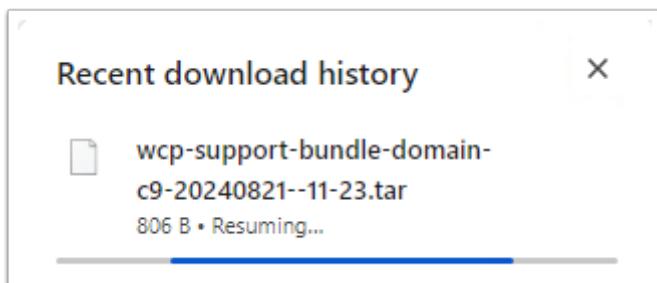


Open the ellipses Menu icon

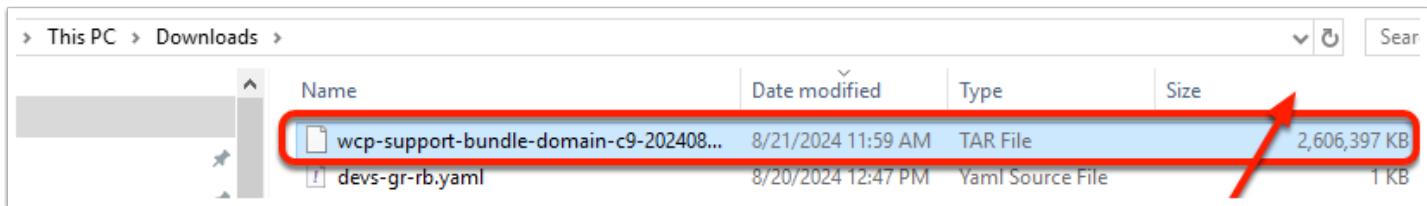
## Choose Workload Management

The screenshot shows the vSphere Client interface with the 'Workload Management' section open. The 'Supervisors' tab is selected (marked with a red box and number 1). A specific supervisor named 'If-tdd-supervisor' is selected in the list (marked with a red box and number 2). The 'EXPORT LOGS' button is highlighted with a red box and number 3.

1. Click the **Supervisors** tab
2. Select the **If-tdd-supervisor**
3. Click **EXPORT LOGS**



The logs are downloaded to the **Downloads** folder



### ! IMPORTANT !!!

This procedure takes significant time, as the logs are being extracted AND compressed at the backend in a TAR file, which comes with a relatively large size!

You don't need to wait for its completion, just move on to the next exercise

## 2.2. Get the ClusterAPI Logs

Use CLI and log in to Supervisor

```
kubectl vsphere login --server=10.80.0.2 -u administrator@vsphere.local --insecure-skip-tls-verify -v 10
```

**password:** VMware123! VMware123!

This command passes the UserID **-u administrator@vsphere.local** as well as passing verbose level 10 to give us more debugging info (**-v 10**) in case we need it.

As the vCenter root certificate is not installed on your windows desktop machine, the **--insecure-skip-tls-verify** flag is required when you log in to the Supervisor

```
> kubeconfig 10.80.0.2 1
✓ Switched to context "10.80.0.2".
>
> tanzu context list 2
NAME          ISACTIVE   TYPE      ENDPOINT  KUBECONFIGPATH  KUBECONTEXT
lf-tdd-supervisor  true    kubernetes  ./kube/config  10.80.0.2
>
> tanzu context use lf-tdd-supervisor 3
[i] Successfully activated context 'lf-tdd-supervisor'
[i] Checking for required plugins for context 'lf-tdd-supervisor'...
[i] All required plugins are already installed and up-to-date
>
```

kubectx 10.80.0.2

tanzu context list

```
tanzu context use lf-tdd-supervisor
```

## Examine the ClusterAPI logs

```
> kubectl get pods -A |grep capw-controller ①
SVC-TKG-DOMAIN-C9          capw-controller-manager-86b544c949-5x7p4
> kubectl logs -n svc-tkg-domain-c9 -c manager capw-controller-manager-86b544c949-5x7p4 ②
I0815 18:32:42.755378       1 main.go:118] setup "msg"="creating manager" "buildnumber"="23829007" "buildtype"="dev" "options"={"LeaderElectionEnabled":true,"defaultQueue":20000000000,"syncPeriod":600000000000,"metricsAddr":"127.0.0.1:8086","TLSMinVersion":"","TLSCipherSuites":"","podNamespace":"svc-tkg-domain-c9","podName":"svc-tkg-domain-c9-capw-controller","maxConcurrentReconciles":3} "version"="1.4.2"
I0815 18:32:42.755559       1 opts.go:70] "msg"="PodName should only be set during testing" "controllerManagerName"="svc-tkg-domain-c9-capw-controller"
I0815 18:32:43.441185       1 network.go:72] svc-tkg-domain-c9-capw-controller "msg"="Pick NSX-T network provider"
I0815 18:32:43.441287       1 main.go:126] setup "msg"="starting manager"
I0815 18:32:43.441846       1 server.go:185] controller-runtime/metrics "msg"="Starting metrics server"
I0815 18:32:43.441865       1 leaderelection.go:250] attempting to acquire leader lease svc-tkg-domain-c9/svc-tkg-domain-c9-capw-controller-runtime...
I0815 18:32:43.442222       1 server.go:224] controller-runtime/metrics "msg"="Serving metrics server" "bindAddress"="127.0.0.1:8086" "secure"=false
I0815 18:33:02.924303       1 leaderelection.go:260] successfully acquired lease svc-tkg-domain-c9/svc-tkg-domain-c9-capw-controller-runtime
I0815 18:33:02.924886       1 recorder.go:104] events "msg"="4200b85deb74d6a8da04a9144e884_5a093ef-4cc4-40f5-8170-02650c9265c6 became leader" "object"={"kind":"Lease","namespace":"svc-tkg-domain-c9","name":"svc-tkg-domain-c9-capw-controller-runtime","uid": "90e7ca84-a197-4cc4-b491-7fc049c6b38e","apiVersion": "coordination.k8s.io/v1","resourceVersion": "1290000"} "reason"="LeaderElection" "type"="Normal"
I0815 18:33:02.925899       1 controller.go:178] "msg"="Starting EventSource" "controller"="WCPMachine" "controllerGroup"="infrastructure.cluster.vmware.com" "controllerKind"="WCPMachine" "source"="kind source: *v1beta1.WCPMachine"
I0815 18:33:02.926225       1 controller.go:178] "msg"="Starting EventSource" "controller"="WCPMachine" "controllerGroup"="infrastructure.cluster.vmware.com" "controllerKind"="WCPMachine" "source"="kind source: *v1alpha1.VirtualMachine"
I0815 18:33:02.928299       1 controller.go:178] "msg"="Starting EventSource" "controller"="WCPMachine" "controllerGroup"="infrastructure.cluster.vmware.com" "controllerKind"="WCPMachine" "source"="kind source: *v1beta1.Machine"
I0815 18:33:02.928327       1 controller.go:186] "msg"="Starting Controller" "controller"="WCPMachine" "controllerGroup"="infrastructure.cluster.vmware.com" "controllerKind"="WCPMachine"
I0815 18:33:02.928176       1 controller.go:178] "msg"="Starting EventSource" "controller"="WCPCluster" "controllerGroup"="infrastructure.cluster.vmware.com" "controllerKind"="WCPCluster" "source"="kind source: *v1beta1.WCPCluster"
I0815 18:33:02.928627       1 controller.go:178] "msg"="Starting EventSource" "controller"="WCPCluster" "controllerGroup"="infrastructure.cluster.vmware.com" "controllerKind"="WCPCluster" "source"="kind source: *v1beta1.WCPMachine"
I0815 18:33:02.928797       1 controller.go:186] "msg"="Starting Controller" "controller"="WCPCluster" "controllerGroup"="infrastructure.cluster.vmware.com" "controllerKind"="WCPCluster"
I0815 18:33:03.094966       1 controller.go:220] "msg"="Starting workers" "controller"="WCPMachine" "controllerGroup"="infrastructure.cluster.vmware.com" "controllerKind"="WCPMachine" "workerCount"=3
I0815 18:33:03.095208       1 controller.go:220] "msg"="Starting workers" "controller"="WCPCluster" "controllerGroup"="infrastructure.cluster.vmware.com" "controllerKind"="WCPCluster" "workerCount"=3
```

1. Get the **name** of the **capw-controller-manager** pod and the **namespace** it is running in
2. Get the logs from it

```
kubectl get pods -A |grep capw-controller
```

```
kubectl logs -n svc-tkg-domain-c9 -c manager capw-controller-manager-86b544c949-5x7p4
```

## 2.3. Get the Health Status for TKG Components on the Supervisor

You can get the Health Status of TKG components simply inspecting if all pods are in Running state

NAME	READY	STATUS	RESTARTS	AGE
kube-system/coredns-84787595bc-l4z8z	1/1	Running	9	7d3h
kube-system/coredns-84787595bc-nznnq	1/1	Running	0	7d3h
kube-system/coredns-84787595bc-t7zbn	1/1	Running	0	7d3h
kube-system/docker-registry-4200b7c56b369021b669a236712a0913	1/1	Running	0	7d3h
kube-system/docker-registry-4200b5deb7f4d6a8daf04a914a4e884	1/1	Running	0	7d3h
kube-system/docker-registry-4200e37786185a120ff8b9d0b4305b	1/1	Running	0	7d3h
kube-system/etc-4200b7c56b369021b669a236712a0913	1/1	Running	0	7d3h
kube-system/etc-4200b85deb74d6a8daf04a914a4e884	1/1	Running	3	7d3h
kube-system/etc-4200e737786185a120ff8b9d0b4305b	1/1	Running	0	7d3h
kube-system/kube-apiserver-4200b7c56b369021b669a236712a0913	1/1	Running	29 (5d18h ago)	7d3h
kube-system/kube-apiserver-4200b85deb7f4d6a8daf04a914a4e884	1/1	Running	25 (5d18h ago)	7d3h
kube-system/kube-apiserver-4200e737786185a120ff8b9d0b4305b	1/1	Running	32 (5d18h ago)	7d3h
kube-system/kube-controller-manager-4200b7c56b369021b669a236712a0913	1/1	Running	8 (5d17h ago)	7d3h
kube-system/kube-controller-manager-4200b85deb74d6a8daf04a914a4e884	1/1	Running	8 (5d18h ago)	7d3h
kube-system/kube-controller-manager-4200e737786185a120ff8b9d0b4305b	1/1	Running	11 (5d18h ago)	7d3h
kube-system/kube-proxy-57ck	1/1	Running	0	7d3h
kube-system/kube-proxy-c7lzg	1/1	Running	0	7d3h
kube-system/kube-proxy-xl2ww	1/1	Running	0	7d3h
kube-system/kube-scheduler-4200b7c56b369021b669a236712a0913	2/2	Running	8 (5d18h ago)	7d3h
kube-system/kube-scheduler-4200b85deb7f4d6a8daf04a914a4e884	2/2	Running	7 (5d18h ago)	7d3h
kube-system/kube-scheduler-4200e737786185a120ff8b9d0b4305b	2/2	Running	10 (5d17h ago)	7d3h
kube-system/kubectl-plugin-vsphere-4200b7c56b369021b669a236712a0913	1/1	Running	3 (5d18h ago)	7d3h
kube-system/kubectl-plugin-vsphere-4200b85deb7f4d6a8daf04a914a4e884	1/1	Running	5 (7d3h ago)	7d3h
kube-system/snapshot-controller-dbbcd8bb7-564xq	1/1	Running	4 (5d18h ago)	7d3h
kube-system/snapshot-controller-dbbcd8bb7-9rg7f	1/1	Running	3 (5d18h ago)	7d3h
kube-system/snapshot-validation-controller-dbbcd8bb7-pmx84	1/1	Running	16	7d3h
kube-system/snapshot-validation-deployment-756c49c776-xb2cx	1/1	Running	10	7d3h
kube-system/snapshot-validation-deployment-756c49c776-zgvbr	1/1	Running	0	7d3h
Kube-system	1/1	Running	0	7d3h

```
kubectl get pods -A | more
```

Pressing the space bar on your keyboard will pass the information to you on portions.  
However, you can simply filter out the pods that are NOT in a Running state:

NAME	READY	STATUS	RESTARTS	AGE
tmc-agent-installer-28737372-5bk9q	0/1	Completed	0	56s

```
kubectl get pods -A | grep -v Running
```

## 2.4. Get the password for SSH connection

Let's check which TKG workload clusters are available in the relevant namespace

NAME	NAMESPACE	STATUS	CONTROLPLANE	WORKERS	KUBERNETES	ROLES	PLAN	TKR
tkg-cluster-c001	lf-tdd-namespace	running	1/1	1/1	v1.29.4+vmware.3-fips.1	<none>	v1.29.4---vmware.3-fips.1-tkg.1	

```
tanzu cluster list -n lf-tdd-namespace
```

Use the terminal to identify the secret that contains the SSH password for the cluster nodes

NAME	TYPE	DATA	AGE
cluster-autoscaler-secret	kubernetes.io/service-account-token	3	6d3h
tkg-cluster-c001-antrea-data-values	Opaque	1	2d3h
tkg-cluster-c001-auth-svc-cert	kubernetes.io/tls	3	2d3h
tkg-cluster-c001-c5g89-ccm-secret	kubernetes.io/service-account-token	3	2d3h
tkg-cluster-c001-c5g89-pvbackupdriver-secret	kubernetes.io/service-account-token	3	2d3h
tkg-cluster-c001-c5g89-pvcsi-secret	kubernetes.io/service-account-token	3	2d3h
tkg-cluster-c001-ca	cluster.x-k8s.io/secret	2	2d3h
tkg-cluster-c001-capabilities-package	clusterbootstrap-secret	1	2d3h
tkg-cluster-c001-encryption	Opaque	1	2d3h
tkg-cluster-c001-etcd	cluster.x-k8s.io/secret	2	2d3h
tkg-cluster-c001-extensions-ca	kubernetes.io/tls	3	2d3h
tkg-cluster-c001-gateway-api-package	clusterbootstrap-secret	0	2d3h
tkg-cluster-c001-guest-cluster-auth-service-data-values	Opaque	1	2d3h
tkg-cluster-c001-kapp-controller-data-values	Opaque	2	2d3h
tkg-cluster-c001-kubeconfig	cluster.x-k8s.io/secret	1	2d3h
tkg-cluster-c001-ls96c-46rzs	cluster.x-k8s.io/secret	2	2d3h
tkg-cluster-c001-metrics-server-package	clusterbootstrap-secret	0	2d3h
tkg-cluster-c001-pinniped-package	clusterbootstrap-secret	1	2d3h
tkg-cluster-c001-proxy	cluster.x-k8s.io/secret	2	2d3h
tkg-cluster-c001-sa	cluster.x-k8s.io/secret	2	2d3h
tkg-cluster-c001-secretgen-controller-package	clusterbootstrap-secret	1	2d3h
tkg-cluster-c001-ssh	kubernetes.io/ssh-auth	1	2d3h
tkg-cluster-c001-ssh-password	Opaque	1	2d3h
tkg-cluster-c001-ssh-password-hashed	Opaque	1	2d3h
tkg-cluster-c001-tkg-cluster-nodepool-c001-qw6r2-5vn2s	cluster.x-k8s.io/secret	2	2d3h
tkg-cluster-c001-vsphere-cpi-data-values	Opaque	1	2d3h
tkg-cluster-c001-vsphere-pv-csi-data-values	Opaque	1	2d3h

```
kubectl get secrets -n lf-tdd-namespace
```

That should be `tkg-cluster-c001-ssh-password`

Now you can retrieve the `ssh-passwordkey` value from the secret:

```
> kubectl get secrets -n lf-tdd-namespace tkg-cluster-c001-ssh-password -o yaml | grep "ssh-passwordkey"
ssh-passwordkey: czNTaS9MWk1pRWdhY1h6ZW5rMFd3UnRTcER1VDBBU2EydGl0Z1gwTjlv0D0=
```

```
kubectl get secrets -n lf-tdd-namespace tkg-cluster-c001-ssh-password -o yaml | grep "ssh-passwordkey"
```

Record the value in VCD and save it like C:\Downloads\ssh-password.txt file

```
>
> code C:\Users\Administrator\Downloads\ssh-password.txt
```

```
code C:\Users\Administrator\Downloads\ssh-password.txt
```

```
C: > Users > Administrator > Downloads > ssh-password.txt  
1 czNTaS9MWk1pRWdhY1h6ZW5rMFd3UnRTcER1VDBBU2EydgLoZ1gwTjlvOD0=
```

Just as an example:

**ssh-passwordkey:**

```
czNTaS9MWk1pRWdhY1h6ZW5rMFd3UnRTcER1VDBBU2EydgLoZ1gwTjlvOD0=
```

Double check you the file is saved:

```
> ls .\Downloads\ssh-password.txt  
  
Directory: C:\Users\Administrator\Downloads  
  
Mode LastWriteTime Length Name  
---- ----- ----- ----  
-a--- 8/21/2024 12:35 PM 60 ssh-password.txt  
  
>
```

```
ls .\Downloads\ssh-password.txt
```

Decode this password in an output file

```
>  
> certutil -decode .\Downloads\ssh-password.txt .\Downloads\ssh-password-decoded.txt  
Input Length = 60  
Output Length = 44  
CertUtil: -decode command completed successfully.  
>
```

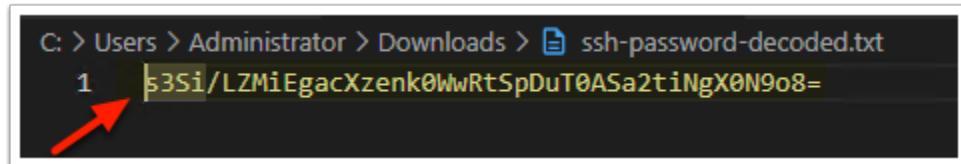
```
certutil -decode .\Downloads\ssh-password.txt .\Downloads\ssh-password-decoded.txt
```

Open the decoded password in VCD

```
>  
> code .\Downloads\ssh-password-decoded.txt  
>
```

```
code .\Downloads\ssh-password-decoded.txt
```

Okay, you have the SSH password now:



C: > Users > Administrator > Downloads > ssh-password-decoded.txt

1 | s3Si/LZMiEgacXzenk0WwRtSpDuT0ASa2tiNgX0N9o8=

A red arrow points to the first character of the password string.

Next, you can get the IP addresses of the TKG workload cluster nodes:



```
> kubectl get vspheremachines -n lf-tdd-namespace -l cluster.x-k8s.io/cluster-name=tkg-cluster-c001 -o wide
NAME                                ZONE      PROVIDERID
tkg-cluster-c001-tkg-cluster-nodepool-c001-8df48-ch6pg   vsphere://42009987-4640-10aa-ab13-3848cab6e576
tkg-cluster-c001-xj2cd-hlrm4        domain-c9  vsphere://42007e9c-862a-81c9-29d4-f5ab4e340889
>
```

IPADDR  
10.244.0.99  
10.244.0.98

A red arrow points to the second IP address.

```
kubectl get vspheremachines -n lf-tdd-namespace -l cluster.x-k8s.io/cluster-name=tkg-
cluster-c001 -o wide
```

You can SSH to the target cluster node as the `vmware-system-user` with the password you decoded in previous steps.

## 2.5. Create a Linux Jump Host VM

To SSH a workload cluster nodes using a password, you can create a Jump Box VM that connects to the workload network and the management of the frontend network for SSH tunneling.

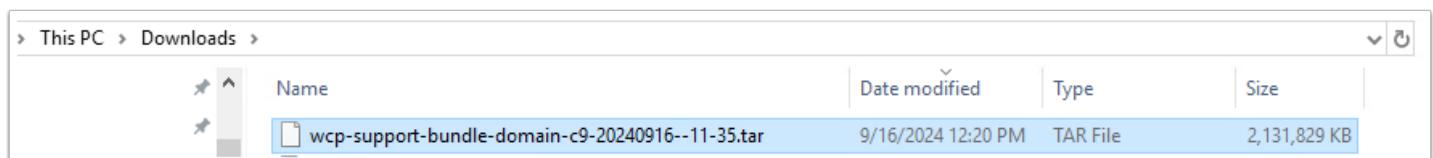
The instructions use PhotonOS which you can download here: <https://github.com/vmware/photon/wiki/Downloading-Photon-OS>.

This exercise won't take place in the current TDD Lab session.

## 2.6. End of the chapter

By now, the supervisor logs from section 2.1 should got downloaded :-)

You can check and see the TAR file



This is the end of your Lab activities!