

VMware Tanzu Kubernetes Grid Integrated Edition

VMware Tanzu Kubernetes Grid Integrated Edition 1.16



You can find the most up-to-date technical documentation on the VMware website at:
<https://docs.vmware.com/>

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

Copyright © 2023 VMware, Inc. All rights reserved. [Copyright and trademark information.](#)

Contents

VMware Tanzu Kubernetes Grid Integrated Edition	70
Overview	70
What Tanzu Kubernetes Grid Integrated Edition Adds to Kubernetes	70
Features	71
Tanzu Kubernetes Grid Integrated Edition Prerequisites	71
Release Notes	72
TKGI v1.16.0	72
Product Snapshot	72
Upgrade Path	73
Breaking Changes	73
Features and Enhancements	74
Telemetry Enhancements	74
vSphere CSI Driver Enhancements	74
Supports the Ubuntu Jammy Stemcell	75
Compatible with Ops Manager v3.0	75
Supports Migrating TKGI to NSX Policy API	75
Supports the Velero vSphere Plugin	75
vSphere CSI Supports Multiple Data Centers	75
Additional Features	76
Resolved Issues	76
Deprecations	77
Known Issues	78
TKGI MC Unable to Manage TKGI after Restoring the TKGI Control Plane from Backup	78
Kubernetes Pods on NSX-T Become Stuck in a Creating State	78
Error: Could Not Execute “Apply-Changes” in Azure Environment	78
VMware vRealize Operations Does Not Support Windows Worker-Based Kubernetes Clusters	79
TKGI Wavefront Requires Manual Installation for Windows Workers	79
Pinging Windows Worker Kubernetes Clusters Does Not Work	80
Velero Does Not Support Backing Up Stateful Windows Workloads	80
Tanzu Mission Control Integration Not Supported on GCP	80
TMC Data Protection Feature Requires Privileged TKGI Containers	80
Windows Worker Kubernetes Clusters with Group Managed Service Account Do Not Support Compute Profiles	80

Windows Worker Kubernetes Clusters on Flannel Do Not Support Compute Profiles	80
TKGI CLI Does Not Prevent Reducing the Control Plane Node Count	80
Windows Cluster Nodes Not Deleted After VM Deleted	81
502 Bad Gateway After OIDC Login	81
Difficulty Changing Proxy for Windows Workers	81
Character Limitations in HTTP Proxy Password	81
Error After Modifying Your Harbor Storage Configuration	82
Ingress Controller Statefulset Fails to Start After Resizing Worker Nodes	82
Azure Default Security Group Is Not Automatically Assigned to Cluster VMs	82
One Plan ID Longer than Other Plan IDs	83
Database Cluster Stops After a Database Instance is Stopped	83
Velero Back Up Fails for vSphere PVs Attached to Clusters on Kubernetes v1.20 and Later	83
Creating Two Windows Clusters at the Same Time Fails	84
Deleted Clusters are Listed in Cluster Lists	84
BOSH Director Logs the Error ‘Duplicate vm extension name’	84
The TKGI API FQDN Must Not Include Trailing Whitespace	85
TMC Cluster Data Protection Backup Fails After Upgrading TKGI	85
TMC Cluster Data Protection Restore Fails When Using Antrea CNI	86
TKGI Does Not Support CVDS / NVDS Mixed Environments	86
Occasionally update-cluster Does Not Complete for Windows Workers	86
Harbor Private Projects Are Inaccessible after Upgrading to TKGI v1.13.0	87
Deployments Fail on TKGI Windows Worker-based Kubernetes Clusters after the January 2022 Microsoft Windows Security Patch	87
TKGI Reallocates Network Profile-Allocated FIP Pool Addresses	88
TKGI Clusters Fail after NSX Upgrade If They Use NSGroup Policy API Resources	88
Kubernetes API Server and etcd Daemon Occasionally Fail to Start During BBR Restore	89
‘Input not an X.509 certificate’ When Applying Change on the TKGI Tile	89
Interoperability with VMware Aria Operations Management Pack for Kubernetes Is Unavailable	90
Interoperability with VMware Cloud Foundation Is Unavailable	90
Interoperability with Tanzu Mission Control is Unavailable	90
Limitations on Using the VMware vSphere CSI Driver	90
Limitations on Using a Public Cloud CSI Driver	90
The ‘kube-state-metrics’ ClusterRole Is Deleted during Cluster Upgrade	91
TKGI Management Console v1.16.0	91
Product Snapshot	91
Upgrade Path	92

Breaking Changes	92
Features and Resolved Issues	92
Telemetry Enhancements	92
Deprecations	92
Known Issues	92
vRealize Log Insight Integration Does Not Support HTTPS Connections	92
vSphere HA causes Management Console ovfenv Data Corruption	93
Base64 encoded file arguments are not decoded in Kubernetes profiles	93
Network profiles not immediately selectable	94
Real-Time IP information not displayed for network profiles	94
Error After Modifying Your Harbor Storage Configuration	94
Windows Stemcells Must be Re-Imported After Upgrading Ops Manager	94
Your New Clusters Are Not Shown In Tanzu Mission Control	95
Tanzu Kubernetes Grid Integrated Edition Concepts	96
Tanzu Kubernetes Grid Integrated Edition Architecture	96
Tanzu Kubernetes Grid Integrated Edition Overview	96
TKGI Control Plane Overview	97
TKGI API VM Group	98
UAA	98
TKGI API	98
TKGI Broker	99
TKGI Database VM Cluster	99
High Availability Modes	99
TKGI Control Plane High Availability Mode (Beta)	99
Windows Worker-Based Kubernetes Cluster High Availability	100
About Tanzu Kubernetes Grid Integrated Edition Upgrades	101
Overview	102
Deciding Between Full and Two-Phase Upgrade	102
What Happens During Full TKGI and TKGI Control Plane Upgrades	103
Full TKGI Upgrades	103
TKGI Control Plane Upgrades	104
Control Plane Outages	104
Canary Instances	104
What Happens During Cluster Upgrades	104
Control Plane Nodes Outage	105
Worker Nodes Outage	105
About Switching from the Flannel CNI to the Antrea CNI	105

Switch from the Flannel CNI to Antrea	106
TKGI API Authentication	106
Authentication of TKGI API Requests	106
Routing to the TKGI API VM	107
Load Balancers in Tanzu Kubernetes Grid Integrated Edition	107
Load Balancers in Tanzu Kubernetes Grid Integrated Edition Deployments without NSX-T	107
About the TKGI API Load Balancer	109
About Kubernetes Cluster Load Balancers	109
About Workload Load Balancers	109
Deploy Your Workload Load Balancer with an Ingress Controller	110
Load Balancers in Tanzu Kubernetes Grid Integrated Edition Deployments on vSphere with NSX-T	110
Resizing Load Balancers	111
VM Sizing for Tanzu Kubernetes Grid Integrated Edition Clusters	111
Overview	112
Control Plane Node VM Size	112
Worker Node VM Number and Size	113
Example Worker Node Requirement Calculation	113
Customize Control Plane and Worker Node VM Size and Type	114
VMware CEIP	114
Overview	114
Configure CEIP	115
Proxy Communication	115
System Components	115
Data Dictionary	116
Sink Architecture in Tanzu Kubernetes Grid Integrated Edition	116
Overview	116
Sink Types	116
Sink Architecture	117
Log Sink Architecture	118
Metric Sink Architecture	118
Containerd Container Runtime	119
Overview	119
Benefits of Containerd	120

Installing Tanzu Kubernetes Grid Integrated Edition	121
Tanzu Kubernetes Grid Integrated Edition Management Console (vSphere Only)	121
Tanzu Kubernetes Grid Integrated Edition on Ops Manager	121
Installing Tanzu Kubernetes Grid Integrated Edition on vSphere	121
Overview	121
Install Tanzu Kubernetes Grid Integrated Edition on vSphere with the Management Console	122
Overview	122
Decide When to Use TKGI Management Console	122
When Should I Use Tanzu Kubernetes Grid Integrated Edition Management Console?	122
When Should I Not Use Tanzu Kubernetes Grid Integrated Edition Management Console?	123
Use the TKGI Management Console After Installing TKGI	123
Prerequisites for Tanzu Kubernetes Grid Integrated Edition Management Console Deployment	123
Network Configurations	124
Virtual Infrastructure Prerequisites	124
Firewall Ports and Protocols Requirements for Tanzu Kubernetes Grid Integrated Edition Management Console	125
Prerequisites for a Bring Your Own Topology Deployment to VMware NSX	126
Overview	126
General Requirements	127
NSX-T Data Center Configuration Requirements	127
Proof-of-Concept Deployments	128
Prerequisites for an Automated NAT Deployment to VMware NSX	128
General Requirements	128
Proof-of-Concept Deployments	129
Prerequisites for vSphere (Antrea and Flannel Networking)	129
Deploy the Tanzu Kubernetes Grid Integrated Edition Management Console	129
Prerequisites	130
Step 1: Deploy the OVA Template	130
Step 2: Log In to Tanzu Kubernetes Grid Integrated Edition Management Console	132
Next Steps	132

Deploy Tanzu Kubernetes Grid Integrated Edition from the Management Console	133
Additional Ops Manager Configurations	133
Deploy Tanzu Kubernetes Grid Integrated Edition by Using the Configuration Wizard	134
Prerequisites	135
Step 0: Launch the Configuration Wizard	135
Step 1: Connect to vCenter Server	136
Step 2: Configure Networking	137
Configure an Automated NAT Deployment to NSX-T Data Center	137
Configure a Bring Your Own Topology Deployment to NSX-T Data Center	140
Configure an Antrea or Flannel Network	142
Step 3: Configure Identity Management	144
Use a Local Database	144
Use an External LDAP Server	145
Use a SAML Identity Provider	146
Optionally Configure UAA and Custom Certificates	148
Step 4: Configure Availability Zones	149
Step 5: Configure Resources and Storage	151
Step 6: Configure Plans	153
Step 7: Configure Integrations	156
Configure a Connection to VMware Tanzu Mission Control	156
Configure a Connection to Wavefront	157
Configure a Connection to VMware vRealize Operations Management Pack for Container Monitoring	158
Configure a Connection to VMware vRealize Log Insight	158
Configure a Connection to Syslog	159
Step 8: Configure Harbor	160
Step 9: Configure CEIP	164
Step 10: Generate Configuration File and Deploy Tanzu Kubernetes Grid Integrated Edition	165
Next Steps	168
Deploy Tanzu Kubernetes Grid Integrated Edition by Importing a YAML Configuration File	168
YAML Files and Passwords	169
Prerequisites	169
Import a YAML Configuration File	170
Networking Options in the YAML File	172
Next Steps	173

Enable Plans with Windows Worker Nodes	173
Prerequisites	174
Step 1: Create a Windows Server Stemcell	174
Step 2: Install the Windows Server Stemcell in Operations Manager	174
Step 3: Reconfigure Your Tanzu Kubernetes Grid Integrated Edition Deployment to Use the Stemcell	175
Install Tanzu Kubernetes Grid Integrated Edition on vSphere with VMware NSX Using Ops Manager	175
Step 1: Prepare to Install Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T	175
Step 2: Install and Configure NSX-T Data Center for Tanzu Kubernetes Grid Integrated Edition	175
Step 3: Create the Management Plane for Tanzu Kubernetes Grid Integrated Edition	176
Step 4: Create the Compute Plane for Tanzu Kubernetes Grid Integrated Edition	176
Step 5: Deploy Ops Manager for Tanzu Kubernetes Grid Integrated Edition with NSX-T	176
Step 6: Generate the NSX-T Management Cluster Root CA Certificate and Key	177
Step 7: Configure BOSH Director for vSphere with NSX-T	177
Step 8: Generate and Register the NSX-T Management Cluster Super User Principal Identity Certificate and Key	177
Step 9: Install Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T	177
Step 10: Install VMware Harbor Registry for Tanzu Kubernetes Grid Integrated Edition	177
Step 11: Install the TKGI and Kubectl CLIs	177
Step 12: Create Admin Users for Tanzu Kubernetes Grid Integrated Edition	178
Step 13: Verify the Installation of Tanzu Kubernetes Grid Integrated Edition	178
Step 14: Perform Desired Post-Installation Configurations	178
Step 15: Create Network Profiles to Customize Cluster Deployments	178
Preparing to Install Tanzu Kubernetes Grid Integrated Edition on vSphere with VMware NSX	178
Preparing to Install Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T	178
vSphere with NSX-T Version Requirements	179
vSphere Version Requirements	179
NSX-T Integration Component Version Requirements	179
Hardware Requirements for Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T	179
vSphere Clusters for Tanzu Kubernetes Grid Integrated Edition	179
Tanzu Kubernetes Grid Integrated Edition Management Cluster	179

Tanzu Kubernetes Grid Integrated Edition Edge Cluster	180
Tanzu Kubernetes Grid Integrated Edition Compute Cluster	180
Tanzu Kubernetes Grid Integrated Edition Management Plane Placement Considerations	180
Configuration Requirements for vSphere Clusters for Tanzu Kubernetes Grid Integrated Edition	180
RPD for Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T	181
RPD for Tanzu Kubernetes Grid Integrated Edition with vSAN	181
Management/Edge Cluster	181
Compute Clusters	182
Storage (vSAN)	182
Future Growth	182
RPD for Tanzu Kubernetes Grid Integrated Edition without vSAN	182
Management/Edge Cluster	183
Compute Clusters	183
Storage (non-vSAN)	183
Future Growth	184
MPD for Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T	184
MPD Topology	184
MPD Configuration Requirements	185
MPD Considerations	185
VM Inventory and Sizes	186
Management Plane VMs and Sizes	186
Storage Requirements for Large Numbers of Pods	186
NSX-T Edge Node VMs and Sizes	187
Kubernetes Cluster Nodes VMs and Sizes	187
Hardware Requirements	187
RPD Hardware Requirements	187
MPD Hardware Requirements	188
Adding Hardware Capacity	188
Firewall Ports and Protocols Requirements for vSphere with NSX-T	189
Overview	189
TKGI Ports and Protocols	190
TKGI Users Ports and Protocols	190
TKGI Core Ports and Protocols	191
VMware Ports and Protocols	194
VMware Virtual Infrastructure Ports and Protocols	195
VMware Optional Integration Ports and Protocols	197

Creating VMware NSX Objects for Tanzu Kubernetes Grid Integrated Edition	198
Create the Nodes IP Block	199
Create the Pods IP Block	200
Create Floating IP Pool	201
Next Step	203
NSX-T Deployment Topologies for Tanzu Kubernetes Grid Integrated Edition	203
NAT Topology	203
No-NAT Topology	204
No-NAT with Virtual Switch (VSS/VDS) Topology	204
No-NAT with Logical Switch (NSX-T) Topology	205
Hybrid Topology	206
vSAN Stretched Cluster Topologies	207
vSphere with VMware NSX Cluster Objects	207
vSphere Virtual Machines	208
NSX-T Logical Switches	208
NSX-T Tier-1 Logical Routers	208
NSX-T Load Balancers	208
NSX-T DDI/IPAM	209
NSX-T Tier-0 Logical Routers	209
NSX-T Distributed Firewall (DFW) Rules	209
Network Planning for Installing Tanzu Kubernetes Grid Integrated Edition with VMware NSX	210
Overview	210
Prerequisites	210
Understand Component Interactions	211
Plan Deployment Topology	211
Plan Network CIDRs	211
Plan IP Blocks	212
Pods IP Block	213
Nodes IP Block	214
Reserved IP Blocks	215
Gather Other Required IP Addresses	216
Considerations for Using the VMware NSX Policy API with TKG	216
NSX Policy API Support	216
NSX Versions	217

NSX Deployment Topologies	217
NSX Installation	217
NSX Objects for Kubernetes Clusters	217
TKGI Configuration	217
Management Console	217
Network Profile	218
 Migrating the NSX Management Plane API to NSX Policy API - Overview	218
Overview	218
NSX Management Plane API to NSX Policy API Migration Features	219
TKGI MP2P Migration Configurations	219
Supported Configurations	219
Unsupported Configurations	220
TKGI MP2P Migration Operational Limitations	220
Before MP2P Migration	220
During MP2P Migration	220
DFW Migration	221
During Cluster Promotion to NSX Policy API	221
 Migrating the NSX Management Plane API to NSX Policy API	222
Overview	222
Prerequisites	224
Prepare for MP2P Migration	225
Enable Migration	225
Migrate DFW Top Firewall Rules	226
Activate NSX Policy API in TKGI	227
Configure the Environment for NSX Policy API	228
Migrate TKGI Clusters from the NSX Management Plane API to NSX Policy API	228
Migrate a TKGI Cluster to NSX Policy API	228
Migrate DFW Bottom Firewall Rules	229
Post-Migration Cleanup	230
 Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with VMware NSX	231
Install NSX-T on vSphere	231
 Prerequisites for Installing and Configuring NSX-T Data Center v3 for TKGI	232
Prerequisites for Installing NSX-T Data Center	232
 Installing and Configuring VMware NSX Managers	232

Prerequisites	232
Deploy NSX-T Manager 1	232
Add vCenter as the Compute Manager	240
Deploy NSX-T Manager 2	243
Deploy NSX-T Manager 3	247
Configure the NSX-T Management VIP	248
Add the NSX-T Manager License	251
Enable the NSX-T Manager Interface (if necessary)	251
 Generate and Register the NSX-T Management TLS Certificate and Private Key	253
Prerequisites	253
Generate and Register the NSX-T Management TLS Certificate and Private Key	253
Generate the SSL Certificate and Private Key	254
Import the SSL Certificate and Private Key to the NSX-T Management Console	255
Register the SSL Certificate and Private Key with the NSX-T API Server	256
 Create an IP Pool for VTEP	258
Create an IP Pool for VTEP	258
 Configuring NSX-T Data Center v3 Transport Zones and Edge Node Switches for Tanzu Kubernetes Grid Integrated Edition	259
Prerequisites	259
Overview of Transport Zones for NSX-T	259
Configure Your NSX Transport Zones for TKGI	260
Option 1: Use the Default Transport Zones with a Single N-VDS Switch	260
Option 2: Create Custom Transport Zones and Use the NSX API to Get the Host Switch Names	261
Option 3: Use the NSX API to Create Custom Transport Zones and NSX Switches	263
TKGI NSX Edge Switch and Transport Zone Host Switch Name Requirements	265
 Configure vSphere Networking for ESXi Hosts	265
Prerequisites	265
Configure vSphere Networking for ESXi Hosts	265
Create vSwitch Port-Groups for Edge Nodes	266
Set vSwitch0 with MTU at 9000	270
Next Step	271
 Install and Configure the NSX-T Edge Nodes	271
Prerequisites	271
Deploy NSX-T Edge Nodes	271

Install and Configure Edge Node 1	271
Configure the N-VDS Switch or Switches for Edge Node 1	274
Complete the Edge Node 1 Installation	274
Install and Configure Edge Node 2	275
Configure the N-VDS Switch or Switches for Edge Node 2	276
Complete the Installation of Edge Node 2	276
Next Steps	276
 Installing and Configuring NSX-T Transport Nodes	277
Prerequisites	277
Prerequisites for Installing NSX-T Data Center v3.0 for Tanzu Kubernetes Grid Integrated Edition	277
Deploy ESXi Host Transport Nodes Using VDS	277
Verify TEP to TEP Connectivity	279
Create NSX-T Edge Cluster	280
Create Uplink Logical Switch	281
Create Tier-0 Router	283
Next Steps	286
 Create the VMware NSX Objects for Kubernetes Clusters Provisioned by TKGI	286
Prerequisites	286
Required NSX-T Objects for the Tanzu Kubernetes Grid Integrated Edition Control Plane	286
Create NSX-T Objects for Kubernetes Clusters Using the Management Interface	287
Create Tier-0 Router Using the Management Interface	287
Configure and Test the Tier-0 Router	291
Create the Nodes IP Block for Kubernetes Clusters Using the Management Interface	293
Create the Pods IP Block for Kubernetes Clusters Using the Management Interface	294
Create the Floating IP Pool for Kubernetes Clusters Using the Management Interface	295
Create NSX-T Objects for Kubernetes Clusters Using the Policy Interface	296
Create a Tier-0 Gateway Using the Policy Interface	297
Configure the Tier-0 Gateway Using the Policy Interface	298
Test the Tier-0 Gateway	300
Create the Nodes IP Block for Kubernetes Clusters Using the Policy Interface	301
Create the Pods IP Block for Kubernetes Clusters Using the Policy Interface	301
Create the Floating IP Pool for Kubernetes Clusters Using the Management Interface	301
Next Steps	302

Create VMware NSX Objects for the Management Plane	302
Prerequisites	302
Create Management Plane	302
Create Tier-1 Router and Switch	303
Create NAT Rules	308
Next Steps	311
Configure VMware NSX Passwords	311
Prerequisites	312
Configure the NSX-T Password Interval (Optional)	312
Update the NSX-T Manager Password and Password Interval	312
SSH into the NSX Manager Node	312
Retrieve the Password Expiration Interval	313
Update the Admin Password	313
Set the Admin Password Expiration Interval	313
Remove the Admin Password Expiration Interval	313
Update the Password for NSX Edge Nodes	313
Enable SSH	314
SSH to the NSX Edge Node	314
Get the Password Expiration Interval for the Edge Node	314
Update the User Password for the Edge Node	314
Set the Password Expiration Interval	314
Remove the Password Expiration Interval	314
Next Step	315
Deploying Ops Manager with VMware NSX for Tanzu Kubernetes Grid Integrated Edition	315
Prerequisites	315
Step 1: Generate SSH Key Pair	315
Step 2: Deploy Ops Manager for Tanzu Kubernetes Grid Integrated Edition	316
Network Selection for vSphere v6.5	325
Step 3: Configure Ops Manager for Tanzu Kubernetes Grid Integrated Edition	326
Next Step	329
Configuring BOSH Director with VMware NSX for Tanzu Kubernetes Grid Integrated Edition	329
Prerequisites	329
How Ops Manager Accesses NSX-T Manager	330
Step 1: Open the BOSH Director Tile	330
Step 2: Configure vCenter for Tanzu Kubernetes Grid Integrated Edition	331
Step 3: Configure BOSH Director	335

Step 4: Create Availability Zones	336
Step 5: Create Networks	340
Step 6: Assign AZs and Networks	342
Step 7: Configure Security	343
Step 8: Configure BOSH DNS	344
Step 9: Configure Logging	344
Step 10: Configure Resources	345
Step 11: (Optional) Add Custom VM Extensions	345
Step 12: Deploy BOSH	345
Step 13: Update Network Availability Zones	348
Next Step	349
Generating and Registering the VMware NSX Manager Superuser Principal Identity Certificate and Key	349
Installation Prerequisites	350
How Ops Manager Accesses NSX-T Manager	350
About the NSX-T Manager Super User Principal Identity	351
Generating the Certificate and Key for Installation	351
Option A: Generate and Register the Certificate and Key Using a Script	352
Step 1: Generate and Register the Certificate and Key	352
Option B: Generate and Register the Certificate and Key Using the Tanzu Kubernetes Grid Integrated Edition Tile	354
Step 1: Generate the Certificate and Key	354
Step 2: Copy the Certificate and Key to the Linux VM	354
Step 3: Export Environment Variables	355
Step 4: Register the Certificate	355
Step 6: Verify the Certificate and Key	355
Rotate the Principal Identity Certificate and Key	355
Next Installation Step	356
Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with VMware NSX	356
Prerequisites	356
Step 1: Install Tanzu Kubernetes Grid Integrated Edition	356
Step 2: Configure Tanzu Kubernetes Grid Integrated Edition	356
Assign AZs and Networks	357
TKGI API	359
Plans	361
Activate a Plan	361
Deactivate a Plan	367
Kubernetes Cloud Provider	368

Networking	369
UAA	373
(Optional) Host Monitoring	375
Syslog	376
VMware vRealize Log Insight Integration	377
(Optional) In-Cluster Monitoring	379
Wavefront	380
VMware vRealize Operations Management Pack for Container Monitoring	381
Metric Sink Resources	381
Log Sink Resources	382
Tanzu Mission Control	382
VMware CEIP	384
Storage	385
Errands	386
Resource Config	387
Step 3: Apply Changes	388
Step 4: Install the TKGI and Kubernetes CLIs	388
Step 5: Verify NAT Rules	388
Step 6: Configure Authentication for Tanzu Kubernetes Grid Integrated Edition	388
Next Steps	389
 Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on vSphere	389
Overview	389
Prerequisites	389
Step 1: Connect to the TKGI API VM	389
Option 1: Connect through the Ops Manager VM	389
Option 2: Connect through a Non-Ops Manager Machine	390
Step 2: Log In as a UAA Admin	390
Step 3: Assign Tanzu Kubernetes Grid Integrated Edition Cluster Scopes	391
Next Step	392
 Advanced Configurations for Tanzu Kubernetes Grid Integrated Edition on vSphere with VMware NSX	392
Post-Installation NSX-T Configurations	392
 Provisioning a VMware NSX Load Balancer for the TKGI API Server	392
About the NSX-T Load Balancer for the TKGI API Server	393
Step 1: Create NSGroup	393
Step 2: Create Two Virtual Servers	394
Create a Virtual Server for the TKGI API Server	394

Create a Virtual Server for the UAA Server	401
Step 3: Create Load Balancer	406
Step 4: Attach the Load Balancer to a Logical Router	407
Troubleshooting LB-Router Attachment	408
Step 5: Attach the Load Balancer to the Virtual Servers	412
Step 6: Configure TKGI to Use the Load Balancer	414
Step 7: Test the Load Balancer	415
Provisioning a Load Balancer for the VMware NSX Management Cluster	417
About the NSX-T Management Cluster	417
Provision the NSX-T Load Balancer for the Management Cluster	418
Step 1: Log in to an NSX-T Manager Node	418
Step 2: Configure a Logical Switch	418
Step 3: Configure a Tier-1 Logical Router	419
Step 4: Advertise the Routes	419
Step 5: Verify Router and Switch Configuration	419
Step 6: Configure a Small Load Balancer	420
Step 7: Attach the Load Balancer to the Router	420
Step 8: Configure a Virtual Server	420
Step 9: Attach the Virtual Server to the Load Balancer	421
Step 10: Verify the Load Balancer.	422
Step 11: Create an Active Health Monitor (HM)	422
Step 12: Create SNAT Rule	423
Step 13: Verify that NSX-T Manager Traffic Is Load Balanced	423
Using Proxies with Tanzu Kubernetes Grid Integrated Edition on VMware NSX	424
Overview	424
Enable TKGI API and Kubernetes Proxy	425
Enable Ops Manager and BOSH Proxy	428
Isolating Tenants	429
About Tenant Isolation	429
Using a Multi-TO Router Configuration for Tenant Isolation	429
Using a VRF Tier-0 Gateway Configuration for Tenant Isolation	430
Prerequisites	431
Multi-TO-Based Tenant Isolation Prerequisites	431
VRF Tier-0 Gateway-Based Tenant Isolation Prerequisites	431
Configure Multi-TO Router-Based Tenant Isolation	432

Step 1: Plan and Provision Additional NSX-T Edge Nodes for Each Multi-TO Router	432
Step 2: Configure Inter-T0 Logical Switch	433
Step 3: Configure a New Uplink Interface on the Shared Tier-0 Router	433
Step 4: Provision Tier-0 Router for Each Tenant	434
Step 5: Create Two Uplink Interfaces on Each Tenant Tier-0 Router	434
Step 6: Verify the Status of the Shared and Tenant Tier-0 Routers	434
Step 7: Configure Static Routes	435
Step 8: Considerations for NAT Topology on Shared Tier-0	436
Step 9: Considerations for NAT Topology on Tenant Tier-0	436
Step 10: Configure BGP on Each Tenant Tier-0 Router	439
Considerations When Configuring BGP on Tenant Tier-0 Routers	439
Configure BGP AS Number	440
Configure BGP Route Distribution	440
Configure IP Prefix Lists	441
Configure BGP Peer	442
Step 11: Configure BGP on the Shared Tier-0 Router	442
Configure BGP AS Number	443
Configure BGP Route Distribution	443
Configure IP Prefix Lists	443
Configure BGP Peer	444
Step 12: Test the Base Configuration	445
Shared Tier-0 Validation	445
Tenant Tier-0 Validation	445
Configure Multi-T0 Security	446
Secure Inter-Tenant Communications	446
Step 1: Define IP Sets	446
Step 2: Create Edge Firewall	448
Step 3: Add Firewall Rules	448
BGP Firewall Rule	449
Clusters Masters Firewall Rule	449
Node Network to Management Firewall Rule	450
TKGI Firewall Rule	450
Deny All Firewall Rule	451
(Optional) Step 4: Create DFW Section	451
Secure Intra-Tenant Communications	451
Step 1: Create NSGroup for All Tanzu Kubernetes Grid Integrated Edition Clusters	452
Step 2: Create DFW Section	453
Step 3: Create NSGroups	453

Create NSGroup for Cluster Nodes	453
Create NSGroup for Cluster Pods	454
Create NSGroup for Cluster Nodes and Pods	454
Step 4: Create DFW Rules	455
DFW Rule 1: Deny Everything Else	455
DFW Rule 2: Prevent Pod to Node Communication	456
DFW Rule 3: Allow Node to Node and Nodes to Pods Communications	456
Configure VRF Tier-0 Gateway-Based Tenant Isolation	457
Step 1: Review Your Network Configuration	457
Step 2: Create VRF Gateway Segments	457
Step 3: Create VRF Gateways	458
Step 4: Create a Network Profile	459
Step 5: Configure a Cluster with a VRF Gateway	460
Implementing a Multi-Foundation Tanzu Kubernetes Grid Integrated Edition Deployment	461
About Multi-Foundation Tanzu Kubernetes Grid Integrated Edition	461
Requirements	462
Install Tanzu Kubernetes Grid Integrated Edition on vSphere Using Ops Manager (Antrea and Flannel Networking)	463
Install the TKGI and Kubernetes CLIs	463
vSphere Prerequisites and Resource Requirements	463
Prerequisites	464
vSphere Version Requirements	464
Resource Requirements	464
Storage Requirements for Large Numbers of Pods	464
Ephemeral VM Resources	464
Kubernetes Cluster Resources	465
Network Communication Requirements	465
Firewall Ports and Protocols Requirements for vSphere (Antrea and Flannel Networking)	465
Overview	465
TKGI Ports and Protocols	466
TKGI Users Ports and Protocols	466
TKGI Core Ports and Protocols	467
VMware Ports and Protocols	470
VMware Virtual Infrastructure Ports and Protocols	470
VMware Optional Integration Ports and Protocols	471

Creating Dedicated Users and Roles for vSphere (Optional)	472
Overview	473
Prerequisites	474
Create the Master Node User Account	474
Grant Storage Permissions	474
Static Only Persistent Volume Provisioning	475
Dynamic Persistent Volume Provisioning (with Storage Policy-Based Volume Placement)	476
Dynamic Volume Provisioning (without Storage Policy-Based Volume Placement)	477
Create the BOSH/Ops Manager User Account	478
Grant Permissions to the BOSH/Ops Manager User Account	478
Configure DNS for the TKGI API	478
Next Installation Step	478
Installing and Configuring Ops Manager on vSphere	478
Prerequisites	478
Install and Configure Ops Manager	479
Next Installation Step	479
Installing Tanzu Kubernetes Grid Integrated Edition on vSphere (Antrea and Flannel Networking)	479
Prerequisites	479
Step 1: Install Tanzu Kubernetes Grid Integrated Edition	479
Step 2: Configure Tanzu Kubernetes Grid Integrated Edition	480
Assign AZs and Networks	480
TKGI API	481
Plans	483
Activate a Plan	483
Deactivate a Plan	490
Kubernetes Cloud Provider	491
Networking	492
UAA	496
(Optional) Host Monitoring	499
Syslog	500
VMware vRealize Log Insight Integration	501
(Optional) In-Cluster Monitoring	503
Wavefront	504
VMware vRealize Operations Management Pack for Container Monitoring	505
Metric Sink Resources	505

Log Sink Resources	506
Tanzu Mission Control	506
VMware CEIP	508
Storage	509
Errands	510
Resource Config	511
Step 3: Apply Changes	512
Next Installation Step	512
 Configuring a TKGI API Load Balancer	512
Overview	512
Prerequisites	512
Step 1: Retrieve the TKGI API Endpoint	512
Step 2: Configure an External Load Balancer	513
Next Installation Step	513
 Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on vSphere	513
Overview	513
Prerequisites	513
Step 1: Connect to the TKGI API VM	513
Option 1: Connect through the Ops Manager VM	513
Option 2: Connect through a Non-Ops Manager Machine	514
Step 2: Log In as a UAA Admin	514
Step 3: Assign Tanzu Kubernetes Grid Integrated Edition Cluster Scopes	515
Next Step	516
 Install Tanzu Kubernetes Grid Integrated Edition on VMware Cloud Foundation	516
About TKGI Integration with VCF	516
Requirements for Installing TKGI on VCF	516
Supported Topologies for TKGI on VCF	517
Topology 1: Workload Domain with a Single vSphere Cluster	517
Topology 2: Workload Domain with Multiple vSphere Clusters	518
TKGI on VCF Deployment Procedure	518
 Installing Tanzu Kubernetes Grid Integrated Edition on Google Cloud Platform (GCP)	520
Install Tanzu Kubernetes Grid Integrated Edition on GCP	520
Install the TKGI and Kubernetes CLIs	520
 GCP Prerequisites and Resource Requirements	521

Prerequisites	521
Resource Requirements	521
Storage Requirements for Large Numbers of Pods	522
Kubernetes Cluster Resources	522
Installing and Configuring Ops Manager on GCP	522
Prerequisites	522
Install and Configure Ops Manager	522
Next Installation Step	523
Creating Service Accounts in GCP for Tanzu Kubernetes Grid Integrated Edition	523
Create the Control Plane Node Service Account	523
Create the Worker Node Service Account	523
Next Installation Step	524
Creating a GCP Load Balancer for the TKGI API	524
Overview	524
Create a Load Balancer	524
Create a Firewall Rule	526
Create a DNS Entry	526
Install Tanzu Kubernetes Grid Integrated Edition	527
Installing Tanzu Kubernetes Grid Integrated Edition on GCP (Antrea and Flannel Networking)	527
Prerequisites	527
Step 1: Install Tanzu Kubernetes Grid Integrated Edition	527
Step 2: Configure Tanzu Kubernetes Grid Integrated Edition	528
Assign AZs and Networks	528
TKGI API	529
Plans	531
Activate a Plan	532
Deactivate a Plan	538
Kubernetes Cloud Provider	539
Networking	540
UAA	541
(Optional) Host Monitoring	543
Syslog	544
(Optional) In-Cluster Monitoring	545
Wavefront	546
cAdvisor	547

Metric Sink Resources	547
Log Sink Resources	548
Tanzu Mission Control	548
VMware CEIP	548
Errands	549
Resource Config	550
Step 3: Apply Changes	551
Step 4: Retrieve the TKGI API Endpoint	551
Step 5: Configure External Load Balancer	552
Step 6: Install the TKGI and Kubernetes CLIs	552
Step 7: Configure Authentication for Tanzu Kubernetes Grid Integrated Edition	552
Next Steps	552
Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on GCP	552
Overview	553
Prerequisites	553
Step 1: Connect to the TKGI API VM	553
Option 1: Connect through the Ops Manager VM	553
Option 2: Connect through a Non-Ops Manager Machine	553
Step 2: Log In as a UAA Admin	554
Step 3: Assign Tanzu Kubernetes Grid Integrated Edition Cluster Scopes	555
Next Step	555
Installing Tanzu Kubernetes Grid Integrated Edition on Amazon Web Services (AWS)	555
Install Tanzu Kubernetes Grid Integrated Edition on AWS	556
Install the TKGI and Kubernetes CLIs	556
AWS Prerequisites and Resource Requirements	556
Prerequisites	556
Resource Requirements	556
Storage Requirements for Large Numbers of Pods	557
Kubernetes Cluster Resources	557
Installing and Configuring Ops Manager on AWS	557
Prerequisites	557
Install and Configure Ops Manager	557
Next Installation Step	558
Installing Tanzu Kubernetes Grid Integrated Edition on AWS (Antrea and Flannel Networking)	558

Prerequisites	558
Step 1: Install Tanzu Kubernetes Grid Integrated Edition	558
Step 2: Configure Tanzu Kubernetes Grid Integrated Edition	558
Assign AZs and Networks	559
TKGI API	560
Plans	562
Activate a Plan	562
Deactivate a Plan	569
Kubernetes Cloud Provider	570
Networking	570
UAA	572
(Optional) Host Monitoring	574
Syslog	575
(Optional) In-Cluster Monitoring	576
Wavefront	577
cAdvisor	578
Metric Sink Resources	578
Log Sink Resources	579
Tanzu Mission Control	579
VMware CEIP	581
Errands	582
Resource Config	583
Step 3: Apply Changes	584
Step 4: Retrieve the TKGI API Endpoint	584
Step 5: Install the TKGI and Kubernetes CLIs	585
Step 6: Configure Authentication for Tanzu Kubernetes Grid Integrated Edition	585
Next Steps	585
 Using Proxies with Tanzu Kubernetes Grid Integrated Edition on AWS	585
Overview	585
Enable TKGI API and Kubernetes Proxy	586
Enable Ops Manager and BOSH Proxy	589
 Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on AWS	590
Overview	590
Prerequisites	590
Step 1: Connect to the TKGI API VM	590
Option 1: Connect through the Ops Manager VM	590
Option 2: Connect through a Non-Ops Manager Machine	591

Step 2: Log In as a UAA Admin	591
Step 3: Assign Tanzu Kubernetes Grid Integrated Edition Cluster Scopes	592
Next Step	593
Installing Tanzu Kubernetes Grid Integrated Edition on Microsoft Azure	593
Install Tanzu Kubernetes Grid Integrated Edition on Azure	593
Install the TKGI and Kubernetes CLIs	593
Azure Prerequisites and Resource Requirements	594
Prerequisites	594
Subscription Requirements	594
Resource Requirements	594
Storage Requirements for Large Numbers of Pods	594
Kubernetes Cluster Resources	595
Installing and Configuring Ops Manager on Azure	595
Prerequisites	595
Install and Configure Ops Manager	595
Next Installation Step	595
Creating Managed Identities in Azure for Tanzu Kubernetes Grid Integrated Edition	595
Retrieve Your Subscription ID and Resource Group	596
Create the Control Plane Node Managed Identity	596
Create the Worker Node Managed Identity	598
Next Installation Step	599
Installing Tanzu Kubernetes Grid Integrated Edition on Azure (Antrea and Flannel Networking)	599
Prerequisites	599
Step 1: Install Tanzu Kubernetes Grid Integrated Edition	599
Step 2: Configure Tanzu Kubernetes Grid Integrated Edition	600
Assign Networks	600
TKGI API	601
Plans	603
Activate a Plan	603
Deactivate a Plan	610
Kubernetes Cloud Provider	611
Networking	613
UAA	615
(Optional) Host Monitoring	617

Syslog	618
(Optional) In-Cluster Monitoring	618
Wavefront	619
cAdvisor	620
Metric Sink Resources	620
Log Sink Resources	621
Tanzu Mission Control	621
VMware CEIP	623
Errands	624
Resource Config	625
Step 3: Apply Changes	626
Step 4: Retrieve the TKGI API Endpoint	626
Step 5: Configure an Azure Load Balancer for the TKGI API	627
Step 6: Install the TKGI and Kubernetes CLIs	627
Step 7: Configure Authentication for Tanzu Kubernetes Grid Integrated Edition	627
Next Steps	627
Configuring an Azure Load Balancer for the TKGI API	627
Overview	628
Create a Load Balancer	628
Create a Backend Pool	628
Create Health Probes	629
Create Load Balancing Rules	629
Create an Inbound Security Rule	630
Add the TKGI API to the Backend Pool	631
Verify TKGI API Hostname Resolution	631
Next Step	631
Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on Azure	631
Overview	632
Prerequisites	632
Step 1: Connect to the TKGI API VM	632
Option 1: Connect through the Ops Manager VM	632
Option 2: Connect through a Non-Ops Manager Machine	633
Step 2: Log In as a UAA Admin	633
Step 3: Assign Tanzu Kubernetes Grid Integrated Edition Cluster Scopes	634
Next Step	635
Firewall Ports and Protocols Requirements (Antrea and Flannel Networking)	635

Overview	635
TKGI Ports and Protocols	635
TKGI Users Ports and Protocols	636
TKGI Core Ports and Protocols	637
Networking Ports and Protocols	639
Antrea Networking Ports and Protocols	639
Installing the TKGI CLI	640
Install the TKGI CLI	640
Mac OS X	640
Linux	640
Windows	641
Installing the Kubernetes CLI	641
Mac OS X	641
Linux	641
Windows	642
Upgrading Tanzu Kubernetes Grid Integrated Edition	643
Upgrade Tanzu Kubernetes Grid Integrated Edition Management Console	643
Prerequisites	643
Step 1: Deploy the New OVA Template	644
Step 2: Log In to the New Version of Tanzu Kubernetes Grid Integrated Edition Management Console	644
Step 3: Migrate the Configuration from the Old Appliance to the New Version	644
Next Steps	646
Upgrading Tanzu Kubernetes Grid Integrated Edition with Ops Manager	647
Upgrade Preparation Checklist for Tanzu Kubernetes Grid Integrated Edition	647
Overview	647
Back Up Your Tanzu Kubernetes Grid Integrated Edition Deployment	648
Review What Happens During Tanzu Kubernetes Grid Integrated Edition Upgrades	648
Review Changes in Tanzu Kubernetes Grid Integrated Edition v1.16	648
Determine Upgrade Order (vSphere Only)	649
Set User Expectations and Restrict Cluster Access	649
Upgrade All Clusters to v1.15	649
Migrate from PSP to PSA Controller	649

Verify Your Clusters Support Upgrading	650
Verify Health of Kubernetes Environment	650
Verify Your Environment Configuration	651
Verify Your vSphere with NSX-T Configuration	651
Verify Your Antrea Environment Configuration	651
Clean Up or Fix Failed Kubernetes Clusters	651
Verify Kubernetes Clusters Have Unique External Hostnames	652
Verify TKGI Proxy Configuration	652
Check PodDisruptionBudget Value	653
(Optional) Configure Node Drain Behavior	653
Configure with the TKGI Tile	653
Configure with the TKGI CLI	653
Upgrade Order for Tanzu Kubernetes Grid Integrated Edition Environments on vSphere	655
Overview	655
TKGI on vSphere with NSX-T Networking	655
Scenario 1: Upgrading to TKGI v1.16	656
Scenario 2: Upgrading to TKGI v1.16 and NSX-T v3.2	656
TKGI on vSphere (Antrea and Flannel Networking)	657
Scenario 1: Upgrading to TKGI v1.16	657
Upgrading Tanzu Kubernetes Grid Integrated Edition (Antrea and Flannel Networking)	658
Overview	658
Prepare to Upgrade	658
Perform the Upgrade	658
Upgrade Ops Manager	659
Download and Import Tanzu Kubernetes Grid Integrated Edition v1.16	659
Download and Import Stemcells	660
Modify Container Network Interface Configuration	661
Verify Errand Configuration	661
Verify Other Configurations	662
Apply Changes to the Tanzu Kubernetes Grid Integrated Edition Tile	662
After the Upgrade	663
Upgrade the TKGI and Kubernetes CLIs	663
Verify the Upgrade	663
Upgrading Tanzu Kubernetes Grid Integrated Edition (VMware NSX Networking)	663
Overview	664

Prerequisites	664
Prepare to Upgrade	664
Perform the Upgrade	664
Upgrade NSX-T Data Center to v3.2.2	664
Upgrade Ops Manager	666
Download and Import TKGI v1.16	668
Download and Import Stemcells	669
Upgrade the TKGI Tile	670
After the Upgrade	671
Upgrade the TKGI and Kubernetes CLIs	671
Upgrade Kubernetes Clusters If Needed	671
Verify TKGI Upgrade	672
Maintaining Workload Uptime	672
About Cluster Upgrades	673
Set Workload Replicas	673
Define an Anti-Affinity Rule	674
Multi-AZ Worker	675
PersistentVolumes	675
Configuring the Upgrade Pipeline	675
Download the Upgrade Pipeline	676
Configure Automated Ops Manager and Ubuntu Jammy Stemcell Downloading	676
Managing Tanzu Kubernetes Grid Integrated Edition	679
Monitor and Manage Tanzu Kubernetes Grid Integrated Edition in the Management Console	679
Obtain General Status Information	679
Obtain Deployment Metadata	680
View Component Deployment Status	681
Reconfigure Your Tanzu Kubernetes Grid Integrated Edition Deployment	681
Reconfigure Your Tanzu Kubernetes Grid Integrated Edition Deployment in the Wizard	682
Reconfigure Your Tanzu Kubernetes Grid Integrated Edition Deployment by Importing a YAML File	682
Which Options Can I Reconfigure?	683
Patch Tanzu Kubernetes Grid Integrated Edition Management Console Components	685

Delete Your Tanzu Kubernetes Grid Integrated Edition Deployment	686
Managing Tanzu Kubernetes Grid Integrated Edition Users	687
Identity Management in the Management Console	687
Add Individual Users	687
Add User Groups	689
Remove Individual Users	690
Remove User Groups	690
Next Steps	690
Managing Tanzu Kubernetes Grid Integrated Edition Users in Ops Manager	691
Connecting Tanzu Kubernetes Grid Integrated Edition to an LDAP Server	691
Overview	691
Integrate UAA with an LDAP Server	691
Complete Your Tile Configuration	695
Next Steps	695
Configuring Okta as a SAML Identity Provider	695
Prerequisites	695
Configure SAML in Okta	695
Configuring Azure Active Directory as a SAML Identity Provider	699
Prerequisites	699
Configure SAML in Azure AD	699
Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider	702
Overview	702
Configure a SAML IdP	703
Integrate UAA with the SAML IdP	703
Complete Your Tile Configuration	707
Next Steps	707
Managing Tanzu Kubernetes Grid Integrated Edition Users with UAA	707
Overview	707
UAA Scopes for Tanzu Kubernetes Grid Integrated Edition Users	707
Prerequisites	708
Log In as a UAA Admin	708
Grant Tanzu Kubernetes Grid Integrated Edition Access to an Individual User	709

Grant Tanzu Kubernetes Grid Integrated Edition Access to an External Group	710
Grant Tanzu Kubernetes Grid Integrated Edition Access to an External LDAP Group	710
Grant Tanzu Kubernetes Grid Integrated Edition Access to an External SAML Group	711
Grant Tanzu Kubernetes Grid Integrated Edition Access to a Client	711
UAA Scopes for Tanzu Kubernetes Grid Integrated Edition Users	712
Overview	712
UAA Scopes	712
OIDC Provider for Kubernetes Clusters	713
Overview	713
UAA as the Default OIDC Provider	713
Custom OIDC Provider	714
After You Configure Your OIDC Provider	714
Managing Kubernetes Cluster Options	714
Limit Resource Usage	714
Assign Resource Quotas to Users in the Management Console	715
Overview	715
Assign a Resource Quota to a User	715
Modify a User Resource Quota	716
Delete a User Resource Quota	717
Next Steps	717
Managing Resource Usage with Quotas	717
Overview	718
Set up Your API Access Token	718
Manage Quotas	719
Add a Quota	719
Modify an Existing Quota	720
Delete a Quota	720
View Quotas	721
View Quotas for a Single User	721
View All Quotas	721
Error Message When User Exceeds Cluster Quota	722
View Usage	722
View Resource Usage by User	722

View All Resource Usage	722
Viewing Usage Quotas	723
Overview	723
Set up Your API Access Token	723
View Quotas	724
View Usage	724
Error Message When You Exceed Cluster Quota	725
Network Profiles (VMware NSX Only)	725
Creating and Managing Network Profiles in the Management Console	725
Using Network Profiles	726
Requirements for Network Profiles	726
Create Cluster with Network Profile	726
Define Network Profile	726
Delete Network Profile	728
Advanced Network Parameters	729
Container Networks Parameters	730
Working with Network Profiles in Ops Manager	730
Creating and Managing Network Profiles (NSX Only)	731
Prerequisite	731
Overview	731
Create a Network Profile	732
Network Profile Example	734
Update an Existing Network Profile	735
Confirm the Network Profile Property Supports Updates	735
Create a Modified Network Profile Configuration	735
Create a Modified Network Profile	736
Update the Cluster with a Modified Network Profile	736
Update-Cluster Network Profile Validation Rules	737
Delete a Network Profile	737
Network Profile Parameters	737
Top-Level Parameters	737
cni_configurations Parameters	740
cni_configurations Extensions Parameters	745
Parameter Descriptions	750
client_ssl_profile	750
enable_hostport	750

enable_nodelocaldns	751
extensions	751
fip_pool_ids	751
lb_connection_multiplexing_enabled	752
lb_connection_multiplexing_number	752
nodes_dns	752
pod_ip_block_ids	752
Network Profile Use Cases	753
Size a Load Balancer	754
Load Balancer Sizing	754
Customizing Pod Networks (NSX-T Only)	755
Custom Pod Networks	755
Pod Subnet Prefix	756
Routable Pod Networks	757
Add Pod IPs	758
Customize Node Networks	758
Configurable Node Network IP Blocks	758
Node IP Block IDs	759
Node Routable	760
Node Subnet Prefix	760
Customize Floating IP Pools	761
Create a Custom Floating IP Pool	761
Modify a Floating IP Pool	762
Configure Bootstrap NSGroups	762
Bootstrap Security Group	762
Configure Edge Router Selection	763
Edge Router Selection	763
Specify Nodes DNS Servers	765
DNS Configuration for Kubernetes Clusters	765
Configure DNS for Pre-Provisioned IPs	766
About DNS Lookup of Pre-Provisioned IP Addresses	766
DNS Lookup Parameters	766
Example API Load Balancer Lookup	766

Performing DNS Lookup of the Ingress Controller Using Network Profile	767
Setting the Control Plane Node IP Address on the Command Line	767
Configure the TCP Layer 4 Load Balancer	768
Overview	768
Configure the TCP Ingress Controller Network Profile	769
NSX-T TCP Ingress Controller Network Profile Configuration	769
Configure Which NSX Load Balancer to Use	771
Configure the Client Source IP Address	772
Configure the Maximum Number of Layer 4 Load Balancer Virtual Servers	773
Configure the Layer 4 Persistence Type	773
Configure the Layer 4 Load Balancer Algorithm	774
Configure the HTTP/S Layer 7 Ingress Controller	774
Overview	774
Configure the HTTP/HTTPS Ingress Controller Network Profile	775
NSX-T HTTP/HTTPS Ingress Controller Network Profile Configuration	775
Configure the NSX Ingress Controller	777
Configure the Ingress IP	778
Configure the Ingress Persistence Settings	779
Defining DFW Section Markers	780
Overview	780
About DFW Section Markers	781
Create a Top Firewall Section Marker	781
Create a Bottom Firewall Section Marker	782
Configure NCP Logging	782
About Logging for NCP Configurations	782
Parameters for NCP Logging	782
Example Network Profile for NCP Logging	783
Log Settings	783
Log Level	783
Log Dropped Traffic	784
enable_err_crd	784
Shared and Dedicated Tier-1 Router Topologies	785
Shared Tier-1 Topology	785
Comparison to Dedicated Tier-1	786
Dedicated Tier-1 Topology	786
Network Profile for Dedicated Tier-1 Topology	787

Implementing a Shared Tier-1 Topology in a Multi-Tier-0 Environment	787
Compute Profiles and Host Groups (vSphere Only)	788
Creating and Managing Compute Profiles in the Management Console	788
About Compute Profiles	789
Create Cluster with Compute Profile	789
Define Compute Profile	791
Delete Compute Profile	793
Creating and Managing Compute Profiles with the CLI (vSphere)	794
About Compute Profiles	794
Create a Compute Profile	795
Compute Profile Format	795
Custom Nodes	795
Worker Node Pools	796
Custom AZs (vSphere with NSX-T Only)	797
Compute Profile Parameters	799
azs Block (vSphere with NSX-T Only)	799
Retrieve the BOSH CPI ID	800
control_plane Block	801
node_pools Block	802
The create-compute-profile Command	804
Manage Compute Profiles	804
View a Compute Profile	805
Delete a Compute Profile	805
Cluster Manager Operations	806
Compute Profiles vs. Plans	806
Using Compute Profiles (vSphere)	806
How Compute Profiles are Created	806
List Compute Profiles	807
Create a Cluster with a Compute Profile	807
Assign a Compute Profile to an Existing Cluster	808
Compute Profile Update and Resize Validation	808
Assign a Compute Profile	809
Resize a Cluster that Has an Existing Compute Profile	810
Using vSphere Host Groups with Tanzu Kubernetes Grid Integrated Edition	811
About vSphere Host Groups	811

Host Group Use Cases for Tanzu Kubernetes Grid Integrated Edition	811
Enabling Support for vSAN Fault Domains	811
Using Host Group as a New AZ in BOSH	811
Defining a Host Group in vSphere	812
Using a Host Group with Tanzu Kubernetes Grid Integrated Edition	812
 Adding Infrastructure Password Changes to the Tanzu Kubernetes Grid Integrated Edition Tile	 813
Manage Your Service Account Passwords	813
Step 1: Update Your Service Account Passwords	813
Step 2: Deploy Your New Service Account Passwords	815
Manage Your NSX Manager Password (vSphere and vSphere with NSX-T only)	815
Troubleshooting	816
‘Failed to authenticate user’ Error When Cluster Service Account Authenticates	816
 Configuring VMware Tanzu Service Mesh by VMware NSX	 817
About VMware Tanzu Service Mesh by VMware NSX	817
Prerequisites	818
Install VMware Tanzu Service Mesh in a Cluster	818
Add the Tanzu Service Mesh Service	818
Onboard a Kubernetes Cluster to Tanzu Service Mesh	818
Install and Configure Istio	819
Known Issue: Timeout	819
 Shutting Down and Restarting Tanzu Kubernetes Grid Integrated Edition	 820
Shutdown Sequence and Tasks	820
Step 1: Deactivate BOSH Resurrection	820
Step 2: Delete All PodDisruptionBudgets	820
Step 3: Shut Down Customer Apps	821
Step 4: Shut Down Kubernetes Clusters	821
Step 5: Stop the TKGI Control Plane	822
Stop TKGI Control Plane Processes	822
Shut Down the TKGI API and Database VMs	823
Step 6: Shut Down VMware Harbor Registry (vSphere Only)	824
Step 7: Shut Down BOSH Director	825
Step 8: Shut Down Ops Manager	825
Step 9: Shut Down NSX-T Components (vSphere NSX-T Only)	826
Step 10: Shut Down vCenter Server (vSphere Only)	827
Step 11: Shut Down ESXi Hosts (vSphere NSX-T Only)	827

Startup Sequence and Tasks	828
Step 1: Start ESXi Hosts (vSphere NSX-T Only)	828
Step 2: Start vCenter (vSphere Only)	829
Step 3: Start NSX-T Components (vSphere NSX-T Only)	829
Step 4: Start Ops Manager	829
Step 5: Start the BOSH Director	829
Step 6: Start the TKGI Control Plane	830
Step 7: Start Harbor Registry (vSphere Only)	830
Step 8: Start the Kubernetes Clusters	831
Start a Cluster with Three Control Plane Nodes	831
Start a Cluster with Five Control Plane Nodes	832
Step 9: Start Customer Apps	833
Step 10: Restore All PodDisruptionBudgets	833
Step 11: Re-enable BOSH Resurrection	834
Deleting Tanzu Kubernetes Grid Integrated Edition	834
Delete the Tanzu Kubernetes Grid Integrated Edition Tile	834
Managing Kubernetes Clusters and Workloads	835
Create and Manage Clusters in the Management Console	835
Create Clusters in the Management Console	836
Create Clusters	836
Update Cluster Configuration	838
Upgrade Clusters to a New Version of Kubernetes	839
Delete Clusters	840
Next Steps	841
Monitor and Manage Clusters, Nodes, and Namespaces in the Management Console	841
Obtain Cluster Information	841
Connect to Clusters with kubectl	843
Obtain Node Information	843
Managing Clusters with the CLI	844
Logging in to Tanzu Kubernetes Grid Integrated Edition	845
Overview	845
Prerequisites	845
Log in to the TKGI CLI	845
Log in to the TKGI CLI as an Automated Client	846

Export TKGI API Access Token	847
Creating Clusters	847
Overview	847
Configure Cluster Access	848
vSphere with NSX-T	848
GCP, AWS, Azure, or vSphere without NSX-T	848
Create a Kubernetes Cluster	849
Identify Kubernetes Cluster Control Plane VMs	853
Next Steps	855
Using Kubernetes Profiles	855
Overview	855
Who Creates and Manages Kubernetes Profiles	855
Validated vs Experimental Customizations	856
“k8s” to “kubernetes” Alias in TKGI CLI	856
Create a Kubernetes Profile	856
Kubernetes Profile Format	856
Kubernetes Profile Parameters	857
The create-k8s-profile Command	858
Manage Kubernetes Profiles	859
List Kubernetes Profiles	859
Delete a Kubernetes Profile	859
View Kubernetes Profile Details	860
Create a Cluster with a Kubernetes Profile	860
Assign a Kubernetes Profile to an Existing Cluster	861
Kubernetes Profile Use Cases	861
Admission Control: ResourceQuota	862
Set Service Node Port Range	862
Restrict Request Header Names	863
Modify the Service Cluster IP Range	863
Using Network Profiles (NSX-T Only)	864
Prerequisite	864
Overview	864
List Network Profiles	864
Create a Cluster with a Network Profile	865
Assign a Network Profile to an Existing Cluster	865
Update an Existing Network Profile	866
Network Profile Use Cases	866

Using BOSH VM Extensions	867
Overview	867
Create a Cluster Using VM Extensions	867
Configure a Cluster Using VM Extensions	868
Create a Cluster Configuration File for BOSH VM Extensions	868
Remove BOSH VM Extensions From a Cluster	870
Viewing Cluster Lists	871
Viewing Cluster Details	872
Viewing Cluster Plans	873
Scaling Existing Clusters	874
Scale Horizontally by Changing the Number of Worker Nodes Using the TKGI CLI	874
Scale Vertically by Changing Cluster Node VM Sizes in the TKGI Tile	876
Tagging Clusters	877
Overview	877
Tag Your Clusters as They Are Created	877
Tag Your Existing Clusters	878
Modify Cluster Tags	878
Modify Your Existing Tags	878
Remove All Tags From Your Cluster	879
Review Your Tags	879
Tagging Rules	880
Tagging Limitations	880
Tags Reserved for BOSH	880
AWS-Specific Tagging Limitations	881
Azure-Specific Tagging Limitations	881
GCP-Specific Tagging Limitations	881
vSphere-Specific Tagging Limitations	881
Upgrading Clusters	882
Overview	882
Prerequisites	882
Upgrade Clusters	883
Upgrade a Single Cluster	883
Upgrade Multiple Clusters	883
Upgrade Clusters in Parallel	884

Upgrade Clusters With Canaries	885
Manage Your Cluster Upgrade Job	886
Monitor Your Clusters	886
Monitor Your Cluster Upgrade Job	887
Stop Your Cluster Upgrade Job	887
After Upgrading Clusters	887
Upgrade Velero	887
(Optional) Restore Cluster Sizing	888
Adding an OIDC Provider	888
Overview	888
Prerequisites	889
Configure a Custom OIDC Provider	889
Set Up Dex Workload	889
Set Up Communication Path	890
Deploy and Expose Dex	890
Create Kubernetes Profile	891
Create Cluster	892
Test Cluster Access	892
Deleting Clusters	895
Delete Cluster	895
Verify Cluster Deletion	896
Delete Cluster without Prompt	897
Supporting Clusters	897
Load Balancing and Ingress	898
Load Balancing and Ingress with VMware NSX	898
Configuring Ingress Using the NSX-T Load Balancer	898
Monitoring Ingress Resources	898
Overview	899
Monitor the NSX-T Load Balancer Service	899
Monitor Your NSX-T Load Balancer Service Using the NSXLoadBalancerMonitor CRD	900
Viewing and Troubleshooting the Health Status of Cluster Network Objects	901
About the NSX Errors CRD	901
Errors Reported by the NSX Errors CRD	901

NSX Errors CRD Example	902
Configuring Ingress Resources and Load Balancer Services	903
Kubernetes Ingress Rules	903
The NSX-T Load Balancer Service	904
Size a Load Balancer	905
Load Balancer Sizing	905
Scaling the HTTP/S Layer 7 Ingress Load Balancers Using the LoadBalancer CRD	906
Overview	906
Prerequisites	907
Scale Ingress Load Balancer Resources	907
Create a New Ingress Load Balancer	907
Configure Your Kubernetes Ingress Resource to Use the New Ingress Load Balancer	909
Configure the HTTP/S Layer 7 Ingress Controller	911
Overview	911
Configure the HTTP/HTTPS Ingress Controller Network Profile	912
NSX-T HTTP/HTTPS Ingress Controller Network Profile Configuration	912
Configure the NSX Ingress Controller	914
Configure the Ingress IP	914
Configure the Ingress Persistence Settings	916
Configure the TCP Layer 4 Load Balancer	917
Overview	917
Configure the TCP Ingress Controller Network Profile	917
NSX-T TCP Ingress Controller Network Profile Configuration	918
Configure Which NSX Load Balancer to Use	920
Configure the Client Source IP Address	921
Configure the Maximum Number of Layer 4 Load Balancer Virtual Servers	922
Configure the Layer 4 Persistence Type	922
Configure the Layer 4 Load Balancer Algorithm	922
Using Ingress URL Rewrite	923
About Support for URL Rewrite for Ingress Resources	923
URL Rewrite Example	923
Creating and Configuring an AWS Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters	924

Prerequisite	924
Configure AWS Load Balancer	924
Step 1: Define Load Balancer	924
Step 2: Assign Security Groups	925
Step 3: Configure Security Settings	925
Step 4: Configure Health Check	925
Step 5: Add EC2 Instances	926
(Optional) Step 6: Add Tags	926
Step 7: Review and Create the Load Balancer	926
Step 8: Create a Cluster	926
Step 9: Point the Load Balancer to All Control Plane VMs	926
Reconfigure AWS Load Balancer	927
Creating and Configuring an Azure Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters	927
Prerequisites	927
Create and Configure a Load Balancer	928
Create Load Balancer	928
Create Backend Pool	928
Create Health Probe	929
Create Load Balancing Rule	929
Create Inbound Security Rule	929
Verify Hostname Resolution	930
Reconfigure Load Balancer	930
Creating and Configuring a GCP Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters	930
Overview	931
Prerequisites	931
Configure GCP Load Balancer	931
Create a GCP Load Balancer	931
Create a DNS Entry	932
Create the Cluster	932
Configure Load Balancer Back End	932
Create a Network Tag	934
Create Firewall Rules	934
Access the Cluster	935
Reconfigure Load Balancer	935
Configuring Ingress Routing	936
Overview	936

Prerequisites	936
Deploy a Kubernetes Ingress Controller	936
Configure DNS	938
(Optional) Configure TLS	938
Deploy an App to the Cluster	939
Using Admission Control Plugins for Tanzu Kubernetes Grid Integrated Edition Clusters	940
Pod Security Admission in Tanzu Kubernetes Grid Integrated Edition	940
About Pod Security Admission	941
Pod Security Admission and TKGI	941
Migrate from PSP to PSA Controller	941
Enabling the SecurityContextDeny Admission Plugin	941
About the SecurityContextDeny Admission Plugin	941
When to Enable the SecurityContextDeny Admission Plugin	942
Impact of Enabling the SecurityContextDeny Admission Plugin	942
Enabling the SecurityContextDeny Admission Plugin	942
Deactivating Admission Control Plugins for Tanzu Kubernetes Grid Integrated Edition Clusters	943
Deactivating a Single Admission Control Plugin	943
Deactivating an Orphaned Admission Control Plugin	944
Retrieving Cluster Credentials and Configuration	946
Retrieve Cluster Credentials	947
Run kubectl Commands	948
Retrieving Cluster Credentials and Configuration	948
Retrieve Cluster Credentials	949
Run kubectl Commands	950
Managing Cluster Access and Permissions	950
Overview	950
Example Workflow	951
Prerequisites	951
Grant Cluster Access to a User	951
Obtain Cluster Access as a User	954
Grant Cluster Access to a Group	955
Authenticate Windows Clusters with Active Directory	957

Overview	957
Prerequisites	957
GMSA Configuration Settings	957
Create and Integrate a Cluster with AD Authentication	958
Change a Cluster's Active Directory Authentication	958
Integrate a Cluster with Active Directory	959
View a Cluster's gMSA Configuration	960
Configure Cluster Proxies	960
Overview	960
Create a Cluster with a Custom Proxy Configuration	961
Change a Cluster's Proxy Configuration	961
Proxy Configuration Settings	961
View a Cluster's Proxy Configuration	962
Getting Started with VMware Harbor Registry	963
Overview	963
Install Harbor	963
Use Harbor	964
Manage Harbor	964
Configuring Cluster Access to Private Docker Registries (Beta)	965
Overview	965
Prerequisites	965
Set up Your API Access Token	966
Create a Cluster with SSL CA Certificates	966
Update a Cluster with SSL CA Certificates	967
SSL CA Certificate Formats	968
Prepare a Certificate String for Command Line Use	968
Deploying and Managing Cloud Native Storage (CNS) on vSphere	969
Overview	969
Requirements and Limitations of the vSphere CSI Driver	970
vSphere CSI Driver Supported Features and Requirements	970
Unsupported Features and Limitations	971
Customize vSphere File Volumes	971
Prerequisites	971
Create a Cluster with Customized File Volume Parameters	971
Modify a Cluster with Customized File Volume Parameters	972
Remove File Volume Parameters from a Cluster	973

File Volume Configuration	973
File Volume DataStores Configuration	974
File Volume NetPermissions Object Configuration	974
Create or Use CNS Block Volumes	975
Create a vSphere Storage Class	975
Create a PersistentVolumeClaim	976
Create Workloads Using Persistent Volumes	976
Customize a Cluster with vSphere Topology-Aware Volume Provisioning	977
Topology Overview	977
Prepare for Topology	978
Topology Limitations and Prerequisites	978
Create a Cluster with Topology	979
Manage Clusters with Topology-Aware Volumes	980
Migrate In-Tree vSphere Storage to the vSphere CSI Driver	980
Prepare for Automatic Migration of Volumes from In-Tree vSphere Storage to CSI	980
Migrate an In-Tree vSphere Storage Volume to the vSphere CSI Driver	981
Customize and Manage vSphere CNS	982
Customize the Maximum Number of Volume Snapshots	982
Configure CNS Data Centers	984
Manage Topology After Switching to the Automatically Deployed vSphere CSI Driver	985
PersistentVolume Storage Options on vSphere	986
Considerations for Running Stateful Apps in Kubernetes	986
Persistent Volume Provisioning Support in Kubernetes	986
vSphere Support for Static and Dynamic PVs	987
Single vSphere Compute Cluster with vSAN Datastore	988
Single vSphere Compute Cluster with File System Datastore	989
Multiple vSphere Compute Clusters Each with vSAN Datastore	990
Multiple vSphere Compute Clusters Each with File System Datastore	990
Multiple vSphere Compute Clusters with Local vSAN and Shared File System Datastore	991
Multiple vSphere Compute Clusters with Shared File System Datastore	992
Configuring and Using PersistentVolumes	993
Provision a Static PV	993
Provision a Static PV for a Deployment Workload	993
Provision a Static PV for a StatefulSets Workload	994
Provision a Dynamic PV	996
Provision a Dynamic PV for Deployment Workloads	997

Provision a Dynamic PV for StatefulSets Workloads	998
Specify a Default StorageClass	999
Provision Dynamic PVs for Use with Tanzu Kubernetes Grid Integrated Edition	999
 Supporting Windows Clusters	1000
 Configuring Windows Worker-Based Kubernetes Clusters	1000
Overview	1000
Prerequisites	1001
vSphere with NSX-T Requirements	1001
vSphere with Flannel Requirements (Beta)	1002
Configure a Windows Worker-Based Kubernetes Cluster	1002
Plans	1002
Activate a Plan	1002
Networking	1008
Upload the Windows Server Stemcell	1012
Create a Windows Worker-Based Cluster	1012
 Creating a Windows Stemcell for vSphere Using Stembuild	1012
Overview of Stembuild	1012
Overview of Windows Stemcell Creation	1012
Prerequisites	1013
Create and Configure a Base VM for the BOSH Stemcell	1014
Construct and Package the BOSH Stemcell	1014
Remove Hidden Devices	1014
Update Ops Manager With the Updated Stemcell	1015
Monthly Stemcell Upgrades	1015
Known Issues	1015
 Using a Windows Pause Image for an Air-Gapped Environment	1016
Overview	1016
Prepare Your Private Docker Registry	1016
Prepare a Windows Pause Image for an Air-Gapped Environment	1016
Configure Tanzu Kubernetes Grid Integrated Edition to Use the Windows Pause Image	1017
 Deploying Workloads	1017
 Deploying and Exposing Basic Linux Workloads	1018
Overview	1018
Prerequisites	1018

vSphere without NSX-T Prerequisites	1018
GCP, AWS, Azure, and vSphere with NSX-T Prerequisites	1018
AWS Prerequisites	1018
Deploy Workloads on vSphere with NSX-T	1019
Configure Your Workload	1019
Deploy and Expose Your Workload	1020
Access Your Workload	1020
Deploy Workloads on GCP, AWS, or Azure, Using a Public-Cloud External Load Balancer	1021
Configure Your Workload	1021
Deploy and Expose Your Workload	1022
Access Your Workload	1022
Deploy AWS Workloads Using an Internal Load Balancer	1023
Configure Your Workload	1023
Deploy and Expose Your Workload	1024
Access Your Workload	1024
Deploy Workloads for a Generic External Load Balancer	1024
Configure Your Workload	1025
Deploy and Expose Your Workload	1025
Access Your Workload	1026
Deploy Workloads without a Load Balancer	1026
Configure Your Workload	1026
Deploy and Expose Your Workload	1027
Access Your Workload	1027
Deploying and Exposing Basic Windows Workloads	1028
Overview	1028
Prerequisites	1029
Access Your Windows-Based Cluster	1029
Deploy a Windows Worker Pod	1030
Configure a Pod Deployment Manifest	1030
Deploy the Pod	1032
Deploy a Service to a Windows Worker Pod	1033
Configure a Service Manifest	1033
Deploy the Service	1034
Adding Custom Linux Workloads	1035
Create YAML Configuration	1035
Apply Custom Workloads	1035

Using Helm with Tanzu Kubernetes Grid Integrated Edition	1035
Overview	1036
Install and Configure Helm	1036
Install and Configure Helm 3	1036
Install and Configure Helm 2	1036
Deploy Components and Apps Using Helm	1037
Deploy Apps Listed in Artifact HUB	1037
Deploy Your Own Apps	1037
Logging and Monitoring Tanzu Kubernetes Grid Integrated Edition	1039
Monitoring TKGI and TKGI-Provisioned Clusters on Linux	1039
Overview	1039
Logs: Syslog and vRLI	1040
Syslog	1040
vRealize Log Insight (vSphere Only)	1040
Metrics: Telegraf	1040
About Node Exporter	1041
Healthwatch	1041
Auditing Tanzu Kubernetes Grid Integrated Edition Logs	1041
TKGI API events	1041
Cluster Creation	1042
Cluster Deletion	1042
Successful Login	1042
Unsuccessful Login	1043
Successful Cluster Credential Retrieval	1043
User Creation	1043
User Deletion	1044
Telemetry Collection	1044
Kubernetes Audit Log Events	1045
Related Links	1045
Downloading Logs from VMs	1045
Overview	1046
Download Logs	1046
Configuring Telegraf in TKGI	1046
Overview	1047
Collect Metrics Using Telegraf	1047
Create a Configuration File	1047

Configure Telegraf in the Tile	1047
Troubleshoot etcd	1050
Monitoring Linux Workers and Workloads	1050
Overview	1050
Sink Resources	1051
Sink Architecture in Tanzu Kubernetes Grid Integrated Edition	1051
Overview	1052
Sink Types	1052
Sink Architecture	1053
Log Sink Architecture	1053
Metric Sink Architecture	1054
Creating and Managing Sink Resources	1055
Overview	1055
Prerequisites	1056
Create LogSink or ClusterLogSink Resources	1056
Create a Syslog ClusterLogSink or LogSink Resource	1057
Create a Webhook ClusterLogSink or LogSink Resource	1059
Create a ClusterLogSink or LogSink Resource with a Fluent Bit Output Plugin	1059
(Optional) Define a Filter for LogSink and ClusterLogSink Resources	1060
Exclude Logs or Events from Sink Output	1061
Fluent Bit ClusterLogSink or LogSink FilterSpecs	1061
Create a Fluent Bit ClusterLogSink or LogSink FilterSpec	1063
Define a Filter Condition	1063
Chose a Filter Condition Type	1064
Create a Filter Key	1065
(Optional) Unsecured ClusterLogSink and LogSink Log Forwarding	1065
Create ClusterMetricSink and MetricSink Resources	1066
ClusterMetricSink Resources	1066
How It Works	1066
MetricSink Resources	1066
When to Use MetricSink vs. ClusterMetricSink	1066
Create a ClusterMetricSink or MetricSink Resource	1067
Create a ClusterMetricSink Resource for Node Exporter Metrics	1068
List Sinks	1069
ClusterLogSink and LogSink Resources	1069
ClusterMetricSink and MetricSink Resources	1069

Delete Sinks	1070
ClusterLogSink and LogSink Resources	1070
ClusterMetricSink and MetricSink Resources	1070
Monitoring Clusters with Log Sinks	1070
Overview	1071
Log Sinks	1071
Log Format	1071
Syslog Format	1071
Pod Logs	1072
Kubernetes API Events	1072
Notable Kubernetes API Events	1072
Failure to Retrieve Containers from Registry	1073
Malfunctioning Containers	1073
Successful Scheduling of Containers	1073
Failure to Schedule Containers	1074
Related Links	1074
Monitoring Windows Worker Clusters and Nodes	1074
Overview	1074
Healthwatch	1075
Wavefront	1075
Prometheus with Grafana	1075
Prerequisites	1075
Overview	1076
Install Prometheus and Grafana	1076
Download the Prometheus Source Code	1076
Generate Your Prometheus Dashboard Configuration	1076
Generate Monitoring Rules	1077
Deploy Prometheus and Grafana	1077
Verify Grafana is Running	1078
Install the Windows Node Exporter	1078
Deploy the Windows Node Exporter	1078
Configure the Windows Node Exporter	1079
apiVersion: v1 kind: Endpoints metadata: labels: k8s-app: wmi-exporter name: wmi-exporter namespace: kube-system subsets: - addresses: - ip: 192.168.1.1 targetRef: kind: Node name: nodeone - ip: 192.168.1.1 targetRef: kind: Node name: nodetwo ports: - name: http-metrics port: 9182 protocol: TCP	1081
apiVersion: v1 kind: Service metadata: labels: k8s-app: wmi-exporter name: wmi-exporter namespace: kube-system spec: clusterIP: None ports: - name: http-metrics port: 9182 protocol: TCP targetPort: 9182 sessionAffinity: None type: ClusterIP	1081
Set Up the Grafana Windows Node Dashboard	1082

Administer Prometheus and Grafana	1082
Logging Windows Worker Workloads	1082
Prerequisites	1082
Overview	1083
Prepare the Working Environment	1083
Configure Docker for Creating Windows Containers	1083
Prepare a Fluent Bit Image	1083
Build a Windows Fluent Bit Docker Image	1083
Configure Fluent Bit	1084
Install Fluent Bit	1087
Deploy Fluent Bit on the Windows Cluster	1088
Validate the Fluent Bit Deployment Using a Sample App	1088
Configure a Sample App	1088
Deploy the Sample App	1089
Validate the Fluent Bit Deployment	1089
Troubleshooting	1089
‘Cannot Find Path’ Error When Creating the Fluent Bit Docker Container	1089
Symptom	1089
Description	1089
Workaround	1090
‘The remote name could not be resolved’ Error When Creating the Fluent Bit Docker Container	1090
Symptom	1090
Description	1091
Workaround	1091
Backing Up and Restoring Tanzu Kubernetes Grid Integrated Edition	1093
Overview of Backing Up and Restoring TKGI	1093
Layers and Tools	1093
Considerations	1094
Develop a Backup and Restore Plan, and Test It Often	1094
Back Up Frequently and Regularly	1094
Back Up Critical Components	1094
Backup and Restore One Kubernetes Namespace at a Time	1094
Restore What Breaks	1094
Understand What Is Restored at Each Layer	1095
Backing Up and Restoring Tanzu Kubernetes Workloads Using Velero with Restic	1095

About Tanzu Kubernetes Workload Backup and Restore	1095
Application Types	1095
Services and Ingress	1095
Storage Types	1096
Tanzu Kubernetes Workload Backup and Restore Requirements	1096
Cluster Configuration	1096
Object Store	1097
Velero CLI	1097
Velero Version	1097
Air-Gapped Environment	1097
Get Started Using Velero and Restic	1097
Scenarios for Backing Up and Restoring TKGI Workloads	1097
Kubernetes Workload Backup and Restore Scenarios Using Velero	1097
Installing Velero and Restic	1098
Prerequisites	1098
Deploy an Object Store	1099
Install MinIO	1099
Start MinIO	1099
Enable MinIO as a Service	1100
Create MinIO Bucket	1101
Configure MinIO Bucket	1102
Install the Velero CLI on Your Workstation	1103
Download the Velero CLI Binary	1103
Install the Velero CLI	1103
Install Velero and Restic on the Target Kubernetes Cluster	1104
Prerequisites	1104
Set Up the kubectl Context	1104
Install Velero and Restic	1105
Modify the Host Path	1107
Adjust Velero Memory Limits If Necessary	1108
Install Velero and Restic in an Air-Gapped Environment	1108
Prerequisites	1109
Procedure	1109
Installing Velero vSphere Plugin	1111
Prerequisites	1111
Deploy an Object Store	1111
Install the Velero CLI on Your Workstation	1111

Download the Velero CLI Binary	1111
Install the Velero CLI	1112
Install Velero on the Target Kubernetes Cluster	1112
Prerequisites	1113
Set Up the kubectl Context	1113
Install Velero	1114
Create a Velero vSphere Credential Secret	1115
Create the Velero vSphere Plugin Configuration File	1115
Install Velero vSphere Plugin	1116
Back up the VCP Volumes Migrated to vSphere CSI Driver	1117
Adjust Velero Memory Limits If Necessary	1117
Install Velero in an Air-Gapped Environment	1118
Prerequisites	1118
Procedure	1118
 Backup and Restore Stateless App with Namespace	1119
Overview	1119
Prerequisites	1120
Deploy Guestbook App	1120
Backup the Guestbook App using Namespace	1121
Restore the Guestbook App	1124
Conclusions	1125
 Backup and Restore Stateless App with Label	1126
Overview	1126
Prerequisites	1126
Deploy Guestbook App	1126
Apply a Common Label to all App Objects	1127
Backup the Guestbook App Using Label	1128
Restore the Guestbook App Using Label	1130
Conclusions	1132
 Backup and Restore Stateful App Using Namespace	1132
Overview	1132
Prerequisites	1132
Deploy Guestbook App	1133
Backup the Guestbook App Using Namespace	1135
Restore the Guestbook App	1136
 Backup and Restore Stateful App with Label	1139

Overview	1139
Prerequisites	1140
Deploy Guestbook App	1140
Apply a Common Label to all App Objects	1141
Backup the Guestbook App Using Label	1142
Restore the Guestbook App Using Label	1144
Conclusions	1146
Backup and Restore Stateful App with Namespace (CSI Edition)	1146
Overview	1146
Prerequisites	1146
Create CSI Storage Class	1147
Deploy Guestbook App	1147
Backup the Guestbook App Using Namespace	1149
Restore the Guestbook App	1150
Conclusions	1152
Backup and Restore StatefulSet App with Namespace	1152
Overview	1152
Prerequisites	1152
Create the Storage Class	1153
Deploy CassandraDB App	1153
Create and Populate a Database and Table in Cassandra	1154
Add Annotations	1156
Backup the CassandraDB App using Namespace	1156
Restore the CassandraDB App	1157
Backup and Restore StatefulSet App with Label	1160
Overview	1160
Prerequisites	1161
Create the Storage Class	1161
Deploy CassandraDB App	1161
Create and Populate a Database and Table in Cassandra	1162
Add Annotations	1164
Backup the CassandraDB App Using Label	1164
Restore the CassandraDB App	1165
Backup and Restore StatefulSet App with Namespace	1168
Overview	1168
Prerequisites	1169

Create the Storage Class	1169
Deploy CassandraDB App	1169
Create and Populate a Database and Table in Cassandra	1170
Add Annotations	1172
Back Up the CassandraDB App Using Namespace	1172
Restore the CassandraDB App	1173
Conclusions	1175
Backup and Restore Stateful App with Static IP for Load Balancer Service	1175
Overview	1176
Prerequisites	1176
Configure Wordpress YAML Files	1176
Deploy Wordpress App	1177
Backup the Wordpress App Using Namespace	1179
Restore the Wordpress App	1180
Conclusions	1182
Backup and Restore Stateful App with Static IP for Ingress	1182
Overview	1183
Prerequisites	1183
Create a Network Profile	1183
Deploy the Coffee-Tea App	1184
Back Up the Coffee-Tea App Using Namespace	1187
Restore the Coffee-Tea App	1189
Conclusions	1194
Backup and Restore Stateful App with Static IP for Load Balancer Service (CSI Edition)	1194
Overview	1194
Prerequisites	1194
Configure Wordpress YAML Files	1195
Deploy Wordpress App	1196
Backup the Wordpress App Using Namespace	1198
Restore the Wordpress App	1199
Conclusions	1201
Backup and Restore All Cluster Workloads	1201
Overview	1202
Prerequisites	1202
Deploy Stateless Guestbook App	1202

Deploy StatefulSet CassandraDB App	1203
Create and Populate Cassandra Database	1204
Add Annotations	1205
Perform Velero Backup of the Cluster	1205
Restore All Cluster Workloads	1206
Conclusions	1209
 Backing Up and Restoring Kubernetes Clusters Provisioned by TKGI	1210
Overview	1210
Testing Considerations	1210
 Installing and Configuring BOSH Backup and Restore	1211
Overview	1211
Prerequisites	1211
Install and Configure BOSH Backup and Restore	1212
Configure Your Jumpbox for BBR	1212
Install BBR on Your Jumpbox	1212
Verify Your BBR Installation	1213
Configure BBR Logging	1213
 Backing Up Tanzu Kubernetes Grid Integrated Edition	1213
Overview	1213
Recommendations	1214
Prepare to Back Up	1214
Verify Your BBR Version	1214
Retrieve the BBR SSH Credentials	1215
Ops Manager Installation Dashboard	1215
Ops Manager API	1215
Save the BBR SSH Credentials to File	1216
Retrieve the BOSH Director Credentials	1216
Ops Manager Installation Dashboard	1216
Ops Manager API	1216
Retrieve the UAA Client Credentials	1217
Retrieve the BOSH Director Address	1217
Log In To BOSH Director	1217
Download the Root CA Certificate	1218
Retrieve the BOSH Command Line Credentials	1218
Retrieve Your Cluster Deployment Names	1218
Back Up Tanzu Kubernetes Grid Integrated Edition	1219
Connect to Your Jumpbox	1219

Connect with SSH	1220
Connect with BOSH_ALL_PROXY	1220
Back Up Kubernetes Clusters Provisioned by TKGI	1221
Verify Your Provisioned Clusters	1221
Back Up Kubernetes Clusters Provisioned by TKGI	1223
Back Up All Kubernetes Clusters	1223
Back Up a Single Kubernetes Cluster	1224
Cancel a Cluster Backup	1225
After Backing Up Tanzu Kubernetes Grid Integrated Edition	1226
Manage Your Backup Artifact	1226
Recover from a Failing Command	1226
Clean Up after a Failed Backup	1227
Restoring Kubernetes Clusters Provisioned Using the Tanzu Kubernetes Grid Integrated Edition	1228
Overview	1229
Compatibility of Restore	1229
Prepare to Restore a Backup	1229
Verify Your BBR Version	1230
Retrieve the BBR SSH Credentials	1230
Ops Manager Installation Dashboard	1230
Ops Manager API	1230
Save the BBR SSH Credentials to File	1231
Retrieve the BOSH Director Credentials	1231
Ops Manager Installation Dashboard	1231
Ops Manager API	1232
Retrieve the UAA Client Credentials	1232
Retrieve the BOSH Director Address	1232
Log In To BOSH Director	1232
Download the Root CA Certificate	1233
Retrieve the BOSH Command Line Credentials	1233
Retrieve Your Cluster Deployment Names	1233
Transfer Artifacts to Your Jumpbox	1234
Restore Kubernetes Clusters Provisioned by TKGI	1235
Redeploy Clusters	1235
Redeploy All Kubernetes Clusters	1235
Redeploy a Single Kubernetes Cluster	1235
Restore Kubernetes Clusters	1236
Register Restored Worker VMs	1237
Delete Nodes	1237

Restart kubelet	1238
Resolve a Failing BBR Restore Command	1238
Cancel a Restore	1239
Clean Up After a Failed Restore	1239
Backing Up and Restoring the TKGI Management Plane	1241
Overview	1241
Testing Considerations	1241
Installing and Configuring BOSH Backup and Restore	1242
Overview	1242
Prerequisites	1242
Install and Configure BOSH Backup and Restore	1243
Configure Your Jumpbox for BBR	1243
Install BBR on Your Jumpbox	1243
Verify Your BBR Installation	1244
Configure BBR Logging	1244
Backing Up TKGI Management Plane Components	1244
Overview	1244
Recommendations	1245
Prepare to Back Up	1245
Verify Your BBR Version	1246
Retrieve the BBR SSH Credentials	1246
Ops Manager Installation Dashboard	1246
Ops Manager API	1246
Save the BBR SSH Credentials to File	1247
Retrieve the BOSH Director Credentials	1247
Ops Manager Installation Dashboard	1247
Ops Manager API	1247
Retrieve the UAA Client Credentials	1248
Retrieve the BOSH Director Address	1248
Log In To BOSH Director	1248
Download the Root CA Certificate	1249
Retrieve the BOSH Command Line Credentials	1249
Retrieve Your Cluster Deployment Names	1249
Back Up Tanzu Kubernetes Grid Integrated Edition	1250
Connect to Your Jumpbox	1250
Connect with SSH	1251
Connect with BOSH_ALL_PROXY	1251

Back Up Ops Manager Installation Settings	1252
Export Settings Using the Ops Manager UI	1253
Export Settings Using the Ops Manager API	1253
Back Up the Tanzu Kubernetes Grid Integrated Edition BOSH Director	1253
Validate the Tanzu Kubernetes Grid Integrated Edition BOSH Director	1254
Back Up the Tanzu Kubernetes Grid Integrated Edition BOSH Director	1254
Back Up the Tanzu Kubernetes Grid Integrated Edition Control Plane	1255
Locate the Tanzu Kubernetes Grid Integrated Edition Deployment Name	1255
Validate the Tanzu Kubernetes Grid Integrated Edition Control Plane	1256
Back Up the Tanzu Kubernetes Grid Integrated Edition Control Plane	1257
Cancel a Backup	1258
After Backing Up Tanzu Kubernetes Grid Integrated Edition	1258
Manage Your Backup Artifact	1258
Recover from a Failing Command	1259
Clean Up after a Failed Backup	1259
 Restoring TKGI Management Plane Components	1261
Overview	1261
Compatibility of Restore	1262
Prepare to Restore a Backup	1262
Verify Your BBR Version	1262
Retrieve the BBR SSH Credentials	1263
Ops Manager Installation Dashboard	1263
Ops Manager API	1263
Save the BBR SSH Credentials to File	1263
Retrieve the BOSH Director Credentials	1264
Ops Manager Installation Dashboard	1264
Ops Manager API	1264
Retrieve the UAA Client Credentials	1265
Retrieve the BOSH Director Address	1265
Log In To BOSH Director	1265
Download the Root CA Certificate	1266
Retrieve the BOSH Command Line Credentials	1266
Retrieve Your Cluster Deployment Names	1266
Transfer Artifacts to Your Jumpbox	1267
Restore the BOSH Director	1267
Deploy Ops Manager	1268
Import Installation Settings	1268
(Optional) Configure Ops Manager for New Resources	1269

Remove BOSH State File	1270
Deploy the BOSH Director	1270
Restore the BOSH Director	1270
Remove All Stale Deployment Cloud IDs	1271
Restore the Tanzu Kubernetes Grid Integrated Edition Control Plane	1272
Determine the Required Stemcell	1272
Upload Stemcells	1273
Redeploy the Tanzu Kubernetes Grid Integrated Edition Control Plane	1273
Restore the TKGI Control Plane	1274
Resolve a Failing BBR Restore Command	1275
Cancel a Restore	1275
Clean Up After a Failed Restore	1275
 Backing Up and Restoring the Data Center for TKGI	1278
Overview	1278
Test Considerations	1278
 Backing Up and Restoring VMware NSX Manager	1279
NSX-T Data Center Backup and Recover	1279
Deployment Assumptions	1279
Backup Procedure	1279
Restore Procedure	1280
Testing Procedure	1280
 Backing Up and Restoring the vCenter Server	1280
vCenter Cluster Configuration	1281
vCenter Server Backup and Recover	1281
Data Protection	1281
 Tanzu Kubernetes Grid Integrated Edition Security	1282
CIS Kubernetes Benchmarks	1282
 Tanzu Kubernetes Grid Integrated Edition Security Disclosure and Release Process	1282
Security Issues in Tanzu Kubernetes Grid Integrated Edition	1282
Security Issues in Kubernetes	1282
Security Issues from CFF	1283
Security Issues in VMware NSX	1283
Security Issues in VMware Harbor	1283
 Tanzu Kubernetes Grid Integrated Edition Certificates	1283

Overview of TKGI Certificates	1283
Check Certificate Expiration Dates	1284
Rotating Certificates	1284
Rotating Tanzu Kubernetes Grid Integrated Edition Control Plane Certificates	1286
Overview	1287
Check Certificate Expiration Dates	1287
Rotate TKGI Control Plane Certificates	1287
Rotate Kubernetes Cluster Certificates	1288
Overview	1288
Procedure	1288
List TLS Certificates	1289
Rotate TLS Certificates	1289
Rotate All Cluster Certificates	1290
Rotate All Cluster Certificates Except NSX-T	1290
Rotate NSX-T Certificates Only	1290
Rotate Custom CA	1290
Rotate TLS Certificates Using the TKGI CLI	1291
Rotate VMware NSX Certificates for Kubernetes Clusters	1291
About NSX-T Certificate Rotation for Kubernetes Clusters Provisioned by TKGI	1291
List TLS Certificates Created for NSX-T	1291
Rotate the TLS Certificates for NSX-T	1293
Use a Custom CA for Kubernetes Clusters	1294
Custom CA Support	1294
Create Cluster with Custom CA	1294
Update Cluster with Custom CA	1294
Rotate Custom CA	1295
Troubleshoot Custom CA	1295
Custom CA Requirements	1295
Custom CA Formats	1296
Encrypt Secrets in an etcd Database	1296
Create Kubernetes Profile	1296
Create Kubernetes Cluster	1298
Ensure Existing Data is Encrypted	1299
Verify Your Data is Encrypted	1299
Rotate Encryption Key for Secrets in etcd Database	1301

Decrypt Secrets in the etcd Database	1306
Securing Access to the AWS Metadata Service	1307
Overview	1307
Secure Access to AWS Instance Metadata for a Namespace	1308
Deny Access from a Specific Namespace	1308
kubectl apply -f np.yml	1309
kubectl get networkpolicy	1309
Grant Access to Specific Apps in a Namespace	1309
kubectl apply -f np-allow.yml	1310
kubectl get networkpolicy	1310
Secure Access for All Namespaces Using an Antrea Cluster-Wide Network Policy	1311
Deny Access to All Namespaces Using Antrea	1311
kubectl apply -f np-cluster-deny.yml	1312
kubectl get clusternetworkpolicies.crd.antrea.io -owide	1312
Allow Access to a Specific App Using Antrea	1312
kubectl apply -f np-cluster-allow.yml	1313
kubectl get clusternetworkpolicies.crd.antrea.io -owide	1313
Diagnosing and Troubleshooting Tanzu Kubernetes Grid Integrated Edition	1315
General Troubleshooting	1315
TKGI API is Slow or Times Out	1315
All Cluster Operations Fail	1315
Cluster Creation Fails	1317
Cluster Deletion Fails	1317
Cannot Re-Create a Cluster that Failed to Deploy	1318
Windows Stemcell for vSphere Creation Fails with Login Issue	1319
Cannot Access Add-On Features or Functions	1320
Resurrecting VMs Causes Incorrect Permissions in vSphere HA	1320
Worker Node Hangs Indefinitely	1321
Cannot Authenticate to an OpenID Connect-Enabled Cluster	1323
Cannot Access Apps Deployed to Clusters That Utilize Websocket	1324
Login Failed Error: Credentials were rejected	1324

Storage Requirements for Large Numbers of Pods	1325
Login Failed Errors Due to Server State	1325
Error: Failed Jobs	1325
Error: No Such Host	1326
Error: FailedMount	1326
Error: Plan Not Found	1327
Using BOSH Diagnostic Commands in Tanzu Kubernetes Grid Integrated Edition	1327
Overview	1327
Log in to the BOSH Director VM	1327
SSH into the TKGI API VM	1328
SSH into the TKGI Database VM	1329
SSH into a Kubernetes Cluster VM	1330
View Log Files	1331
Verifying Deployment Health	1332
Verify Kubernetes Node and Pod Health	1332
Verify Kubernetes Cluster Health	1332
Retrieve Cluster Upgrade Task ID	1337
Verify NCP Health (NSX-T Only)	1337
Service Interruptions	1339
Stemcell or Service Update	1339
Impact	1339
Required Actions	1339
VM Process Failure on a Cluster Control Plane	1339
Impact	1339
Required Actions	1340
VM Process Failure on a Cluster Worker	1340
Impact	1340
Required Actions	1340
VM Process Failure on the TKGI API VM	1340
Impact	1340
Required Actions	1340
VM Failure	1341
Impact	1341
Required Actions	1341
AZ Failure	1341
Impact	1341

Required Actions	1341
Region Failure	1341
Impact	1341
Required Actions	1342
Troubleshooting Tanzu Kubernetes Grid Integrated Edition Management Console	1342
Deployment of the Tanzu Kubernetes Grid Integrated Edition Management Console Fails	1342
Deployment of Tanzu Kubernetes Grid Integrated Edition from the Management Console Fails	1342
Tanzu Kubernetes Grid Integrated Edition Management Console Cannot Retrieve Cluster Data in a Multi-Tier0 Topology	1343
TKGI MC is Unable to Upgrade the TKGI Control Plane After Migrating from N-VDS to VDS	1343
Obtain the vRealize Log Insight Agent ID for Tanzu Kubernetes Grid Integrated Edition Management Console	1345
Connect to Operations Manager	1346
Connect to Operations Manager with SSH	1346
Log In to the Operations Manager UI	1346
Using the BOSH CLI	1346
Using the BOSH CLI from the Tanzu Kubernetes Grid Integrated Edition Management Console VM	1347
Using BOSH SSH	1347
Solution Guides for Tanzu Kubernetes Grid Integrated Edition	1348
Solution Guides	1348
Solution Guides for Tanzu Kubernetes Grid Integrated Edition	1348
Solution Guides	1348
TKGI CLI	1349
Overview	1349
TKGI CLI Commands	1349
tkgi cancel-task	1349
Synopsis	1349
Examples	1350
Options	1350
tkgi certificates	1350
Synopsis	1350
Examples	1350
Options	1350

tkgi cluster	1350
Synopsis	1351
Examples	1351
Options	1351
tkgi clusters	1351
Synopsis	1351
Examples	1351
Options	1351
tkgi compute-profile	1351
Synopsis	1352
Examples	1352
Options	1352
tkgi compute-profiles	1352
Synopsis	1352
Examples	1352
Options	1352
tkgi create-cluster	1352
Synopsis	1353
Examples	1353
Options	1353
tkgi create-compute-profile	1353
Synopsis	1353
Examples	1353
Options	1354
tkgi create-kubernetes-profile	1354
Synopsis	1354
Examples	1354
Options	1354
tkgi create-network-profile	1354
Synopsis	1354
Examples	1354
Options	1355
tkgi delete-cluster	1355
Synopsis	1355
Examples	1355
Options	1355
tkgi delete-compute-profile	1355
Synopsis	1355
Examples	1355

Options	1356
tkgi delete-kubernetes-profile	1356
Synopsis	1356
Examples	1356
Options	1356
tkgi delete-network-profile	1356
Synopsis	1356
Examples	1356
Options	1357
tkgi get-credentials	1357
Synopsis	1357
Examples	1357
Options	1357
tkgi get-kubeconfig	1357
Synopsis	1358
Examples	1358
Options	1358
tkgi kubernetes-profile	1358
Synopsis	1358
Examples	1358
Options	1359
tkgi kubernetes-profiles	1359
Synopsis	1359
Examples	1359
Options	1359
tkgi login	1359
Synopsis	1359
Examples	1360
Options	1360
tkgi logout	1360
Synopsis	1360
Examples	1360
Options	1360
tkgi network-profile	1361
Synopsis	1361
Examples	1361
Options	1361
tkgi network-profiles	1361
Synopsis	1361

Examples	1361
Options	1361
tkgi plans	1362
Synopsis	1362
Examples	1362
Options	1362
tkgi promote-cluster-to-policy	1362
Synopsis	1362
Examples	1362
Options	1362
tkgi resize	1363
Synopsis	1363
Examples	1363
Options	1363
tkgi rotate-certificates	1364
Synopsis	1364
Examples	1364
Options	1364
tkgi task	1364
Synopsis	1364
Examples	1364
Options	1364
tkgi tasks	1365
Synopsis	1365
Examples	1365
Options	1365
tkgi update-cluster	1365
Synopsis	1365
Examples	1365
Options	1366
tkgi upgrade-cluster	1366
Synopsis	1366
Examples	1367
Options	1367
tkgi upgrade-clusters	1367
Synopsis	1367
Examples	1367
Options	1367

VMware Tanzu Kubernetes Grid Integrated Edition

VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) enables operators to provision, operate, and manage enterprise-grade Kubernetes clusters using BOSH and Ops Manager.

Overview

Tanzu Kubernetes Grid Integrated Edition deploys Kubernetes to [BOSH](#) and [Ops Manager](#), and uses the [On-Demand Broker](#) to dynamically instantiate, deploy, and manage highly-available Kubernetes clusters on-premises or on a public cloud.

After operators install TKGI, developers can use the TKGI Command Line Interface (TKGI CLI) to provision Kubernetes clusters, and run container-based workloads on the clusters with the Kubernetes CLI, [kubectl](#).

Operators install TKGI as a tile on the Ops Manager Installation Dashboard, or from the TKGI Management Console on vSphere.

You can run TKGI standalone or alongside VMware Tanzu Application Service for VMs on Ops Manager.

What Tanzu Kubernetes Grid Integrated Edition Adds to Kubernetes

The following table details the features that Tanzu Kubernetes Grid Integrated Edition adds to the Kubernetes platform.

Feature	Included in K8s	Included in Tanzu Kubernetes Grid Integrated Edition
Single tenant ingress	✓	✓
Secure multi-tenant ingress		✓
Stateful sets of pods	✓	✓
Multi-container pods	✓	✓
Rolling upgrades to pods	✓	✓
Rolling upgrades to cluster infrastructure		✓
Pod scaling and high availability	✓	✓
Cluster provisioning and scaling		✓

Monitoring and recovery of cluster VMs and processes		✓
Persistent disks	✓	✓
Secure container registry		✓
Embedded, hardened operating system		✓

Features

Tanzu Kubernetes Grid Integrated Edition has the following features:

- **Kubernetes compatibility:** Constant compatibility with current stable release of Kubernetes
- **Production-ready:** Highly available from applications to infrastructure, with no single points of failure
- **BOSH advantages:** Built-in health checks, scaling, auto-healing and rolling upgrades
- **Fully automated operations:** Fully automated deploy, scale, patch, and upgrade experience
- **Multi-cloud:** Consistent operational experience across multiple clouds

Tanzu Kubernetes Grid Integrated Edition Prerequisites

For information about the resource requirements for installing Tanzu Kubernetes Grid Integrated Edition, see the topic that corresponds to your cloud provider:

- [vSphere Prerequisites and Resource Requirements](#)
- [vSphere with NSX-T Version Requirements and Hardware Requirements for Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T](#)
- [GCP Prerequisites and Resource Requirements](#)
- [AWS Prerequisites and Resource Requirements](#)
- [Azure Prerequisites and Resource Requirements](#)

Release Notes

This topic contains release notes for Tanzu Kubernetes Grid Integrated Edition (TKGI) v1.16.

TKGI v1.16.0

Release Date: February 28, 2023

Product Snapshot

Release	Details	
Version	v1.16.0	
Release date	February 28, 2023	
Component	Version	
Antrea	v1.6.0*	Release Notes
cAdvisor	v0.39.1	
Containerd	Linux: v1.6.6 Windows: v1.6.6	
CoreDNS	v1.9.3+vmware.4*	
CSI Driver for vSphere	v2.7.0*	Release Notes
etcd	v3.5.6*	
Harbor	v2.6.2*	Release Notes
Kubernetes	v1.25.4*	Release Notes
Metrics Server	v0.6.1	
NCP	v4.1.0*	
Percona XtraDB Cluster (PXC)	v0.44.0	
UAA	v74.5.63*	
Velero	v1.9.5*	Release Notes
VMware Cloud Foundation (VCF)	Incompatible**	

Wavefront	Wavefront Collector: v1.12.0* Wavefront Proxy: v12.0*
Compatibilities	Versions
Ops Manager	See VMware Tanzu Network .
VMware NSX	See VMware Product Interoperability Matrices*** .
vSphere	
Windows stemcells	v2019.55* or later
Ubuntu Jammy stemcells	See VMware Tanzu Network .

* Components marked with an asterisk have been updated.

** VCF is not supported with TKGI v1.16 at this time. For more information, see [Interoperability with VMware Cloud Foundation Is Unavailable](#) below.

*** In-Tree vSphere Storage Volume support requires vSphere 7.0u2 and later. Migration from NSX Management Plane API to NSX Policy API requires VMware NSX v4.0.1.1 or later.

Upgrade Path

The supported upgrade paths to Tanzu Kubernetes Grid Integrated Edition v1.16.0 are from TKGI v1.15.2 and earlier TKGI v1.15 patches.

Breaking Changes

TKGI v1.16.0 has the following breaking changes:

- Existing Telemetry Program configuration settings are ignored and telemetry must be reconfigured.

The terms of the Telemetry & Customer Experience Improvement program have been updated. Your previous selection will be reset upon upgrading to TKGI 1.16. Review VMware's [Customer Experience Improvement Program](#), and indicate your willingness to participate in TKGI's CEIP Tab.

To reconfigure Telemetry, see [VMware CEIP](#) in the *Installing Tanzu Kubernetes Grid Integrated Edition* topic for your IaaS.

For more information on the Telemetry enhancements in this release, see [Telemetry Enhancements](#).

- Upgrades Kubernetes to v1.25:
 - Kubernetes no longer serves the following:
 - `batch/v1beta1` API version of CronJob.
 - `discovery.k8s.io/v1beta1` API version of EndpointSlice.
 - `events.k8s.io/v1beta1` API version of Event.
 - `autoscaling/v2beta1` API version of HorizontalPodAutoscaler.

- `policy/v1beta1` API version of PodDisruptionBudget.
- PodSecurityPolicy in the `policy/v1beta1` API.



Note: Pod Security Policy configurations must be migrated to Pod Security Admission security before upgrading to TKGI v1.16. For more information, see [Migrate from PSP to PSA Controller in Pod Security Admission in TKGI](#).

- RuntimeClass in the `node.k8s.io/v1beta1` API.

For more information on Kubernetes v1.25 API removals, see [Deprecated API Migration Guide - v1.25](#) in the Kubernetes documentation.

- No longer supports In-Tree vSphere Storage Volumes on vSphere 7.0u1 and earlier. Kubernetes v1.25 supports In-Tree vSphere Storage Volumes on vSphere 7.0u2 and later only.

For information about additional changes in Kubernetes v1.25, see [CHANGELOG-1.25](#) in the Kubernetes GitHub repository.

- Support for the Xenial Stemcell has been removed. For more information, see [Supports the Ubuntu Jammy Stemcell](#) below.
- The TKGI API requires a CA certificate with a SAN field:
 - The custom CA certificate used to secure TKGI API connections must include a SAN field. If the TKGI API certificate does not include a SAN field, TKGI CLI commands will return the following error:

An error occurred in the PKS API when processing

Features and Enhancements

TKGI v1.16.0 has the following features:

Telemetry Enhancements

Customers who participate in the CEIP receive proactive support benefits that include a weekly report based on telemetry data. Contact your Customer Success Manager to subscribe to this report. You can view a sample report at [TKGI Platform Operations Report](#).

vSphere CSI Driver Enhancements

TKGI v1.16.0 includes the following vSphere CSI Driver enhancements:

- Supports the snapshot and restore feature for persistent volumes. For more information, see [Customize the Maximum Number of Volume Snapshots](#).

- Supports using the vSphere Container Storage Interface (CSI) Driver on cluster worker nodes that are distributed across multiple data centers. For more information, see [Configure CNS Data Centers](#) in *Deploying and Managing Cloud Native Storage (CNS) on vSphere*.

Supports the Ubuntu Jammy Stemcell

Supports the [Ubuntu Jammy Stemcell](#).

Support for the Ubuntu Jammy Stemcell replaces support for the Xenial Stemcell. TKGI Kubernetes cluster node VMs and the TKGI Control Plane now use the Ubuntu Jammy Stemcell.

Upgrading an existing TKGI cluster to TKGI v1.16 automatically switches the cluster to the Ubuntu Jammy Stemcell.

You must import a supported Ubuntu Jammy Stemcell before upgrading TKGI to TKGI v1.16.0 or later. For more information, see [Download and Import Stemcells](#) in *Upgrading Tanzu Kubernetes Grid Integrated Edition*.

Compatible with Ops Manager v3.0

TKGI v1.16 supports Ops Manager v3.0 and Ops Manager v2.10. For more information about the new features and improvements in Ops Manager v3.0, see [Ops Manager v3.0 Release Notes](#) in the Ops Manager documentation.

For information about the Ops Manager v3.0 and v2.10 patch release versions supported by TKGI v1.16.0, see [Product Snapshot](#) above.

Supports Migrating TKGI to NSX Policy API

Supports promoting TKGI and TKGI Kubernetes clusters and workloads from NSX Management Plane API to NSX Policy API on vSphere with VMware NSX v4.0.1.1 or later.

For more information, see [Migrating the NSX Management Plane API to NSX Policy API - Overview](#).

Supports the Velero vSphere Plugin

Supports backing up and restoring TKGI and TKGI Kubernetes clusters on vSphere using the Velero vSphere Plugin.

For more information, see [Installing Velero vSphere Plugin](#).

vSphere CSI Supports Multiple Data Centers

Supports using the vSphere Container Storage Interface (CSI) Driver on cluster worker nodes that are distributed across multiple data centers. For more information, see [Configure CNS Data Centers](#)

in *Deploying and Managing Cloud Native Storage (CNS) on vSphere*.

Additional Features

TKGI v1.16.0 includes the following additional features:

- Supports nested LDAP group searching and configuring search depth. For more information, see **Group Max Search Depth** configuration in [Integrate UAA with an LDAP Server](#) in [Connecting Tanzu Kubernetes Grid Integrated Edition to an LDAP Server](#) or [Use an External LDAP Server](#) in [Deploy TKGI by Using the Configuration Wizard](#).
- Supports adding new node subnets to an existing Network Profile Node Network IP Block configuration. For more information, see [Configurable Node Network IP Blocks](#) in [Customize Node Networks](#).
- Supports running the vROPs cAdvisor daemonset without Privileged permission. No longer requires that cAdvisor be run with Privileged permission. See [Interoperability with VMware Aria Operations Management Pack for Kubernetes Is Unavailable](#) below for more information about VMware Aria interoperability.
- Enables the VMware vSphere Container Storage Plug-in ability to suspend a specific datastore for volume provisioning using Cloud Native Storage Manager. For more information, see [VMware vSphere Container Storage Plug-in 2.5 Release Notes](#).
- Component bumps:
 - Upgrades the `fluent-plugin-vmware-loginsight` Fluentd output plugin to [v1.3.1](#). `fluent-plugin-vmware-loginsight` forwards logs to VMware Log Insight.

Resolved Issues

TKGI v1.16.0 has the following resolved issues:

- **[Security Fix]** Component bumps fix the following:
 - Upgrades Kubernetes to v1.25.4:
 - Fixes [CVE-2021-25749](#): runAsNonRoot logic bypass for Windows containers.
 - Upgrades Spring to v2.5.14:
 - Fixes `spring-security` CVEs: [CVE-2021-22112](#), [CVE-2022-22976](#), [CVE-2022-22978](#).
 - Fixes `sprint-framework` CVEs: [CVE-2022-22950](#), [CVE-2022-22965](#), [CVE-2022-22968](#), [CVE-2022-22970](#).
 - Fixes `tomcat` CVEs: [CVE-2020-9484](#), [CVE-2020-17527](#), [CVE-2021-24122](#), [CVE-2021-25122](#), [CVE-2021-25329](#), [CVE-2021-30640](#), [CVE-2021-33037](#), [CVE-2021-41079](#), [CVE-2022-23181](#), [CVE-2022-29885](#), [CVE-2022-34305](#).
- Component bumps fix the following Known Issues:
 - Fluent Bit:
 - Fixes [Fluent Bit Does Not Merge Containerd Runtime Cluster Multi-Line Entries](#).

- ◊ NCP:
 - Fixes Misconfigured Ingress Invalidates All Ingresses.
- ◊ CSI Driver for vSphere to v2.7.0:
 - Fixes Persistent Volumes Fail to Detach from Nodes.
- Fixes Pods on Clusters Using the containerd-Runtime Enter a CrashLoopBackOff State.
- Fixes Timeout While Switching Container Runtimes If the Docker Directory Is Too Large.
- Fixes Windows Worker Nodes Are Unresponsive after Update-Cluster and Upgrade-Cluster.
- Fixes MP2P Migration does not migrate a cluster's SSL Profile if the cluster's Network Profile is configured to use the default SSL Profile `nsx-default-client-ssl-profile`.
- Fixes Telegraf In-Host Monitoring Does Not Collect kubelet Metrics.
- Fixes Updating and Upgrading a Cluster Can Timeout While Stopping Containerd.
- Fixes CSI Driver Image Missing After High Disk Utilization.
- Fixes Switching Your Default CNI to Antrea is Not Supported.

Deprecations

The following TKGI features have been deprecated or removed from TKGI v1.16:

- **Google Cloud Platform:** Support for the Google Cloud Platform (GCP) is deprecated. Support for GCP will be entirely removed in a future TKGI version.
- **The `log_dropped_traffic` CNI Configuration parameter:** In TKGI v1.16.0 and later, the `log_dropped_traffic` CNI Configuration parameter is ignored. To configure logging in a Network Profile, modify the `log_firewall_traffic` parameter. For more information, see `log_settings` in the `cni_configurations` Parameters section in *Creating and Managing Network Profiles*.
- **Pod Security Policy Support:** Support for Kubernetes Pod Security Policy (PSP) has been entirely removed in Kubernetes v1.25. Kubernetes v1.25 instead supports Pod Security Admission. For more information, see [Enabling and Configuring Pod Security Admission](#).
- **Flannel Support:** Support for the Flannel Container Networking Interface (CNI) is deprecated. VMware recommends that you switch your Flannel CNI-configured clusters to the Antrea CNI. For more information about Flannel CNI deprecation, see [About Switching from the Flannel CNI to the Antrea CNI](#) in *About Tanzu Kubernetes Grid Integrated Edition Upgrades*.
- **In-Tree vSphere Storage Volume Support:** In-Tree vSphere Storage volume support has been deprecated and will be entirely removed in a future Kubernetes version. The TKGI v1.17 upgrade will automatically migrate TKGI clusters from in-tree vSphere storage to vSphere CSI. VMware strongly recommends that you migrate your in-tree vSphere storage volumes to vSphere CSI volumes as soon as possible. For information on how to manually migrate In-Tree vSphere Storage volumes on existing TKGI clusters from In-Tree vSphere Storage to the automatically installed vSphere CSI Driver, see [Migrate an In-Tree vSphere](#)

[Storage Volume to the vSphere CSI Driver](#) in *Deploying and Managing Cloud Native Storage (CNS) on vSphere*.

Known Issues

TKGI v1.16.0 has the following known issues.

TKGI MC Unable to Manage TKGI after Restoring the TKGI Control Plane from Backup

Symptom

After you restore Ops Manager and the TKGI API VM from backup, TKGI functions normally, but your TKGI MC tabs include the following error: "...product 'pivotal-container service' is not deployed...".

Explanation

TKGI MC is associated with an Ops Manager with a specific name. If you rename Ops Manager with a new name while restoring, your TKGI MC will not recognize the restored Ops Manager and cannot manage it.

Kubernetes Pods on NSX-T Become Stuck in a Creating State

Symptom

The pods in your TKGI Kubernetes clusters on NSX-T become stuck in a creating state. The connections between nsx-node-agent and hyperbus repeatedly close, log `Couldn't connect to 'tcp://...'` (`error: 111-Connection refused`), and have a status of `COMMUNICATION_ERROR`.

Explanation

For information and workaround steps for this Known Issue, see [Issue 2795268: Connection between nsx-node-agent and hyperbus flips and Kubernetes pod is stuck at creating state in NSX Container Plugin 3.1.2 Release Notes](#) in the VMware documentation.

Error: Could Not Execute “Apply-Changes” in Azure Environment

Symptom

After clicking **Apply Changes** on the TKGI tile in an Azure environment, you experience an error '... could not execute "apply-changes"...' with either of the following descriptions:

- `{"errors": {"base": ["undefined method 'location' for nil:NilClass"]}}`
- `FailedError.new("Resource Groups in region '#{location}' do not support Availability Zones")`

For example:

```
INFO | 2020-09-21 03:46:49 +0000 | Vessel::Workflows::Installer#run | Install product
(apply changes)
2020/09/21 03:47:02 could not execute "apply-changes": installation failed to trigger:
request failed: unexpected response from /api/v0/installations:
HTTP/1.1 500 Internal Server Error
```

```

Transfer-Encoding: chunked
Cache-Control: no-cache, no-store
Connection: keep-alive
Content-Type: application/json; charset=utf-8
Date: Mon, 21 Sep 2020 17:51:50 GMT
Expires: Fri, 01 Jan 1990 00:00:00 GMT
Pragma: no-cache
Referrer-Policy: strict-origin-when-cross-origin
Server: Ops Manager
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Content-Type-Options: nosniff
X-Download-Options: noopen
X-Frame-Options: SAMEORIGIN
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: f5fc99c1-21a7-45c3-7f39
X-Runtime: 9.905591
X-Xss-Protection: 1; mode=block

44
{"errors": {"base": ["undefined method `location' for nil:NilClass"]}}
0

```

Explanation

The Azure CPI endpoint used by Ops Manager has been changed and your installed version of Ops Manager is not compatible with the new endpoint.

Workaround

Run the following Ops Manager CLI command:

```
om --skip-ssl-validation --username USERNAME --password PASSWORD --target https://OPSMAN-API curl --silent --path /api/v0/staged/director/verifiers/install_time/IaaSConfigurationVerifier -x PUT -d '{ "enabled": false }'
```

Where:

- **USERNAME** is the account to use to run Ops Manager API commands.
- **PASSWORD** is the password for the account.
- **OPSMAN-API** is the IP address for the Ops Manager API

For more information, see [Error ‘undefined method location’ is received when running Apply Change on Azure](#) in the VMware Tanzu Knowledge Base.

VMware vRealize Operations Does Not Support Windows Worker-Based Kubernetes Clusters

VMware vRealize Operations (vROPs) does not support Windows worker-based Kubernetes clusters and cannot be used to manage TKGI-provisioned Windows workers.

TKGI Wavefront Requires Manual Installation for Windows Workers

To monitor Windows-based worker node clusters with a Wavefront collector and proxy, you must first install Wavefront on the clusters manually, using Helm. For instructions, see the [Wavefront](#)

section of the *Monitoring Windows Worker Clusters and Nodes* topic.

Pinging Windows Worker Kubernetes Clusters Does Not Work

TKGI-provisioned Windows worker-based Kubernetes clusters inherit a Kubernetes limitation that prevents outbound ICMP communication from workers. As a result, pinging Windows workers does not work.

For information about this limitation, see [Limitations > Networking](#) in the *Windows in Kubernetes* documentation.

Velero Does Not Support Backing Up Stateful Windows Workloads

You can use Velero to back up stateless TKGI-provisioned Windows workers only. You cannot use Velero to back up stateful Windows applications. For more information, see [Velero on Windows](#) in *Basic Install* in the Velero documentation.

Tanzu Mission Control Integration Not Supported on GCP

TKGI on Google Cloud Platform (GCP) does not support Tanzu Mission Control (TMC) integration, which is configured in the **Tanzu Kubernetes Grid Integrated Edition** tile > the **Tanzu Mission Control** pane.

If you intend to run TKGI on GCP, skip this pane when configuring the Tanzu Kubernetes Grid Integrated Edition tile.

TMC Data Protection Feature Requires Privileged TKGI Containers

TMC Data Protection feature supports privileged TKGI containers only. For more information, see [Plans](#) in the *Installing TKGI* topic for your IaaS.

Windows Worker Kubernetes Clusters with Group Managed Service Account Do Not Support Compute Profiles

Windows worker-based Kubernetes clusters integrated with group Managed Service Account (gMSA) cannot be managed using compute profiles.

Windows Worker Kubernetes Clusters on Flannel Do Not Support Compute Profiles

On vSphere with NSX-T networking you can use compute profiles with both Linux and Windows worker-based Kubernetes clusters. On vSphere with Flannel networking, you can apply compute profiles only to Linux clusters.

TKGI CLI Does Not Prevent Reducing the Control Plane Node Count

TKGI CLI does not prevent accidentally reducing a cluster's control plane node count using a compute profile.



Warning: Reducing a cluster's control plane node count can destroy the cluster. Do not scale out or scale in existing control plane nodes by reconfiguring the TKGI tile or by using a compute profile. Reducing a cluster's number of control plane nodes might remove a control plane node and cause the cluster to become inactive.

Windows Cluster Nodes Not Deleted After VM Deleted

Symptom

After you delete a VM using the management console of your infrastructure provider, you notice a Windows worker node that had been on that VM is now in a `notReady` state.

Solution

1. To identify the leftover node:

```
kubectl get no -o wide
```

2. Locate nodes on the returned list that are in a `notReady` state and have the same IP address as another node in the list.
3. To manually delete a `notReady` node:

```
kubectl delete node NODE-NAME
```

Where `NODE-NAME` is the name of the node in the `notReady` state.

502 Bad Gateway After OIDC Login

Symptom

You experience a “502 Bad Gateway” error from the NSX load balancer after you log in to OIDC.

Explanation

A large response header has exceeded your NSX-T load balancer maximum response header size. The default maximum response header size is 10,240 characters and should be resized to 50,000.

Workaround

If you experience this issue, manually reconfigure your NSX-T `request_header_size` and `response_header_size` to 50,000 characters. For information about configuring NSX-T default header sizes, see [OIDC Response Header Overflow](#) in the Knowledge Base.

Difficulty Changing Proxy for Windows Workers

You must configure a global proxy in the Tanzu Kubernetes Grid Integrated Edition tile > **Networking** pane before you create any Windows workers that use the proxy.

You cannot change the proxy configuration for Windows workers in an existing cluster.

Character Limitations in HTTP Proxy Password

For vSphere with NSX-T, the HTTP Proxy password field does not support the following special characters: & or ;.

Error After Modifying Your Harbor Storage Configuration

Symptom

You receive the following error after modifying your existing Harbor installation's storage configuration:

```
Error response from daemon: manifest for ... not found: manifest unknown: manifest unk  
nown
```

Explanation

Harbor does not support modifying an existing Harbor installation's storage configuration.

Workaround

To modify your Harbor storage configuration, re-install Harbor. Before starting Harbor, configure the new Harbor installation with the desired configuration.

Ingress Controller Statefulset Fails to Start After Resizing Worker Nodes

Symptom

Permissions are removed from your cluster's files and processes after resizing the persistent disk during a cluster upgrade. The ingress controller statefulset fails to start.

Explanation

When resizing a persistent disk, Bosh migrates the data from the old disk to the new disk but does not copy the files' extended attributes.

Workaround

To resolve the problem, complete the steps in [Ingress controller statefulset fails to start after resize of worker nodes with permission denied]
(https://community.pivotal.io/s/article/5000e00001nCJxT1603094435795?language=en_US) in the VMware Tanzu Knowledge Base.

Azure Default Security Group Is Not Automatically Assigned to Cluster VMs

Symptom

You experience issues when configuring a load balancer for a multi-control plane node Kubernetes cluster or creating a service of type [LoadBalancer](#). Additionally, in the Azure portal, the **VM > Networking** page does not display any inbound and outbound traffic rules for your cluster VMs.

Explanation

As part of configuring the Tanzu Kubernetes Grid Integrated Edition tile for Azure, you enter **Default Security Group** in the **Kubernetes Cloud Provider** pane. When you create a Kubernetes cluster,

Tanzu Kubernetes Grid Integrated Edition automatically assigns this security group to each VM in the cluster. However, on Azure the automatic assignment might not occur.

As a result, your inbound and outbound traffic rules defined in the security group are not applied to the cluster VMs.

Workaround

If you experience this issue, manually assign the default security group to each VM NIC in your cluster.

One Plan ID Longer than Other Plan IDs

Symptom

One of your plan IDs is one character longer than your other plan IDs.

Explanation

In TKGI, each plan has a unique plan ID. A plan ID is normally a UUID consisting of 32 alphanumeric characters and 4 hyphens. However, the **Plan 4** ID consists of 33 alphanumeric characters and 4 hyphens.

Solution

You can safely configure and use **Plan 4**. The length of the **Plan 4** ID does not affect the functionality of **Plan 4** clusters.

If you require all plan IDs to have identical length, do not activate or use **Plan 4**.

Database Cluster Stops After a Database Instance is Stopped

Symptom

After you stop one instance in a multiple-instance database cluster, the cluster stops, or communication between the remaining databases times out, and the entire cluster becomes unreachable.

The following might be in your UAA log:

```
WSREP has not yet prepared node for application use
```

Explanation

The database cluster is unable to recover automatically because a member is no longer available to reconcile quorum.

Velero Back Up Fails for vSphere PVs Attached to Clusters on Kubernetes v1.20 and Later

Symptom

Backing up vSphere persistent volumes using Velero fails and your Velero backup log includes the following error:

```
rpc error: code = Unknown desc = Failed during IsObjectBlocked check: Could not translate selfLink to CRD name
```

Explanation

This is a known issue when backing up clusters on Kubernetes v1.20 and later using the Velero Plugin for vSphere v1.1.0 or earlier.

Workaround

To resolve the problem, complete the steps in [Velero backups of vSphere persistent volumes fail on Kubernetes clusters version 1.20 or higher \(83314\)](#) in the VMware Tanzu Knowledge Base.

Creating Two Windows Clusters at the Same Time Fails

Symptom

The first time that you try to create two Windows clusters at the same time, the creation of one of the clusters fails. If you run `pks cluster CLUSTER-NAME` to examine the last action taken on the cluster, you see the following:

```
Last Action: Create Last Action State: failed Last Action Description: Instance provisioning failed: There was a problem completing your request. ... operation: create, error-message: Failed to acquire lock ... locking task id is 11, description: 'create deployment'
```

Explanation

This is a known issue that occurs the first time that you create two Windows clusters concurrently.

Workaround

Recreate the failed cluster. This issue only occurs the first time that you create two Windows clusters concurrently.

Deleted Clusters are Listed in Cluster Lists

Symptom

After running `tkgi delete-cluster` and cluster deletion has completed, the deleted cluster continues to be listed when running `tkgi clusters`.

Workaround

You must manually remove the deleted cluster using a customized version of the ncp_cleanup script. For more information, see [Deleting a Tanzu Kubernetes Grid Integrated Edition cluster with “tkgi delete-cluster” stuck “in progress” status](#) in the VMware Tanzu Knowledge Base.

BOSH Director Logs the Error ‘Duplicate vm extension name’

Symptom

After you uninstall TKGI, then reinstall TKGI in the same environment, BOSH Director logs errors similar to the following:

```
.../gems/bosh-director-0.0.0/lib/bosh/director/deployment_plan/cloud_manifest_parser.r
b:120:in `parse_vm_extensions': Duplicate vm extension name 'disk_enable_uuid' (Bosh::
Director::DeploymentDuplicateVmExtensionName)
```

Explanation

The `pivotal-container-service` cloud-config was not removed when you uninstalled the TKGI tile, and it remained active. When you reinstalled the TKGI tile, an additional `pivotal-container-service` cloud-config was created, causing the `metrics_server` to fall into a crash-loop state.

Workaround

You must manually remove the `pivotal-container-service` cloud-config after removing your TKGI deployment, including after removing the TKGI tile from Ops Manager.

For more information, see “[Duplicate vm extension name” error when metrics_server runs on Director VM in Tanzu Kubernetes Grid Integrated Edition](#) in the VMware Tanzu Community Knowledge Base.

The TKGI API FQDN Must Not Include Trailing Whitespace

Symptom

Your TKGI logs include the following error:

```
'uaa'. Errors are:- Error filling in template 'uaa.yml.erb' (line 59: Client redirect-
uri is invalid: uaa.clients.pks_cli.redirect-uri Client redirect-uri is invalid: uaa.c
lients.pks_cluster_client.redirect-uri)
```

Explanation

The TKGI API fully-qualified domain name (FQDN) for your cluster contains leading or trailing whitespace.

Workaround

Do not include whitespace in the TKGI tile **API Hostname (FQDN)** field.

TMC Cluster Data Protection Backup Fails After Upgrading TKGI

The TMC Cluster Data Protection Backup fails in TKGI environments upgraded from an earlier version.

Symptom

The TMC Cluster Data Protection Backup fails to back up your existing clusters and logs the following error:

```
error executing custom action (groupResource=customresourcedefinitions.apiextensions.k
8s.io, namespace=, name=ncpconfigs.nsx.vmware.com): rpc error: code = e
rror fetching v1beta1 version of ncpconfigs.nsx.vmware.com: the server could not find
the requested resource
```

Explanation

Kubernetes v1.22 disallows the `spec.preserveUnknownFields: true` configuration in your existing

clusters and the creation of a v1 CustomResourceDefinitions configuration fails.

TMC Cluster Data Protection Restore Fails When Using Antrea CNI

The TMC Cluster Data Protection Restore operation can fail when restoring multiple Antrea resources.

Symptom

The TMC Cluster Data Protection Restore fails and logs errors that requests to restore the `admission webhook` have been denied.

Explanation

Velero has encountered a race condition while operating a resource. For more information, see [Allow customizing restore order for Kubernetes controllers and their managed resources](#) in the Velero GitHub repository.

TKGI Does Not Support CVDS / NVDS Mixed Environments

TKGI does not support environments where there are multiple matching networks, such as a mixed CVDS/NVDS environment.

Symptom

TKGI logs errors similar to the following in an environment with multiple matching networks:

```
LastOperationstatus='failed', description='Instance provisioning failed:  
There was a problem completing your request. Please contact your operations team providing the following information:  
service: p.pks, service-instance-guid: ..., broker-request-id: ..., task-id: ..., operation: create,  
error-message: Unknown CPI error 'Unknown' with message 'undefined method `mob' for <VmSdk::Vim::OpaqueNetwork:' in create_vm' CPI method'
```

Explanation

TKGI cannot identify which of the matching networks you intend to use and has selected the wrong network.

Occasionally `update-cluster` Does Not Complete for Windows Workers

Occasionally, `tkgi update-cluster` hangs while updating a Windows worker node instance and the BOSH task cannot finish and exits.

Symptom

The `ovsdb-server` service has stopped but other processes report that it is running.

Explanation

The `ovsdb-server.pid` file uses the pid for a process that is not the ovsdb-server.

To confirm that this is the root cause for `tkgi update-cluster` to hang:

- To verify the `ovsdb-server` service has actually stopped, run the PowerShell `Get-services` command on the Windows worker node.

- To verify that other processes report the `ovsdb-server` service is still running:

1. Review the `ovsdb-server job-service-wrapper.err.log` log file.

The `job-service-wrapper.err.log` log file is located at:

```
C:\var\vcap\sys\log\openvswitch-windows\ovsdb-server\job-service-wrapper.err.log
```

2. Confirm that after the flushing processes, the log includes an error similar to the following:

```
Pid-Guard : ovsdb-server is already running, please stop it first
At C:\var\vcap\jobs\openvswitch-windows\bin\ovsdb-server_ctl.ps1:30 char:
5
+     Pid-Guard $PIDFILE "ovsdb-server"
+     ~~~~~~
+ CategoryInfo          : NotSpecified: ( [Write-Error], WriteErrorEx
ception
+ FullyQualifiedErrorId : Microsoft.PowerShell.Commands.WriteErrorExc
ption,Pid-Guard
```

- To verify the root cause:

1. Run the following PowerShell commands on the Windows worker node:

```
$RUN_DIR = "C:\var\vcap\sys\run\openvswitch-windows"
$PIDFILE = "$RUN_DIR\ovsdb-server.pid"
$pid1 = Get-Content $PidFile -First 1
echo $pid1
$rst = Get-Process -Id $pid1 -ErrorAction SilentlyContinue
echo $rst
```

2. Confirm the returned `ProcessName` is not `ovsdb-server`.

Workaround

To resolve this issue for a single Windows worker:

1. SSH to the affected worker node.
2. Run the following:

```
rm C:\var\vcap\sys\run\openvswitch-windows\ovsdb-server.pid
```

3. Wait for the `ovsdb-server` process to start.
4. Confirm the dependent services also start.

Harbor Private Projects Are Inaccessible after Upgrading to TKGI v1.13.0

If LDAP is enabled, Harbor private projects are inaccessible after upgrading to TKGI v1.13.0. For more information, see [Private projects become inaccessible after upgrading Harbor for TKGI to v2.4.x with LDAP feature enabled](#) in the VMware Tanzu Knowledge Base.

Deployments Fail on TKGI Windows Worker-based Kubernetes Clusters after the January 2022 Microsoft Windows Security Patch

Microsoft changed Microsoft Windows' support for tar file commands in the January 2022 Microsoft Windows security patch.

Packaging scripts that use tar commands for Windows worker-based Kubernetes Cluster deployments can fail after the Microsoft tar command patch update has been applied.

The BOSH agent used by vSphere stemcells built by stembuild v2019.43 and earlier use tar commands that are no longer supported and will fail if the Microsoft Windows security patch has been applied.

Workaround

stembuild v2019.44 and later include a version of the BOSH agent that does not use unsupported tar commands.

If you use vSphere stemcells, use stembuild 2019.44 or later to avoid the BOSH agent tar error.

TKGI Reallocates Network Profile-Allocated FIP Pool Addresses

Symptom

The pre-start script for `tkgi create-cluster` fails and logs floating IP pool allocation errors in the `pre-start.stderr.log` similar to the following:

```
level=error msg="operation failed with [POST /pools/ip-pools/{pool-id}][409] allocateOrReleaseFromIpPoolConflict &{RelatedAPIError:{Details: ErrorCode:5141 ErrorData:<nil> ErrorMessage:Requested IP Address ... is already allocated. ModuleName:id-allocation service} RelatedErrors:[]}\n"

level=warning msg="failed to allocate FIP from (pool: ...: [POST /pools/ip-pools/{pool-id}][409] allocateOrReleaseFromIpPoolConflict &{RelatedAPIError:{Details: ErrorCode: 5141 ErrorData:<nil> ErrorMessage:Requested IP Address ... is already allocated. ModuleName:id-allocation service} RelatedErrors:[]})\n"

Error: an error occurred during FIP allocation
```

Explanation

TKGI administrators can allocate floating IP pool IP Addresses in a Network Profile configuration. The TKGI control plane allocates IP Addresses from the floating IP pool without accounting for the IPs allocated using Network Profiles.

Workaround

TKGI allocates IP addresses starting from the beginning of a floating IP pool range. When configuring a Network Profile, allocate IP Addresses starting at the end of the floating IP pool range instead of those at the beginning.

TKGI Clusters Fail after NSX Upgrade If They Use NSGroup Policy API Resources

TKGI supports clusters that use NSGroup Policy API resources, but Policy API NSGroups created in

one NSX version will be empty after upgrading NSX to a newer version.

Workaround

BOSH reconfigures a deployment's NSGroup members if the deployment is redeployed.

After upgrading NSX, redeploy affected deployments to reconfigure their NSGroup members:

1. **Re-Apply Changes** on the Ops Manager UI to redeploy TKGI tile deployments.
 2. Re-deploy the affected cluster deployments.
-

Kubernetes API Server and etcd Daemon Occasionally Fail to Start During BBR Restore

The Kubernetes API server or the etcd daemon on a cluster control plane node might not start during a BBR restore, stopping the restore.

Symptom

During a BBR restore, the `post-restore-unlock` script occasionally times out while starting the etcd daemon or Kubernetes API server.

For example, the `post-restore-unlock` script shows the following when the etcd daemon fails to start:

```
Error attempting to run post-restore-unlock for job bbr-etcd on master...
+ NAME=post-restore-unlock
+ LOG_DIR=/var/vcap/sys/log/bbr-etcd
+ exec
++ tee -a /var/vcap/sys/log/bbr-etcd/post-restore-unlock.stdout.log
...
monit has started etcd
+ timeout 1200 /bin/bash
waiting for etcd daemon to start
Process 'etcd'      not monitored - start pending
...
waiting for etcd daemon to start
Process 'etcd'      initializing
etcd daemon was unable to start after 1200 seconds
+ exit 1 - exit code 1
```

Workaround

Restart the BBR restore if the Kubernetes API server or the etcd daemon fails to start.

'Input not an X.509 certificate' When Applying Change on the TKGI Tile

The TKGI tile might report an error similar to the following when [Applying Changes](#) with a correctly formatted certificate.

```
Setting up key store, trust store and installing certs.
keytool error: java.lang.Exception: Input not an X.509 certificate
pre-start.stdout.log
```

Explanation

The certificate contains one or more certificate keywords, for example, `BEGIN` or `END`, and does not

validate.

Interoperability with VMware Aria Operations Management Pack for Kubernetes Is Unavailable

Interoperability with VMware Aria Operations Management Pack for Kubernetes is temporarily unavailable.

VMware Aria Operations Management Pack for Kubernetes is currently not compatible with TKGI v1.16. Interoperability between VMware Aria Operations Management Pack for Kubernetes and TKGI v1.16 is expected at a later time.

Interoperability with VMware Cloud Foundation Is Unavailable

Interoperability with VMware Cloud Foundation (VCF) is temporarily unavailable.

VCF is currently not compatible with TKGI v1.16. Interoperability between VCF and TKGI v1.16 is expected at a later time.

Interoperability with Tanzu Mission Control is Unavailable

Interoperability with Tanzu Mission Control (TMC) is temporarily unavailable.

TMC is currently not compatible with Kubernetes v1.25 and cannot manage TKGI v1.16 Kubernetes clusters. Interoperability between TMC and TKGI v1.16 is expected at a later time.

Limitations on Using the VMware vSphere CSI Driver

The VMware vSphere CSI Driver supports a limited set of VMware vSphere features. Before enabling the vSphere CSI Driver on a TKGI cluster, confirm the cluster and storage configuration are supported by the driver. For more information, see [Unsupported Features and Limitations in Deploying and Managing Cloud Native Storage \(CNS\) on vSphere](#).

Limitations on Using a Public Cloud CSI Driver

If you enable a public cloud CSI Driver on a TKGI cluster, you must take additional steps before deleting, upgrading, or updating the cluster.

Deleting a Cluster on a Public Cloud

When deleting a cluster that uses a public cloud CSI Driver:

1. Manually delete the workload PVCs and PVs before deleting the cluster.
2. Delete the cluster. For more information on deleting clusters, see [Deleting Clusters](#).

Upgrading a Cluster on a Public Cloud

When upgrading a cluster that uses a public cloud CSI Driver:

- No preparation steps are needed when upgrading a multi-worker node cluster.
- To prepare a single-worker node cluster for upgrading:

1. Resize the cluster to two or more worker nodes before upgrading the cluster. For more information, see [Scaling Existing Clusters](#).
2. Upgrade the cluster. For more information on upgrading clusters, see [Upgrading Clusters](#).

Updating a Cluster on a Public Cloud

When updating a cluster that uses a public cloud CSI Driver:

- No preparation step are needed when updating a multi-worker node cluster.
- To prepare a single-worker node cluster for updating:
 1. Resize the cluster to two or more worker nodes before updating the cluster. For more information, see [Scaling Existing Clusters](#).
 2. Update the cluster.

The ‘kube-state-metrics’ ClusterRole Is Deleted during Cluster Upgrade

The `wavefront-proxy-errand` deletes the `kube-state-metrics` ClusterRole during cluster upgrade. The deleted ClusterRole must be manually restored after upgrading a cluster.

TKGI Management Console v1.16.0

Release Date: February 28, 2023



Note: Tanzu Kubernetes Grid Integrated Edition Management Console provides an opinionated installation of TKGI. The supported versions might differ from or be more limited than what is generally supported by TKGI.

Product Snapshot

Element	Details	
Version	v1.16.0	
Release date	February 28, 2023	
Installed TKGI version	v1.16.0	
Installed Ops Manager version	v2.10.53	Release Notes
Component	Version	
Installed Kubernetes version	v1.25.4*	Release Notes
Installed Harbor Registry version	v2.6.2*	Release Notes

Ubuntu Jammy stemcell	v1.83*	Release Notes
-----------------------	--------	-------------------------------

* Components marked with an asterisk have been updated.

Upgrade Path

The supported upgrade paths to Tanzu Kubernetes Grid Integrated Edition Management Console v1.16.0 are from TKGI MC v1.15.2 and earlier TKGI v1.15 patches.

Breaking Changes

- Existing Telemetry Program configuration settings are ignored and telemetry must be reconfigured.
To reconfigure Telemetry, see [VMware CEIP](#) in the *Installing Tanzu Kubernetes Grid Integrated Edition* topic for your IaaS.
For more information on the Telemetry enhancements in this release, see [Telemetry Enhancements](#).

Features and Resolved Issues

TKGI Management Console v1.16.0 has the following features:

Telemetry Enhancements

Customers who participate in the CEIP receive proactive support benefits that include a weekly report based on telemetry data. Contact your Customer Success Manager to subscribe to this report. You can view a sample report at [TKGI Platform Operations Report](#).

Deprecations

The following TKGI features have been deprecated or removed from TKGI Management Console v1.16:

Known Issues

The Tanzu Kubernetes Grid Integrated Edition Management Console v1.16.0 has the following known issues:

vRealize Log Insight Integration Does Not Support HTTPS Connections

Symptom

The Tanzu Kubernetes Grid Integrated Edition Management Console integration to vRealize Log

Insight does not support connections to the HTTPS port on the vRealize Log Insight server.

Workaround

1. Use SSH to log in to the Tanzu Kubernetes Grid Integrated Edition Management Console appliance VM.
2. Open the file `/lib/systemd/system/pks-loginsight.service` in a text editor.
3. Add `-e LOG_SERVER_ENABLE_SSL_VERIFY=false`.
4. Set `-e LOG_SERVER_USE_SSL=true`.

The resulting file should look like the following example:

```
ExecStart=/bin/docker run --privileged --restart=always --network=pks
-v /var/log/journal:/var/log/journal
--name=pks-loginsight
-e TYPE=gear2-vm
-e LOG_SERVER_HOST=${LOGINSIGHT_HOST}
-e LOG_SERVER_PORT=${LOGINSIGHT_PORT}
-e LOG_SERVER_ENABLE_SSL_VERIFY=false
-e LOG_SERVER_USE_SSL=true
-e LOG_SERVER_AGENT_ID=${LOGINSIGHT_ID}
pksoctopus/vrli-journald:v07092019
```

5. Save the file and run `systemctl daemon-reload`.
6. To restart the vRealize Log Insight service, run `systemctl restart pks-loginsight.service`.

Tanzu Kubernetes Grid Integrated Edition Management Console can now send logs to the HTTPS port on the vRealize Log Insight server.

vSphere HA causes Management Console ovfenv Data Corruption

Symptom

If you enable vSphere HA on a cluster, if the TKGI Management Console appliance VM is running on a host in that cluster, and if the host reboots, vSphere HA recreates a new TKGI Management Console appliance VM on another host in the cluster. Due to an issue with vSphere HA, the `ovfenv` data for the newly created appliance VM is corrupted and the new appliance VM does not boot up with the correct network configuration.

Workaround

1. In the vSphere Client, right-click the appliance VM and select **Power > Shut Down Guest OS**.
2. Right-click the appliance again and select **Edit Settings**.
3. Select **VM Options** and click **OK**.
4. Verify under Recent Tasks that a `Reconfigure virtual machine` task has run on the appliance VM.
5. Power on the appliance VM.

Base64 encoded file arguments are not decoded in Kubernetes profiles

Symptom

Some file arguments in Kubernetes profiles are base64 encoded. When the management console displays the Kubernetes profile, some file arguments are not decoded.

Workaround

Run `echo "$content" | base64 --decode`

Network profiles not immediately selectable

Symptom

If you create network profiles and then try to apply them in the Create Cluster page, the new profiles are not available for selection.

Workaround

Log out of the management console and log back in again.

Real-Time IP information not displayed for network profiles

Symptom

In the cluster summary page, only default IP pool, pod IP block, node IP block values are displayed, rather than the real-time values from the associated network profile.

Workaround

None

Error After Modifying Your Harbor Storage Configuration

Symptom

You receive the following error after modifying your existing Harbor installation's storage configuration:

```
Error response from daemon: manifest for ... not found: manifest unknown: manifest unknown
```

Explanation

Harbor does not support modifying an existing Harbor installation's storage configuration.

Workaround

To modify your Harbor storage configuration, re-install Harbor. Before starting Harbor, configure the new Harbor installation with the desired configuration.

Windows Stemcells Must be Re-Imported After Upgrading Ops Manager

Symptom

After upgrading Ops Manager, your Management Console does not recognize a Windows stemcell imported when using the prior version of Ops Manager.

Workaround

If your Management Console does not recognize a Windows stemcell after upgrading Ops Manager:

1. Re-import your previously imported Windows stemcell.
 2. **Apply Changes** to TKGI MC.
-

Your New Clusters Are Not Shown In Tanzu Mission Control

Symptom

After you create a cluster, Tanzu Mission Control does not include the cluster in cluster lists. You have a “*Resource not found*” error similar to the following in your BOSH logs:

```
Cluster Name in TMC: cluster-1
Cluster Name Prefix: tkgi-my-prefix-
Group Name in TMC: my-prefix-clusters
Cluster Description in TMC: VMware Enterprise PKS Attaching cluster ''tkgi-my-prefix-cluster-1'' to TMC
Fetching token successful
request POST:/v1alpha1/clusters,
response 404 Not Found:{"error":"Resource not found - clustergroup(my-prefix-clusters) org id(d859dc9f-g622-426d-8c91-939a9f13dea9)", "code":5,"message":"Resource not found - clustergroup(my-prefix-clusters)"}  
"code":5,"message":"Resource not found - clustergroup(my-prefix-clusters)"
```

Explanation

The cluster group you assign a cluster to must be defined in Tanzu Mission Control before you assign your cluster to the cluster group in the TKGI Management Console.

Workaround

To resolve the problem, complete the steps in [Attaching a Tanzu Kubernetes Grid Integrated \(TKGI\) cluster to Tanzu Mission Control \(TMC\) fails with “Resource not found - clustergroup\(cluster-group-name\)”](#) in the VMware Tanzu Knowledge Base.

Tanzu Kubernetes Grid Integrated Edition Concepts

This topic describes VMware Tanzu Kubernetes Grid Integrated Edition concepts. See the following sections:

- [Tanzu Kubernetes Grid Integrated Edition Architecture](#)
- [About Tanzu Kubernetes Grid Integrated Edition Upgrades](#)
- [TKGI API Authentication](#)
- [Load Balancers in Tanzu Kubernetes Grid Integrated Edition](#)
- [VM Sizing for Tanzu Kubernetes Grid Integrated Edition Clusters](#)
- [Telemetry](#)
- [Sink Architecture in Tanzu Kubernetes Grid Integrated Edition](#)
- [Containererd Container Runtime](#)

Tanzu Kubernetes Grid Integrated Edition Architecture

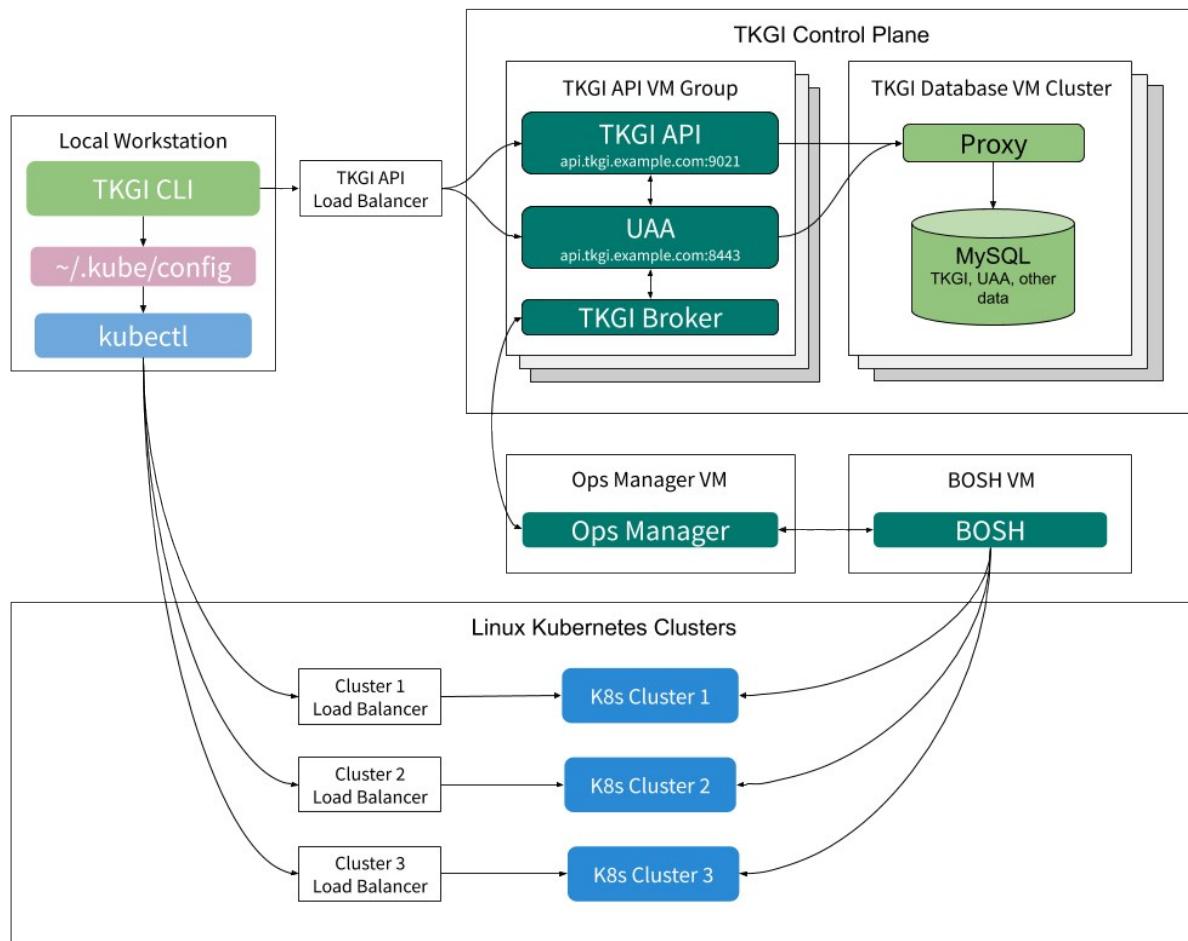
This topic describes how VMware Tanzu Kubernetes Grid Integrated Edition manages the deployment of Kubernetes clusters.

Tanzu Kubernetes Grid Integrated Edition Overview

A Tanzu Kubernetes Grid Integrated Edition environment consists of a TKGI Control Plane and one or more workload clusters.

Tanzu Kubernetes Grid Integrated Edition administrators use the TKGI Control Plane to deploy and manage Kubernetes clusters. The workload clusters run the apps pushed by developers.

The following illustrates the interaction between Tanzu Kubernetes Grid Integrated Edition components:



Administrators access the TKGI Control Plane through the TKGI Command Line Interface (TKGI CLI) installed on their local workstations.

Within the TKGI Control Plane the TKGI API and TKGI Broker use BOSH to execute the requested cluster management functions. For information about the TKGI Control Plane, see [TKGI Control Plane Overview](#) below. For instructions on installing the TKGI CLI, see [Installing the TKGI CLI](#).

Kubernetes deploys and manages workloads on Kubernetes clusters. Administrators use the Kubernetes CLI, `kubectl`, to direct Kubernetes from their local workstations. For information about `kubectl`, see [Overview of kubectl] (<https://kubernetes.io/docs/reference/kubectl/overview/>) in the Kubernetes documentation.

TKGI Control Plane Overview

The TKGI Control Plane manages the lifecycle of Kubernetes clusters deployed using Tanzu Kubernetes Grid Integrated Edition.

The control plane provides the following via the TKGI API:

- View cluster plans
- Create clusters
- View information about clusters
- Obtain credentials to deploy workloads to clusters

- Scale clusters
- Delete clusters
- Create and manage network profiles for VMware NSX-T

In addition, the TKGI Control Plane can upgrade all existing clusters using the **Upgrade all clusters** BOSH errand. For more information, see [Upgrade Kubernetes Clusters](#) in *Upgrading Tanzu Kubernetes Grid Integrated Edition (Antrea and Flannel Networking)*.

TKGI Control Plane is hosted on a pair of VM groups:

- The [TKGI API VM Group](#) for hosting cluster management services.
- The [TKGI Database VM Cluster](#) to store cluster management data.

TKGI API VM Group

The instances in the TKGI API VM Group host the following services:

- User Account and Authentication (UAA)
- TKGI API
- TKGI Broker
- Billing and Telemetry

The following sections describe UAA, TKGI API, and TKGI Broker services, the primary services hosted on the TKGI API VM.

UAA

When a user logs in to or logs out of the TKGI API through the TKGI CLI, the TKGI CLI communicates with UAA to authenticate them. The TKGI API permits only authenticated users to manage Kubernetes clusters. For more information about authenticating, see [TKGI API Authentication](#).

UAA must be configured with the appropriate users and user permissions. For more information, see [Managing Tanzu Kubernetes Grid Integrated Edition Users with UAA](#).

TKGI API

Through the TKGI CLI, users instruct the TKGI API service to deploy, scale up, and delete Kubernetes clusters as well as show cluster details and plans. The TKGI API can also write Kubernetes cluster credentials to a local kubeconfig file, which enables users to connect to a cluster through `kubectl`.

On AWS, GCP, and vSphere without NSX-T deployments the TKGI CLI communicates with the TKGI API within the control plane via the TKGI API Load Balancer. On vSphere with NSX-T deployments the TKGI API host is accessible via a DNAT rule. For information about enabling the TKGI API on vSphere with NSX-T, see the [Share the TKGI API Endpoint](#) section in *Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T Integration*.

The TKGI API sends all cluster management requests, except read-only requests, to the TKGI Broker.

TKGI Broker

When the TKGI API receives a request to modify a Kubernetes cluster, it instructs the TKGI Broker to make the requested change.

The TKGI Broker consists of an [On-Demand Service Broker](#) and a Service Adapter. The TKGI Broker generates a BOSH manifest and instructs the BOSH Director to deploy or delete the Kubernetes cluster.

For Tanzu Kubernetes Grid Integrated Edition deployments on vSphere with NSX-T, there is an additional component, the Tanzu Kubernetes Grid Integrated Edition NSX-T Proxy Broker. The TKGI API communicates with the TKGI NSX-T Proxy Broker, which in turn communicates with the NSX Manager to provision the Node Networking resources. The TKGI NSX-T Proxy Broker then forwards the request to the On-Demand Service Broker to deploy the cluster.

TKGI Database VM Cluster

The instances in the TKGI Database VM Cluster host MySQL, proxy, and other data-related services. These data-related functions persist TKGI Control Plane data for the the following services:

- TKGI API
- UAA
- Billing
- Telemetry

High Availability Modes

Tanzu Kubernetes Grid Integrated Edition can be configured for TKGI Control Plane and workload high availability.

TKGI Control Plane High Availability Mode (Beta)

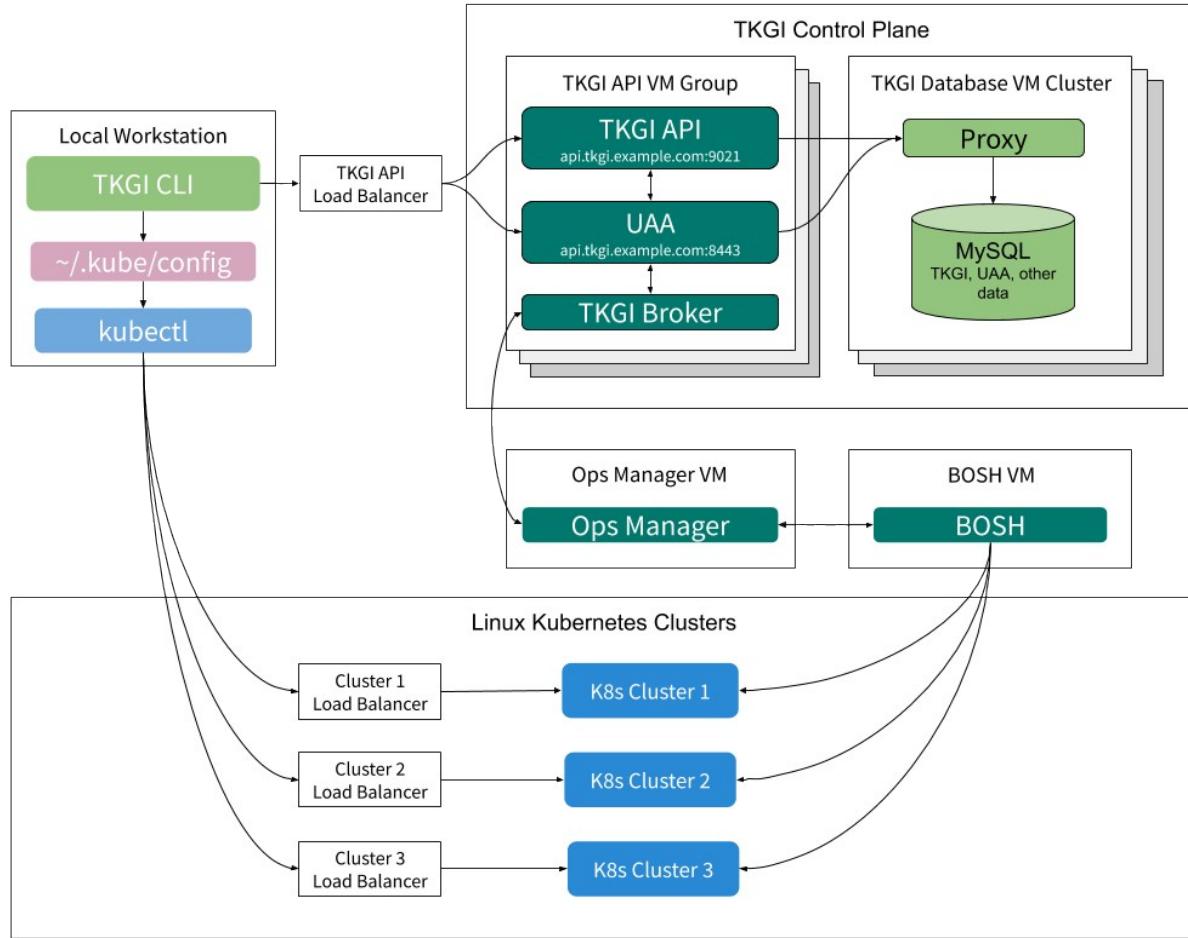
The TKGI Control Plane can be configured in either standard or high availability (beta) modes.

- In standard mode:
 - The TKGI API is hosted on the `pivotal-container-service` VM.
 - The TKGI Database is hosted on the `pks-db` VM.
- In high availability mode (beta):
 - The TKGI API is hosted on multiple `pivotal-container-service` VMs.
 - The TKGI Database is hosted on three `pks-db` VMs.



Warning: High availability mode is a beta feature. Do not scale your **TKGI API** or **TKGI Database** to more than one instance in production environments.

The following illustrates the interaction between Tanzu Kubernetes Grid Integrated Edition components in high availability mode:



You establish HA mode during the resource configuration phase of TKGI tile deployment. You can change the number of instances from 1 to 2 or 3 for the TKGI API, and from 1 to 3 for the TKGI Database. Once you set HA mode and increase the number of instances beyond 1, you cannot decrease the number of instances.

Resource Config

JOB	INSTANCES	VM TYPE	PERSISTENT DISK TYPE	SAVE
TKGI Database(HA is BETA)	Automatic: 1	Automatic: large.disk (cpu: 2, ram: 8 GB, c)	Automatic: 10 GB	
TKGI API(HA is BETA)	Automatic: 1	Automatic: large.disk (cpu: 2, ram: 8 GB, c)	Automatic: 10 GB	

Windows Worker-Based Kubernetes Cluster High Availability

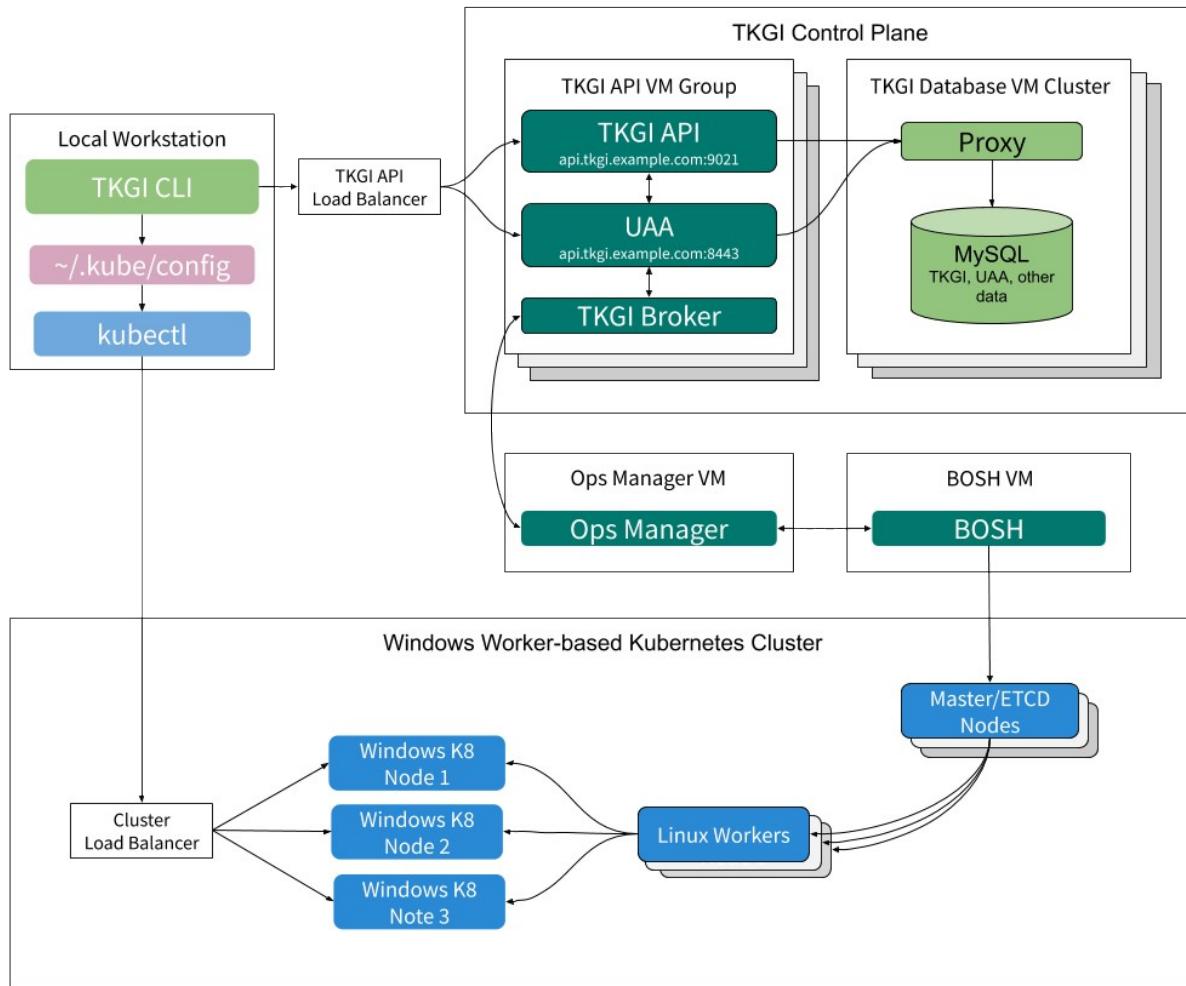
Windows worker-based cluster Linux nodes can be configured in either standard or high availability modes.

- In standard mode, a single control plane/etcd node and a single Linux worker manage a

cluster's Windows Kubernetes VMs.

- In high availability mode, multiple control plane/etcd and Linux worker nodes manage a cluster's Windows Kubernetes VMs.

The following illustrates the interaction between the Tanzu Kubernetes Grid Integrated Edition Management Plane and Windows worker-based Kubernetes clusters:



To configure Tanzu Kubernetes Grid Integrated Edition Windows worker-based clusters for high availability, set these fields in the **Plan** pane as described in [Plans in Configuring Windows Worker-Based Kubernetes Clusters](#):

- **Enable HA Linux workers**
- **Master/ETCD Node Instances**
- **Worker Node Instances**

About Tanzu Kubernetes Grid Integrated Edition Upgrades

This topic provides conceptual information about Tanzu Kubernetes Grid Integrated Edition upgrades, including upgrading the TKGI control plane and TKGI-provisioned Kubernetes clusters.

For step-by-step instructions on upgrading Tanzu Kubernetes Grid Integrated Edition and TKGI-provisioned Kubernetes clusters, see:

- [Upgrading Tanzu Kubernetes Grid Integrated Edition \(Antrea and Flannel Networking\)](#)
- [Upgrading Tanzu Kubernetes Grid Integrated Edition \(NSX-T Networking\)](#)
- [Upgrading Clusters](#)

Overview

An Tanzu Kubernetes Grid Integrated Edition upgrade modifies the version of Tanzu Kubernetes Grid Integrated Edition, for example, from v1.11.x to v1.12.0 or from v1.12.0 to v1.12.1.

By default, Tanzu Kubernetes Grid Integrated Edition is set to perform a full upgrade, which upgrades both the TKGI control plane and all TKGI-provisioned Kubernetes clusters.

However, you can choose to upgrade Tanzu Kubernetes Grid Integrated Edition in two phases by upgrading the TKGI control plane first and then upgrading your TKGI-provisioned Kubernetes clusters later.

You can use either the Tanzu Kubernetes Grid Integrated Edition tile or the TKGI CLI to perform TKGI upgrades:

- To perform a full upgrade of the TKGI control plane and TKGI-provisioned Kubernetes clusters, use the Tanzu Kubernetes Grid Integrated Edition tile .
- To upgrade the TKGI control plane only, use the Tanzu Kubernetes Grid Integrated Edition tile.
- To upgrade TKGI-provisioned Kubernetes clusters, use either the TKGI CLI or the Tanzu Kubernetes Grid Integrated Edition tile.

Upgrade Method	Supported Upgrade Types		
	Full TKGI upgrade	TKGI control plane only	Kubernetes clusters only
TKGI Tile	✓	✓	✓
TKGI CLI	*	*	✓

Typically, if you choose to upgrade TKGI-provisioned Kubernetes clusters only, you will upgrade them through the TKGI CLI.

Deciding Between Full and Two-Phase Upgrade

When deciding whether to perform the default full upgrade or to upgrade the TKGI control plane and TKGI-provisioned Kubernetes clusters separately, consider your organization needs.

For example, if your organization runs TKGI-provisioned Kubernetes clusters in both development and production environments and you want to upgrade only one environment first, you can achieve your goal by upgrading the TKGI control plane and TKGI-provisioned Kubernetes separately instead of performing a full upgrade.

Examples of other advantages of upgrading Tanzu Kubernetes Grid Integrated Edition in two phases include:

- Faster Tanzu Kubernetes Grid Integrated Edition tile upgrades. If you have a large number of clusters in your Tanzu Kubernetes Grid Integrated Edition deployment, performing a full upgrade can significantly increase the amount of time required to upgrade the Tanzu

Kubernetes Grid Integrated Edition tile.

- More granular control over cluster upgrades. In addition to enabling you to upgrade subsets of clusters, the TKGI CLI supports upgrading each cluster individually.
- Not a monolithic upgrade. This helps isolate the root cause of an error when troubleshooting upgrades. For example, when a cluster-related upgrade error occurs during a full upgrade, the entire Tanzu Kubernetes Grid Integrated Edition tile upgrade might fail.



Warning: If you deactivate the default full upgrade and upgrade only the TKGI control plane, you must upgrade all your TKGI-provisioned Kubernetes clusters before the next Tanzu Kubernetes Grid Integrated Edition tile upgrade. Deactivating the default full upgrade and upgrading only the TKGI control plane cause the TKGI version tagged in your Kubernetes clusters to fall behind the Tanzu Kubernetes Grid Integrated Edition tile version. If your TKGI-provisioned Kubernetes clusters fall more than one version behind the tile, Tanzu Kubernetes Grid Integrated Edition cannot upgrade the clusters.

What Happens During Full TKGI and TKGI Control Plane Upgrades

During a Tanzu Kubernetes Grid Integrated Edition control plane upgrade to v1.12, the Tanzu Kubernetes Grid Integrated Edition tile does the following:

1. **Recreates the Control Plane VMs** which host the TKGI API and UAA servers.
 - If the Tanzu Kubernetes Grid Integrated Edition installation is not scaled for high availability (beta), this control plane recreation causes temporary outages as described in [Non-HA Control Plane Outages](#), below.
2. **(Optional) Upgrades Clusters**
 - Upgrading Tanzu Kubernetes Grid Integrated Edition only upgrades its Kubernetes clusters if the **Upgrade all clusters errand** checkbox is enabled in the **Errands** pane.
 - The cluster upgrade process recreates all clusters, which might cause cluster outages.

For more information, see the [What Happens During Cluster Upgrades](#) section of the [Upgrading Clusters](#) topic.

You can perform full TKGI upgrades and TKGI control plane upgrades only through the Tanzu Kubernetes Grid Integrated Edition tile.

After you add a new Tanzu Kubernetes Grid Integrated Edition tile version to your staging area on the Ops Manager Installation Dashboard, Ops Manager automatically migrates your configuration settings into the new tile version.

For more information, see:

- [Full TKGI Upgrades](#)
- [TKGI Control Plane Upgrades](#)

Full TKGI Upgrades

During a **full TKGI upgrade**, the Tanzu Kubernetes Grid Integrated Edition tile does the following:

1. Upgrades the TKGI control plane, which includes the TKGI API and UAA servers and the TKGI database. This control plane upgrade causes temporary outages as described in [Control Plane Outages](#) below.
2. Upgrades TKGI-provisioned Kubernetes clusters.
 - ◊ Upgrading TKGI-provisioned Kubernetes clusters is controlled by the **Upgrade all clusters errand** in the Tanzu Kubernetes Grid Integrated Edition tile.
 - ◊ The cluster upgrade process recreates all clusters, which might cause cluster outages. For more information, see [What Happens During Cluster Upgrades](#) below.

TKGI Control Plane Upgrades

When upgrading the **TKGI control plane only**, the Tanzu Kubernetes Grid Integrated Edition tile follows the process described in [Full TKGI Upgrades](#) above, step 1. It does not upgrade TKGI-provisioned Kubernetes clusters, step 2.

Control Plane Outages

When the Tanzu Kubernetes Grid Integrated Edition control plane is not scaled for high availability (beta), upgrading it temporarily interrupts the following:

- Logging in to the TKGI CLI and using all `tkgi` commands
- Using the TKGI API to retrieve information about clusters
- Using the TKGI API to create and delete clusters
- Using the TKGI API to resize clusters

These outages do not affect the Kubernetes clusters themselves. During a TKGI control plane upgrade, you can still interact with clusters and their workloads using the Kubernetes Command Line Interface, `kubectl`.

For more information about the TKGI control plane and high availability (beta), see [TKGI Control Plane Overview](#) in *Tanzu Kubernetes Grid Integrated Edition Architecture*.

Canary Instances

The Tanzu Kubernetes Grid Integrated Edition tile is a BOSH deployment.

BOSH-deployed products can set a number of canary instances to upgrade first, before the rest of the deployment VMs. BOSH continues the upgrade only if the canary instance upgrade succeeds. If the canary instance encounters an error, the upgrade stops running and other VMs are not affected.

The Tanzu Kubernetes Grid Integrated Edition tile uses one canary instance when deploying or upgrading Tanzu Kubernetes Grid Integrated Edition.

What Happens During Cluster Upgrades

Upgrading TKGI-provisioned Kubernetes clusters updates their Kubernetes version to the version included with the Tanzu Kubernetes Grid Integrated Edition tile. It also updates the TKGI version

tagged in your clusters to the Tanzu Kubernetes Grid Integrated Edition tile version.

You can upgrade TKGI-provisioned Kubernetes clusters either through the Tanzu Kubernetes Grid Integrated Edition tile or the TKGI CLI. See the table below.

This method	Upgrades
The Upgrade all clusters errand in the Tanzu Kubernetes Grid Integrated Edition tile > Errands	All clusters. Clusters are upgraded serially.
<code>tkgi upgrade-cluster</code>	One cluster.
<code>tkgi upgrade-clusters</code>	Multiple clusters. Clusters are upgraded serially or in parallel.

During an upgrade of TKGI-provisioned clusters, Tanzu Kubernetes Grid Integrated Edition recreates your clusters. This includes the following stages for each cluster you upgrade:

1. Control Plane nodes are recreated.
2. Worker nodes are recreated.

Depending on your cluster configuration, these recreations might cause [Control Plane Nodes Outage](#) or [Worker Nodes Outage](#) as described below.

Control Plane Nodes Outage

When Tanzu Kubernetes Grid Integrated Edition upgrades a single-control plane node cluster, you cannot interact with your cluster, use `kubectl`, or push new workloads.

To avoid this loss of functionality, VMware recommends using multi-control plane node clusters.

Worker Nodes Outage

When Tanzu Kubernetes Grid Integrated Edition upgrades a worker node, the node stops running containers. If your workloads run on a single node, they will experience downtime.

To avoid downtime for stateless workloads, VMware recommends using at least one worker node per availability zone (AZ). For stateful workloads, VMware recommends using a minimum of two worker nodes per AZ.



Note: When the **Upgrade all clusters errand** is enabled in the Tanzu Kubernetes Grid Integrated Edition tile, updating the tile with a new Linux or Windows stemcell rolls every Linux or Windows VM in each Kubernetes cluster. This automatic rolling ensures that all your VMs are patched. To avoid workload downtime, use the resource configuration recommended in [Control Plane Nodes Outage](#) and [Worker Nodes Outage](#) above and in [Maintaining Workload Uptime](#).

About Switching from the Flannel CNI to the Antrea CNI

Tanzu Kubernetes Grid Integrated Edition supports Antrea, Flannel, and NSX-T as the Container Network Interfaces (CNIs) for TKGI-provisioned clusters.

VMware recommends the Antrea CNI over Flannel. The Antrea CNI provides Kubernetes Network

Policy support for non-NSX-T environments. Antrea CNI-configured clusters are supported on AWS, Azure, and vSphere without NSX-T environments.

For more information about Antrea, see [Antrea](#) in the Antrea documentation.



Note: Support for the Flannel Container Networking Interface (CNI) is deprecated.

VMware recommends that you configure Antrea as the default TKGI-provisioned cluster CNI, and that you switch your Flannel CNI-configured clusters to the Antrea CNI.

Switch from the Flannel CNI to Antrea

You can configure TKGI to network newly created TKGI-provisioned clusters with the Antrea CNI.

Configure the TKGI default CNI during TKGI installation and upgrade only.

During TKGI installation:

- Configure the TKGI default CNI as either Antrea or vSphere with NSX-T.

During TKGI upgrades:

- You can optionally change the TKGI default CNI from the deprecated Flannel CNI to Antrea.
- Do not change the CNI configuration from Antrea to Flannel.
- Do not change the CNI configuration from or to NSX-T after your initial TKGI installation.

If you initially configured TKGI to use Flannel as the default CNI and switch to Antrea as the default CNI during a TKGI upgrade:

- Existing Flannel-configured clusters remain networked using Flannel. Your existing Flannel clusters will not be migrated to Antrea.
- New clusters created after the upgrade are created using Antrea as their CNI.



Warning:

Do not change the TKGI default CNI configuration between upgrades.

For information about selecting and configuring a CNI for TKGI, see the *Networking* section of the installation documentation for your environment:

- [Installing VMware Tanzu Kubernetes Grid Integrated Edition on AWS](#)
- [Installing VMware Tanzu Kubernetes Grid Integrated Edition on Azure](#)
- [Installing VMware Tanzu Kubernetes Grid Integrated Edition on vSphere](#)

TKGI API Authentication

This topic describes how the VMware Tanzu Kubernetes Grid Integrated Edition API works with User Account and Authentication (UAA) to manage authentication and authorization in your Tanzu Kubernetes Grid Integrated Edition deployment.

Authentication of TKGI API Requests

Before users can log in and use the TKGI CLI, you must configure TKGI API access with UAA. For more information, see [Managing Tanzu Kubernetes Grid Integrated Edition Users with UAA](#) and [Logging in to Tanzu Kubernetes Grid Integrated Edition](#).

You use the UAA Command Line Interface (UAAC) to target the UAA server and request an access token for the UAA admin user. If your request is successful, the UAA server returns the access token. The UAA admin access token authorizes you to make requests to the TKGI API using the TKGI CLI and grant cluster access to new or existing users.

When a user with cluster access logs in to the TKGI CLI, the CLI requests an access token for the user from the UAA server. If the request is successful, the UAA server returns an access token to the TKGI CLI. When the user runs TKGI CLI commands, for example, `tkgi clusters`, the CLI sends the request to the TKGI API server and includes the user's UAA token.

The TKGI API sends a request to the UAA server to validate the user's token. If the UAA server confirms that the token is valid, the TKGI API uses the cluster information from the TKGI broker to respond to the request. For example, if the user runs `tkgi clusters`, the CLI returns a list of the clusters that the user is authorized to manage.

Routing to the TKGI API VM

The TKGI API server and the UAA server use different port numbers on the API VM. For example, if your TKGI API domain is `api.tkgi.example.com`, you can reach your TKGI API and UAA servers at the following URLs:

Server	URL
TKGI API	<code>api.tkgi.example.com:9021</code>
UAA	<code>api.tkgi.example.com:8443</code>

Refer to [Ops Manager > Tanzu Kubernetes Grid Integrated Edition tile > TKGI API > API Hostname \(FQDN\)](#) for your TKGI API domain.

Load balancer implementations differ by deployment environment. For Tanzu Kubernetes Grid Integrated Edition deployments on GCP, AWS, or vSphere without NSX-T, you configure a load balancer to access the TKGI API when you install the Tanzu Kubernetes Grid Integrated Edition tile. For example, see [Configuring TKGI API Load Balancer](#).

For overview information about load balancers in Tanzu Kubernetes Grid Integrated Edition, see [Load Balancers in Tanzu Kubernetes Grid Integrated Edition Deployments without NSX-T](#).

Load Balancers in Tanzu Kubernetes Grid Integrated Edition

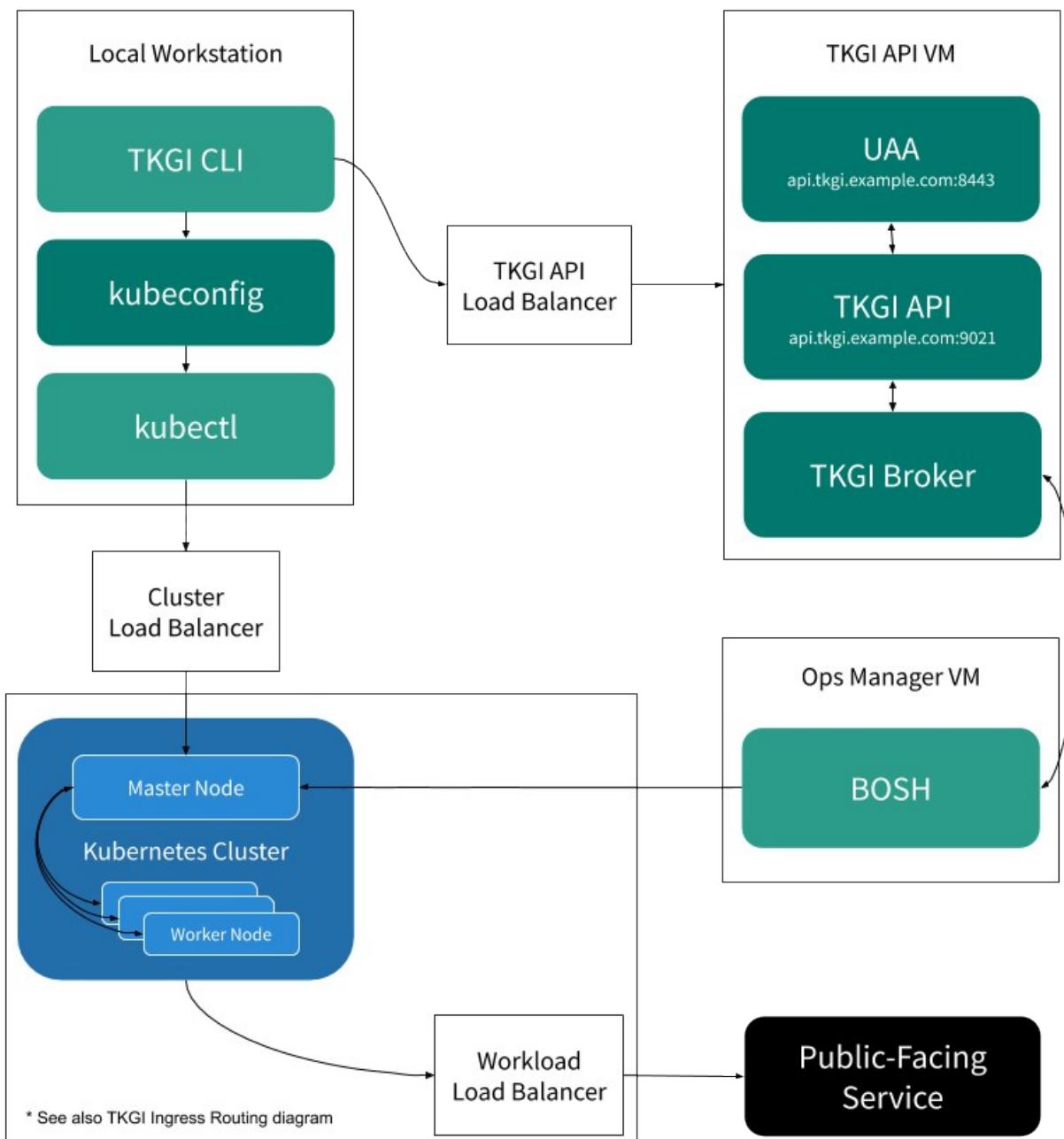
This topic describes the types of load balancers that are used in VMware Tanzu Kubernetes Grid Integrated Edition deployments. Load balancers differ by the type of deployment.

Load Balancers in Tanzu Kubernetes Grid Integrated Edition Deployments without NSX-T

For Tanzu Kubernetes Grid Integrated Edition deployments on GCP, AWS, or vSphere without NSX-T, you can configure load balancers for the following:

- **TKGI API:** Configuring this load balancer enables you to run TKGI Command Line Interface (TKGI CLI) commands from your local workstation.
- **Kubernetes Clusters:** Configuring a load balancer for each new cluster enables you to run Kubernetes CLI (kubectl) commands on the cluster.
- **Workloads:** Configuring a load balancer for your application workloads enables external access to the services that run on your cluster.

The following diagram, applicable to GCP, AWS, and vSphere without NSX-T, shows where each of the above load balancers can be used within your Tanzu Kubernetes Grid Integrated Edition deployment.



If you use either vSphere without NSX-T or GCP, you are expected to create your own load

balancers within your cloud provider console. If your cloud provider does not offer load balancing, you can use any external TCP or HTTPS load balancer of your choice.

About the TKGI API Load Balancer

The TKGI API load balancer enables you to access the TKGI API from outside the network on Tanzu Kubernetes Grid Integrated Edition deployments on GCP, AWS, and on vSphere without NSX-T. For example, configuring a load balancer for the TKGI API enables you to run TKGI CLI commands from your local workstation.

For information about configuring the TKGI API load balancer on vSphere without NSX-T, see [Configuring TKGI API Load Balancer](#).

About Kubernetes Cluster Load Balancers

When you create an Tanzu Kubernetes Grid Integrated Edition cluster on GCP, AWS, and on vSphere without NSX-T, you must configure external access to the cluster by creating an external TCP or HTTPS load balancer. The load balancer enables the Kubernetes CLI to communicate with the cluster.

If you create a cluster in a non-production environment, you can choose not to use a load balancer. To enable `kubectl` to access the cluster without a load balancer, you can do one of the following:

- Create a DNS entry that points to the cluster's control plane VM. For example:

my-cluster.example.com	A	10.0.0.5
------------------------	---	----------

- On the workstation where you run `kubectl` commands, add the control plane IP address of your cluster and `kubo.internal` to the `/etc/hosts` file. For example:

10.0.0.5 kubo.internal

For more information about configuring a cluster load balancer, see the following:

- [Creating and Configuring a GCP Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters](#)
- [Creating and Configuring an AWS Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters](#)
- [Creating and Configuring an Azure Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters](#)

About Workload Load Balancers

To enable external access to your Tanzu Kubernetes Grid Integrated Edition app on GCP, AWS, and on vSphere without NSX-T, you can either create a load balancer or expose a static port on your workload.

For information about configuring a load balancer for your app workload, see [Deploying and Exposing Basic Linux Workloads](#).

If you use AWS, you must configure routing in the AWS console before you can create a load balancer for your workload. You must create a public subnet in each availability zone (AZ) where you are deploying the workload and tag the public subnet with your cluster's unique identifier.

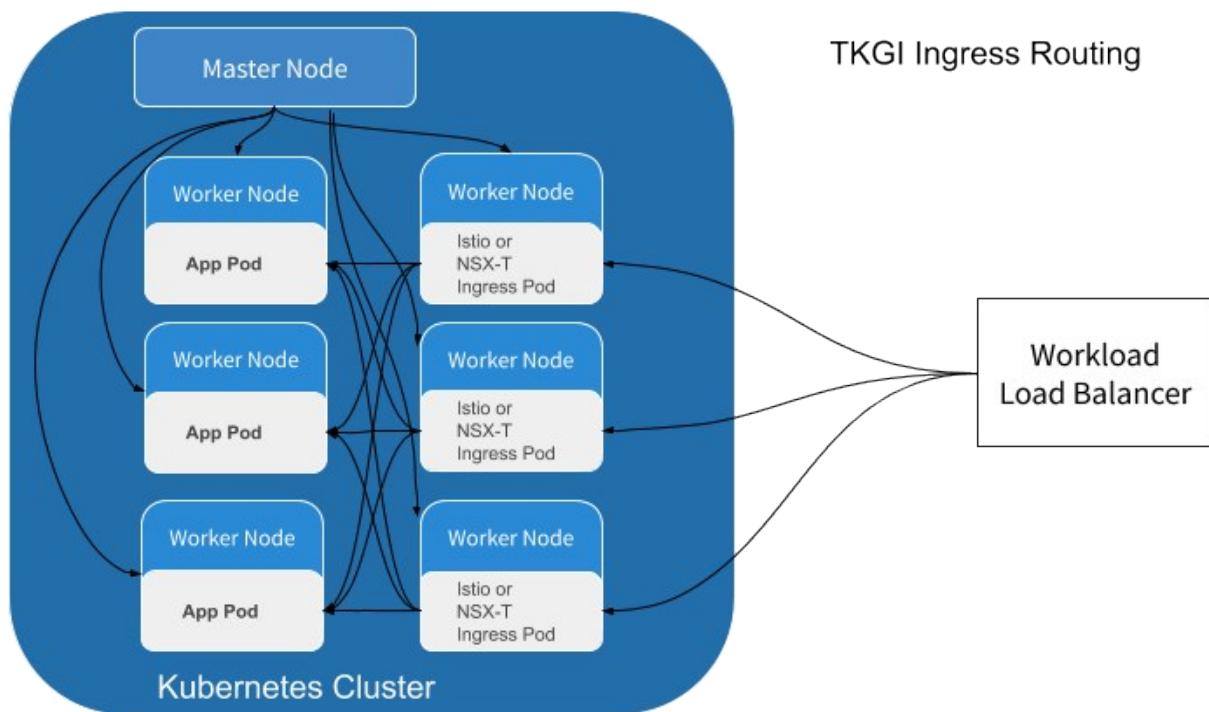
See the [AWS Prerequisites](#) section of *Deploying and Exposing Basic Linux Workloads* before you create a workload load balancer.

Deploy Your Workload Load Balancer with an Ingress Controller

A Kubernetes ingress controller sits behind a load balancer, routing HTTP and HTTPS requests from outside the cluster to services within the cluster. Kubernetes ingress resources can be configured to load balance traffic, provide externally reachable URLs to services, and manage other aspects of network traffic.

If you add an ingress controller to your Tanzu Kubernetes Grid Integrated Edition deployment, traffic routing is controlled by the ingress resource rules you define. VMware recommends configuring Tanzu Kubernetes Grid Integrated Edition deployments with both a workload load balancer and an ingress controller.

The following diagram shows how the ingress routing can be used within your Tanzu Kubernetes Grid Integrated Edition deployment.



The load balancer on Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T is automatically provisioned with Kubernetes ingress resources without the need to deploy and configure an additional ingress controller.

For information about deploying a load balancer configured with ingress routing on GCP, AWS, Azure, and vSphere without NSX-T, see [Configuring Ingress Routing](#). For information about ingress routing on vSphere with NSX-T, see [Configuring Ingress Resources and Load Balancer Services](#).

Load Balancers in Tanzu Kubernetes Grid Integrated Edition Deployments on vSphere with NSX-T

Tanzu Kubernetes Grid Integrated Edition deployments on vSphere with NSX-T do not require a

load balancer configured to access the TKGI API in singleton mode. They require only a DNAT rule configured so that the TKGI API host is accessible. For more information, see [Share the Tanzu Kubernetes Grid Integrated Edition Endpoint](#) in *Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T Integration*.

Tanzu Kubernetes Grid Integrated Edition deployments on vSphere with NSX-T do require a load balancer configured to access the TKGI API in high-availability mode. To configue an NSX-T load balancer for TKGI API traffic, see [Provisioning an NSX-T Load Balancer for the TKGI API Server](#).

At runtime, NSX-T handles load balancer creation, configuration, and deletion automatically as part of the Kubernetes cluster create, update, and delete process. When a new Kubernetes cluster is created, NSX-T creates and configures a dedicated load balancer tied to it. The load balancer is a shared resource designed to provide efficient traffic distribution to control plane nodes as well as services deployed on worker nodes. Each application service is mapped to a virtual server instance, carved out from the same load balancer. For more information, see [Logical Load Balancer](#) in the NSX-T documentation.

Virtual server instances are created on the load balancer to provide access to the following:

- **Kubernetes API and UI services on a Kubernetes cluster.** This enables requests to be load balanced across multiple control plane nodes.
- **Ingress controller.** This enables the virtual server instance to dispatch HTTP and HTTPS requests to services associated with Ingress rules.
- **`type:loadbalancer` services.** This enables the server to handle TCP connections or UDP flows toward exposed services.

Load balancers are deployed in high-availability mode so that they are resilient to potential failures and able to recover quickly from critical conditions.



Note: The `NodePort` Service type is not supported for Tanzu Kubernetes Grid Integrated Edition deployments on vSphere with NSX-T. Only `type:LoadBalancerServices` and Services associated with Ingress rules are supported on vSphere with NSX-T.

Resizing Load Balancers

When a new Kubernetes cluster is provisioned using the TKGI API, NSX-T creates a dedicated load balancer for that new cluster. By default, the size of the load balancer is set to Small.

With network profiles, you can change the size of the load balancer deployed by NSX-T at the time of cluster creation. For information about network profiles, see [Using Network Profiles \(NSX-T Only\)](#).

For more information about the types of load balancers NSX-T provisions and their capacities, see [Scaling Load Balancer Resources](#) in the NSX-T documentation.

VM Sizing for Tanzu Kubernetes Grid Integrated Edition Clusters

This topic describes how VMware Tanzu Kubernetes Grid Integrated Edition recommends you approach the sizing of VMs for cluster components.

Overview

When you configure plans in the Tanzu Kubernetes Grid Integrated Edition tile, you provide VM sizes for the control plane and worker node VMs. For more information about configuring plans, see the Plans section of *Installing Tanzu Kubernetes Grid Integrated Edition* for your IaaS:

- [vSphere](#)
- [vSphere with NSX-T Integration](#)
- [Google Cloud Platform \(GCP\)](#)
- [Amazon Web Services \(AWS\)](#)
- [Azure](#)

You select the number of control plane nodes when you configure the plan.

For worker node VMs, you select the number and size based on the needs of your workload. The sizing of control plane and worker node VMs is highly dependent on the characteristics of the workload. Adapt the recommendations in this topic based on your own workload requirements.

Control Plane Node VM Size

The control plane node VM size is linked to the number of worker nodes. The VM sizing shown in the following table is per control plane node:



Note: If there are multiple control plane nodes, all control plane node VMs are the same size. To configure the number of control plane nodes, see the Plans section of *Installing Tanzu Kubernetes Grid Integrated Edition* for your IaaS.

To customize the size of the Kubernetes control plane node VM, see [Customize Control Plane and Worker Node VM Size and Type](#).

Number of Workers	CPU	RAM (GB)
1-5	1	3.75
6-10	2	7.5
11-100	4	15
101-250	8	30
251-500	16	60
500+	32	120

Do not overload your control plane node VMs by exceeding the recommended maximum number of worker node VMs or by downsizing from the recommended VM sizings listed above. These recommendations support both a typical workload managed by a VM and the higher than usual workload managed by the VM while other VM's in the cluster are upgrading.



Warning: Upgrading an overloaded Kubernetes cluster control plane node VM can result in downtime.

Worker Node VM Number and Size

A maximum of 100 pods can run on a single worker node. The actual number of pods that each worker node runs depends on the workload type as well as the CPU and memory requirements of the workload.

To calculate the number and size of worker VMs you require, determine the following for your workload:

- Maximum number of pods you expect to run [p]
- Memory requirements per pod [m]
- CPU requirements per pod [c]

Using the values above, you can calculate the following:

- Minimum number of workers [w] = $p / 100$
- Minimum RAM per worker = $m * 100$
- Minimum number of CPUs per worker = $c * 100$

This calculation gives you the minimum number of worker nodes your workload requires. We recommend that you increase this value to account for failures and upgrades.

For example, increase the number of worker nodes by at least one to maintain workload uptime during an upgrade. Additionally, increase the number of worker nodes to fit your own failure tolerance criteria.

The maximum number of worker nodes that you can create for a plan in an Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes cluster is set by the **Maximum number of workers on a cluster** field in the **Plans** pane of the Tanzu Kubernetes Grid Integrated Edition tile. To customize the size of the Kubernetes worker node VM, see [Customize Control Plane and Worker Node VM Size and Type](#).

Example Worker Node Requirement Calculation

An example app has the following minimum requirements:

- Number of pods [p] = 1000
- RAM per pod [m] = 1 GB
- CPU per pod [c] = 0.10

To determine how many worker node VMs the app requires, do the following:

1. Calculate the number of workers using $p / 100$:

```
1000/100 = 10 workers
```

2. Calculate the minimum RAM per worker using $m * 100$:

```
1 * 100 = 100 GB
```

3. Calculate the minimum number of CPUs per worker using $c * 100$:

$0.10 * 100 = 10 \text{ CPUs}$

4. For upgrades, increase the number of workers by one:

$10 \text{ workers} + 1 \text{ worker} = 11 \text{ workers}$

5. For failure tolerance, increase the number of workers by two:

$11 \text{ workers} + 2 \text{ workers} = 13 \text{ workers}$

In total, this app workload requires 13 workers with 10 CPUs and 100 GB RAM.

Customize Control Plane and Worker Node VM Size and Type

You select the CPU, memory, and disk space for the Kubernetes node VMs from a set list in the Tanzu Kubernetes Grid Integrated Edition tile. Control Plane and worker node VM sizes and types are selected on a per-plan basis. For more information, see the Plans section of the Tanzu Kubernetes Grid Integrated Edition installation topic for your IaaS. For example, [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T](#).

While the list of available node VM types and sizes is extensive, the list may not provide the exact type and size of VM that you want. You can use the Ops Manager API to customize the size and types of the control plane and worker node VMs. For more information, see [How to Create or Remove Custom VM_TYPE Template using the Operations Manager API] (<https://community.pivotal.io/s/article/how-to-create-or-remove-custom-vmtype-template-using-the-ops-manager-api>) in the Knowledge Base.



Warning: Do not reduce the size of your Kubernetes control plane node VMs below the recommended sizes listed in [Control Plane Node VM Size](#), above. Upgrading an overloaded Kubernetes cluster control plane node VM can result in downtime.

VMware CEIP

This topic describes the VMware Customer Experience Improvement Program (CEIP) used in the Tanzu Kubernetes Grid Integrated Edition tile.

Overview

The CEIP program allows VMware to collect data from customer installations to improve the Tanzu Kubernetes Grid Integrated Edition experience. Collecting data at scale allows VMware to identify patterns and warning signals in Tanzu Kubernetes Grid Integrated Edition installations.

Customers who participate in the CEIP receive proactive support benefits that include a weekly report based on telemetry data. Contact your Customer Success Manager to subscribe to this report. You can view a sample report at [TKGI Platform Operations Report](#).



Note: Tanzu Kubernetes Grid Integrated Edition does not collect any personally identifiable information (PII) at either participation level. For a list of the data Tanzu Kubernetes Grid Integrated Edition collects, see [Data Dictionary](#).

Configure CEIP

To configure CEIP, see the *VMware CEIP* section of the installation topic for your IaaS:

- [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#)
- [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T](#)
- [Installing Tanzu Kubernetes Grid Integrated Edition on AWS](#)
- [Installing Tanzu Kubernetes Grid Integrated Edition on Azure](#)
- [Installing Tanzu Kubernetes Grid Integrated Edition on GCP](#)

Proxy Communication

If you use a proxy server, the Tanzu Kubernetes Grid Integrated Edition proxy settings apply to outgoing CEIP data.

To configure Tanzu Kubernetes Grid Integrated Edition proxy settings for CEIP and other communications, see the following:

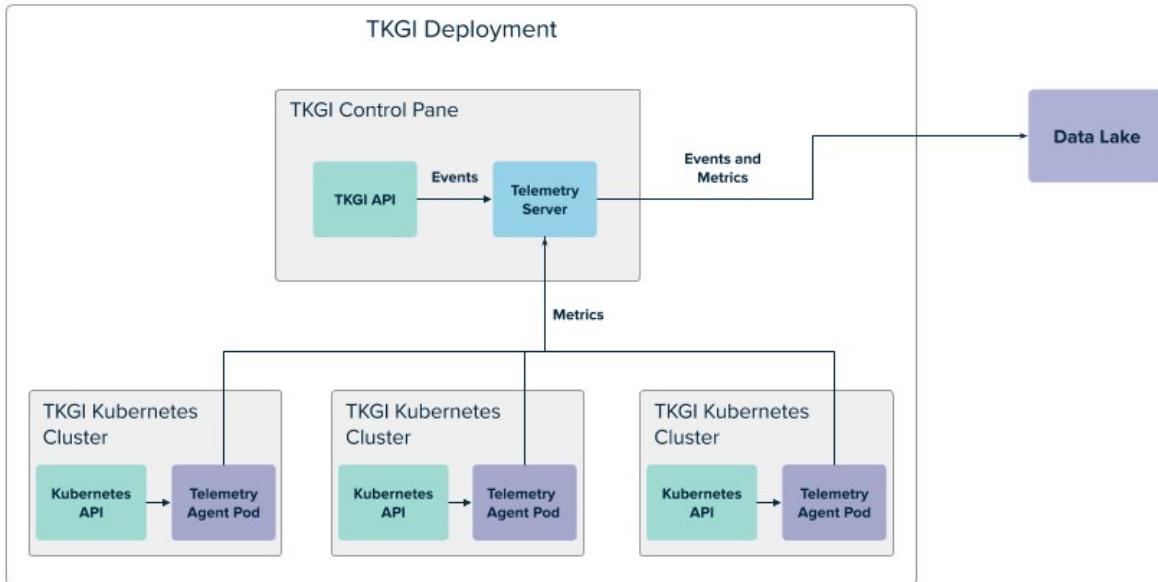
- For AWS, see [Using Proxies with Tanzu Kubernetes Grid Integrated Edition on AWS](#).
- For vSphere, see [Networking in Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#).
- For vSphere with NSX-T, see [Using Proxies with Tanzu Kubernetes Grid Integrated Edition on NSX-T](#).

System Components

The CEIP program use the following components to collect data:

- **Telemetry Server:** This component runs on the TKGI control plane. The server receives CEIP events from the TKGI API and metrics from Telemetry agent pods. The server sends events and metrics to a data lake for archiving and analysis.
- **Telemetry Agent Pod:** This component runs in each Kubernetes cluster as a deployment with one replica. Agent pods periodically poll the Kubernetes API for cluster metrics and send the metrics to the Telemetry server.

The following diagram shows how CEIP data flows through the system components:



[View a larger version of this image.](#)

Data Dictionary

For information about TKGI CEIP collection and reporting, see the [TKGI Telemetry Data](#) spreadsheet, hosted on Google Drive.

Sink Architecture in Tanzu Kubernetes Grid Integrated Edition

This topic describes how VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) implements sinks for collecting logs and metrics from Kubernetes worker nodes and workloads.

For step-by-step instructions on creating sinks in TKGI, see [Creating and Managing Sink Resources](#).

Overview

A sink collects logs or metrics about Kubernetes worker nodes in a TKGI deployment and workloads that are running on them.

For more information, see:

- [Sink Types](#) below
- [Sink Architecture](#) below

Sink Types

You can create two types of sinks:

- Log sinks
- Metric sinks

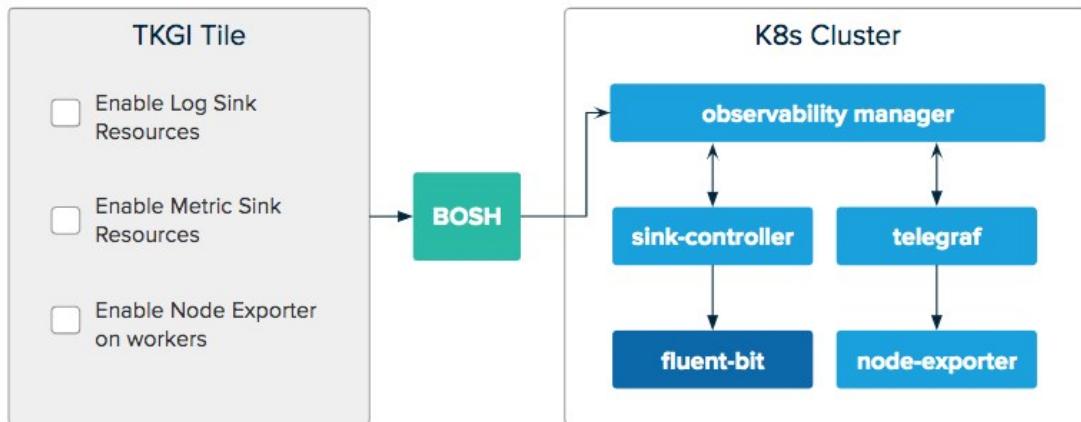
See the table below for information about these sink types.

Sink Type	Sink Resource	Description
Log sink	ClusterLogSink	<p>Forwards logs from a cluster to a log destination. Logs are transported using one of the following:</p> <ul style="list-style-type: none"> • The Syslog Protocol defined in RFC 5424 • WebHook • Fluent Bit output plugins
Log sink	LogSink	<p>Forwards logs from a namespaced subset within a ClusterLogSink resource to a log destination. Logs are transported using one of the following:</p> <ul style="list-style-type: none"> • The Syslog Protocol defined in RFC 5424 • WebHook • Fluent Bit output plugins
Metric sink	ClusterMetricSink	Collects and writes metrics from a cluster to specified outputs using input and output plugins.
Metric sink	MetricSink	Collects and writes metrics from a namespace within a cluster to specified outputs using input and output plugins.

Sink Architecture

TKGI-provisioned Kubernetes clusters include an observability manager that manages log sink and metric sink configurations within a cluster.

The following diagram details TKGI cluster observability architecture:



[View a larger version of this image.](#)

In the **Tanzu Kubernetes Grid Integrated Edition tile** > **In-Cluster Monitoring**:

- **Enable Metric Sink Resources** enables metric sinks.
- **Enable Log Sink Resources** enables log sinks.
- **Enable node exporter on workers** forwards additional infrastructure metrics.

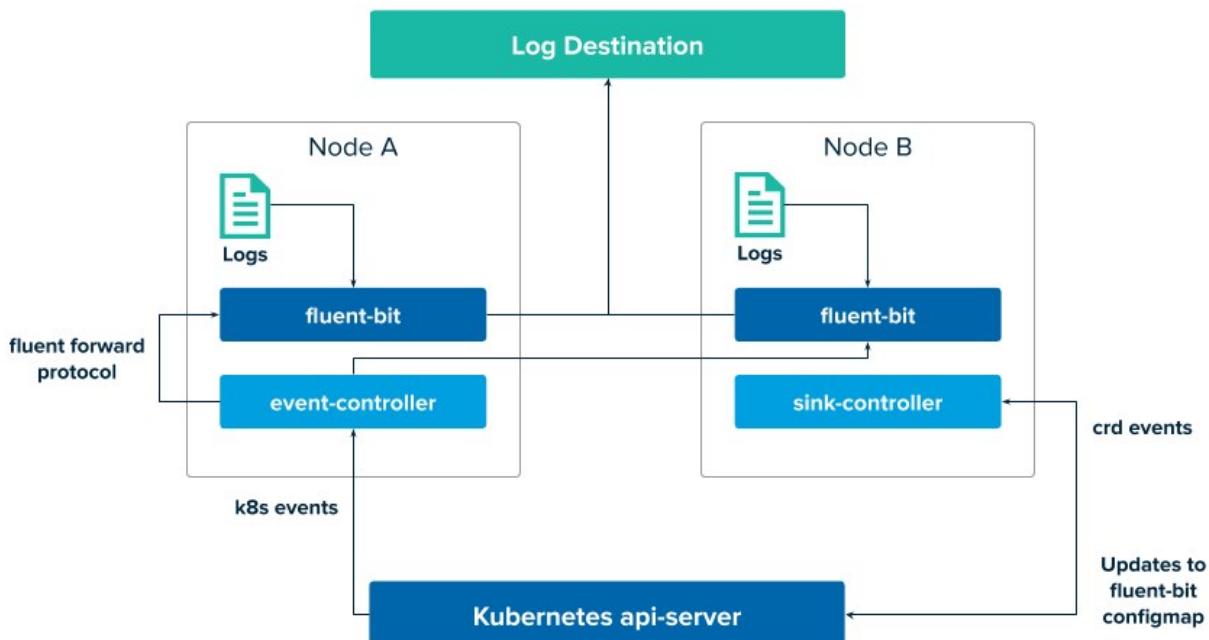
Setting these checkboxes in Ops Manager directs how BOSH configures the observability manager.

For more information about enabling log sinks and metrics sinks, see [\(Optional\) In-Cluster Monitoring](#) in the *Installing* topic for your IaaS.

Log Sink Architecture

The TKGI log sink aggregates workload logs and forwards them to a common log destination.

The following diagram details TKGI log sink architecture:



[View a larger version of this image.](#)

Logs are monitored and aggregated by a Fluent Bit [DaemonSet](#) running as a pod on each worker node.

An event-controller collects Kubernetes API events and sends them to a second Fluent Bit daemon pod for aggregation.

All aggregated log entries are marshaled to a common log destination.

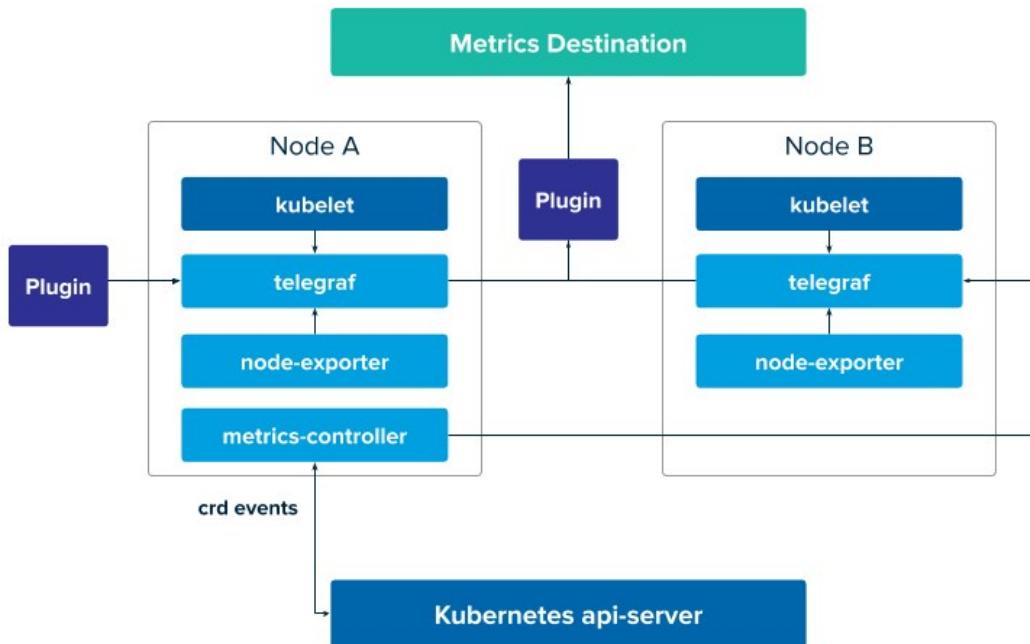


Note: When sinks are added or removed, all of the Fluent Bit pods are refreshed with new sink information.

Metric Sink Architecture

The TKGI metric sink aggregates workload metrics and forwards them to a common metrics destination.

The following diagram details TKGI metric sink architecture:



[View a larger version of this image.](#)

A metric sink collects and writes metrics from a cluster to specified outputs using input and output plugins.

Workload metrics are monitored by a set of third-party plugins. The plugins forward the metrics to a Telegraf service pod.

A pair of kubelets monitors Kubernetes and forwards Kubernetes metrics to a pair of Telegraf service pods.

If Node Exporter is enabled on the worker nodes in the Tanzu Kubernetes Grid Integrated Edition tile, a Node Exporter [DaemonSet](#) is included in all clusters. For more information about Node Exporter metrics, see the [Node Exporter](#) repository in GitHub.

To define the collected unstructured metrics, a metric-controller monitors Kubernetes for custom resource definitions and forwards those definitions to the Telegraf services.

The Telegraf services collect, process, and aggregate gathered metrics. All aggregated metrics are marshaled to an additional plugin for forwarding to a third-party application.



Note: When sinks are added or removed, all of the Telegraf pods are refreshed with new sink information.

Containerd Container Runtime

This topic discusses why VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) uses containerd as the default container runtime instead of Docker.

Overview

The containerd container runtime is a preferred container runtime for use with Kubernetes.

containerd was initially introduced with Docker v1.11 to simplify cloud architecture management.

containerd has become a high-level container runtime that uses the Container Runtime Interface (CRI) created for Kubernetes. containerd is a Graduated project maturity level project within the Cloud Native Computing Foundation.

The Docker container runtime is complex, handling everything from building images and creating volumes to managing network plugins and overlay networks. In Kubernetes v1.20, the Kubernetes community deprecated Docker as an underlying runtime in favor of runtimes that use the CRI and will remove Docker support in Kubernetes v1.22.

containerd is now the default container runtime for newly created TKGI Kubernetes clusters due to the deprecation and anticipated removal of Docker runtime support in future Kubernetes releases.

Benefits of Containerd

containerd provides the following features and benefits:

- As a high-level container runtime, containerd does not require Docker to run. It runs on its own, with runc as its low-level container runtime. For deploying and managing Kubernetes, containerd can replace Docker and Docker-shim with CRI-Containerd.
- Containerd abstracts system calls and operating system-specific functionality to provide a simple container runtime that specializes in running images in containers, pushing and pulling images to the registry, and managing the images themselves. Because of these abstractions, containerd works with Linux and Windows both on-premise and in the cloud.
- Containerd is compatible with other low-level runtimes besides runc, supports tools such as kata-runtime to run containers, and supports running multiple container runtimes within the same environment.
- Containerd provides container lifecycle APIs to create, execute, and manage containers and their tasks, an entire API dedicated to snapshot management.

Installing Tanzu Kubernetes Grid Integrated Edition

Tanzu Kubernetes Grid Integrated Edition Management Console (vSphere Only)

See the following documentation for the Management Console, which is the recommended method for installing Tanzu Kubernetes Grid Integrated Edition on vSphere:

- [Install Tanzu Kubernetes Grid Integrated Edition on vSphere with the Management Console](#)

For more information, see [When Should I Use Tanzu Kubernetes Grid Integrated Edition Management Console?](#).

Tanzu Kubernetes Grid Integrated Edition on Ops Manager

See the following documentation for how to manually install Tanzu Kubernetes Grid Integrated Edition, using Ops Manager, on Ops Manager:

- [vSphere \(Antrea and Flannel Networking\)](#)
- [vSphere \(NSX-T Networking\)](#)
- [Google Cloud Platform \(Antrea and Flannel Networking\)](#)
- [Amazon Web Services \(Antrea and Flannel Networking\)](#)
- [Azure \(Antrea and Flannel Networking\)](#)



Note: Tanzu Kubernetes Grid Integrated Edition supports air-gapped deployments on vSphere with Antrea or Flannel Networking or with NSX-T integration.

Installing Tanzu Kubernetes Grid Integrated Edition on vSphere

This topic describes options for installing Tanzu Kubernetes Grid Integrated Edition (TKGI) on vSphere.

Overview

You can install TKGI on vSphere in four ways:

- [Install TKGI on vSphere with the Management Console](#)
- [Install TKGI on vSphere with NSX-T Using Ops Manager](#)

- [Install TKGI on vSphere with Flannel Using Ops Manager](#)
- [Install TKGI on VMware Cloud Foundation](#)

Which way you chose to install TKGI depends on whether you use the TKGI Management Console, and which container networking overlay you use.

Where possible, VMware recommends using the TKGI Management Console to install TKGI on vSphere. For more information, see [Decide When to Use Management Console](#) in *Install Tanzu Kubernetes Grid Integrated Edition on vSphere with the Management Console*.

Install Tanzu Kubernetes Grid Integrated Edition on vSphere with the Management Console

VMware Tanzu Kubernetes Grid Integrated Edition Management Console provides a unified installation experience for deploying VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) to vSphere.

Overview

The VMware Tanzu Kubernetes Grid Integrated Edition Management Console provides a graphical user interface to assist you with configuring and deploying TKGI to vSphere.

The TKGI Management Console automates installation of the following:

- Deploys Ops Manager
- Deploys BOSH Director
- Deploys Tanzu Kubernetes Grid Integrated Edition
- Deploys Harbor Registry
- Configures networking for Tanzu Kubernetes Grid Integrated Edition
- Generates and registers SSL certificates

The TKGI Management Console is provided as a virtual appliance that you deploy to vSphere by using an OVA template. If you are experienced with installing TKGI on vSphere, the help and the tool tips in the installer UI should be enough to complete the process. If you are new to TKGI, refer to the following documentation as needed to assist with the installation:

- [Prerequisites for Tanzu Kubernetes Grid Integrated Edition Management Console Deployment](#)
- [Deploy the Tanzu Kubernetes Grid Integrated Edition Management Console](#)
- [Deploy Tanzu Kubernetes Grid Integrated Edition from the Management Console](#)

Decide When to Use TKGI Management Console

The TKGI Management Console provides an opinionated process for deploying Tanzu Kubernetes Grid Integrated Edition. Whether you should use the management console depends on your situation.

When Should I Use Tanzu Kubernetes Grid Integrated Edition Management Console?

Use the TKGI Management Console to simplify deploying Tanzu Kubernetes Grid Integrated Edition to less complex vSphere environments.

Users on vSphere who do not have a custom Ops Manager installation might prefer to install TKGI using the curated workflow provided by the TKGI Management Console.

When Should I Not Use Tanzu Kubernetes Grid Integrated Edition Management Console?

Users who already have Ops Manager installed for other uses, for example to run [VMware Tanzu Application Service for VMs](#), might prefer to install TKGI using Ops Manager. Also, if you require more flexibility in configuring your TKGI deployment, especially in complex NSX-T Data Center deployments, it might be more appropriate to install TKGI using Ops Manager.

Do not use the TKGI Management Console if any the following conditions apply to your environment, they are not supported:

- Deployments to a No-NAT topology with a vSphere Standard Switch or a vSphere Distributed Switch.
- [Multi-Foundation](#) deployments.

Before using TKGI Management Console to deploy TKGI, consider the following factors:

- If you want to deploy TKGI Management Console to a No-NAT topology with an NSX-T Data Center logical switch, you must perform a BYOT deployment.
- Deployments to a [Multi-Tier-0](#) topology are supported in BYOT deployments only and require additional configuration. For information about the additional configuration required, see [Tanzu Kubernetes Grid Integrated Edition Management Console Cannot Retrieve Cluster Data in a Multi-Tier0 Topology](#) in *Troubleshooting Tanzu Kubernetes Grid Integrated Edition Management Console*.

For information about the supported topologies for a manual installation, see [NSX-T Deployment Topologies for TKGI](#).

Use the TKGI Management Console After Installing TKGI

After you have deployed Tanzu Kubernetes Grid Integrated Edition on vSphere, you can use the TKGI Management Console to deploy Kubernetes clusters and manage their lifecycle, and monitor and manage the operation of your Tanzu Kubernetes Grid Integrated Edition deployment. For information about how to use the TKGI Management Console after deployment, see the following topics:

- [Create and Manage Clusters in the Management Console](#)
- [Monitor and Manage Tanzu Kubernetes Grid Integrated Edition in the Management Console](#)
- [Troubleshooting Tanzu Kubernetes Grid Integrated Edition Management Console](#)

Prerequisites for Tanzu Kubernetes Grid Integrated Edition

Management Console Deployment

VMware Tanzu Kubernetes Grid Integrated Edition Management Console is provided as an OVA template that requires at a minimum the vSphere resources described in [Virtual Infrastructure Prerequisites](#).

For more information, see [When Should I Use Tanzu Kubernetes Grid Integrated Edition Management Console?](#)

Network Configurations

Tanzu Kubernetes Grid Integrated Edition Management Console provides 3 network configuration options for your Tanzu Kubernetes Grid Integrated Edition deployments. Each network configuration option has specific prerequisites.

- **Bring your own topology:** Deploy Tanzu Kubernetes Grid Integrated Edition to an existing NSX-T Data Center network that you have fully configured yourself. See [Prerequisites for a Bring Your Own Topology Deployment to NSX-T Data Center](#).
- **Automated NAT deployment:** Deploy Tanzu Kubernetes Grid Integrated Edition to an existing NSX-T Data Center network that you have not fully set up, that Tanzu Kubernetes Grid Integrated Edition Management Console helps to configure for you. See [Prerequisites for an Automated NAT Deployment to NSX-T Data Center](#).
- **vSphere Without NSX-T:** Deploy Tanzu Kubernetes Grid Integrated Edition to vSphere without an NSX-T network that Tanzu Kubernetes Grid Integrated Edition Management Console provisions for you. See [Prerequisites for vSphere Without an NSX-T Network](#).

For the list of firewall ports that must be open for an Tanzu Kubernetes Grid Integrated Edition Management Console deployment, see [Firewall Ports and Protocols Requirements for Tanzu Kubernetes Grid Integrated Edition Management Console](#).

When your environment meets the prerequisites for vSphere and for your chosen type of networking, you can [Deploy the Tanzu Kubernetes Grid Integrated Edition Management Console](#).

Virtual Infrastructure Prerequisites

The vSphere environment to which you deploy the management console OVA requires the following configuration:

- CPU: 2
- RAM: 8 GB
- Disk: 40 GB
- Virtual NIC (vNIC) should be assigned to a network with connectivity to vCenter and NSX Datacenter Manager, if you are using NSX-T Data Center as the container networking interface for Tanzu Kubernetes Grid Integrated Edition.



Note: The OVA requirements described here are the minimum supported configuration.

The following vSphere clusters must exist in the target vCenter Server datacenter before you can deploy Tanzu Kubernetes Grid Integrated Edition from the management console:

- Management cluster for TKGI Management Plane components.
- At least one compute cluster for Kubernetes Cluster nodes, with the recommendation being to deploy more than one, for high-availability purposes.

For information about the supported versions of vSphere, see the [release notes](#).

Firewall Ports and Protocols Requirements for Tanzu Kubernetes Grid Integrated Edition Management Console

Firewalls and security policies are used to filter traffic and limit access in environments with strict inter-network access control policies.

Apps frequently require the ability to pass internal communication between system components on different networks and require one or more conduits through the environment's firewalls. Firewall rules are also required to enable interfacing with external systems such as with enterprise apps or apps and data on the public Internet.

For Tanzu Kubernetes Grid Integrated Edition on vSphere, it is recommended to deactivate security policies that filter traffic between the networks supporting the system. To secure the environment and grant access between system components with Tanzu Kubernetes Grid Integrated Edition, use one of the following methods:

- Enable access to apps through standard Kubernetes load-balancers and ingress controller types. This enables you to designate specific ports and protocols as a firewall conduit.
- Enable access using the NSX-T load balancer and ingress. This enables you to configure external addresses and ports that are automatically mapped and resolved to internal/local addresses and ports.

If you are unable to implement your security policy using these methods, refer to the table below, which identifies the flows between the system components in an Tanzu Kubernetes Grid Integrated Edition Management Console deployment.



Notes: The Source Component is IP address of the Tanzu Kubernetes Grid Integrated Edition Management Console VM.

In a standard Tanzu Kubernetes Grid Integrated Edition deployment, it is assumed that Ops Manager and BOSH are already deployed before you deploy Tanzu Kubernetes Grid Integrated Edition. This is not the case with Tanzu Kubernetes Grid Integrated Edition deployments from the management console, in which you do not know the IP addresses in the deployment network that will be assigned to TKGI API VM, BOSH VM, and Ops Manager VM. As a consequence, it is recommended to create a firewall rule that allows access by the management console VM to the entire deployment subnet.

Source Component	Destination Component	Destination Protocol	Destination Port	Service
------------------	-----------------------	----------------------	------------------	---------

Management Console VM	All System Components	TCP	22	ssh
Management Console VM	All System Components	TCP	80	http
Management Console VM	All System Components	TCP	443	https
Management Console VM	Cloud Foundry BOSH Director	TCP	25555	bosh director rest api
Management Console VM	DNS validation for Ops Manager	TCP	53	netcat
Management Console VM	Kubernetes Cluster API Server - LB VIP	TCP	8443	httpsca
Management Console VM	Pivotal Cloud Foundry Operations Manager	TCP	22	ssh
Management Console VM	Pivotal Cloud Foundry Operations Manager	TCP	443	https
Management Console VM	TKGI Controller	TCP	9021	tkgi api server
Management Console VM	vCenter Server	TCP	443	https

Prerequisites for a Bring Your Own Topology Deployment to VMware NSX

This topic provides requirements for installing the Tanzu Kubernetes Grid Integrated Edition Management Console (TKGI MC) in a bring your own topology environment.

Overview

A bring your own topology environment is an NSX-T Data Center instance that you have fully configured yourself for use with Tanzu Kubernetes Grid Integrated Edition. For example, an NSX-T Data Center instance that you have used in a previous deployment of Tanzu Kubernetes Grid Integrated Edition.

The following objects must be in place before you start a production deployment.

- 3 NSX Manager Nodes deployed
- NSX Management Cluster formed
- Virtual IP address assigned for Management Cluster or load balancer

For information about the supported versions of NSX-T Data Center, see the [release notes](#).



Note: In BYOT mode, Tanzu Kubernetes Grid Integrated Edition Management Console automatically retrieves the tier0 HA mode from your NSX-T Data Center environment and creates NAT rules on the tier 0 or tier 1 router.

General Requirements

The following are requirements for deploying the Tanzu Kubernetes Grid Integrated Edition Management Console:

- Do not use the network on which you deploy the Tanzu Kubernetes Grid Integrated Edition Management Console VM as the network for the management plane when you deploy Tanzu Kubernetes Grid Integrated Edition.



Note: Using the same network for the management console VM and the management plane requires additional NSX-T Data Center configuration and is not recommended.

- A router. The TKGI Management Console supports Tier-0 and Tier-1 routers in either Active-Active or Active-Standby mode.



Note: If you are deploying Tanzu Kubernetes Grid Integrated Edition in a multiple-tier0 topology, additional post-deployment configuration of the management console VM is required. For information, see [Tanzu Kubernetes Grid Integrated Edition Management Console Cannot Retrieve Cluster Data in a Multi-Tier0 Topology](#) in *Troubleshooting the Management Console*.

- A logical switch on an NSX-T Virtual Distributed Switch (N-VDS) for use by the TKGI management plane is prepared. The switch must be either under the Tier-0 router, or under the Tier-1 router if the Tier-1 router is directly under the Tier-0 router.
- Edge Cluster with at least 2 NSX-T Data Center Edge Nodes configured with connectivity to an uplink network.
- Overlay Transport Zone created, with the edge nodes included.
- VLAN Transport Zone created, with the edge nodes included.
- MTU of all transport nodes and physical interfaces configured to 1600 or more.
- In NSX-T 3.0, you can create host transport nodes by installing NSX-T on a VDS switch. The NSX-T logical switch is created on the VDS as a distributed virtual port group.
- If your NSX-T Data Center environment uses custom certificates, obtain the CA certificate for NSX Manager.



Note: If NSX-T Data Center uses custom certificates and you do not provide the CA certificate for NSX Manager, Tanzu Kubernetes Grid Integrated Edition Management Console automatically generates one and registers it with NSX Manager. This can cause other services that are integrated with NSX Manager not to function correctly.

NSX-T Data Center Configuration Requirements

The following are NSX-T Data Center requirements for deploying the Tanzu Kubernetes Grid Integrated Edition Management Console:

- Virtual IP for the Tier-0 Router configured
- Floating IP Pool configured
- Pod IP Block ID created
- Node IP Block ID created
- Logical Switch configured for TKGI Management Plane
- Tier-1 Router configured and connected to the Tier-0 Router
- Routing for TKGI Floating IPs configured to point to the Tier-0 HA Virtual IP

Proof-of-Concept Deployments

The requirements above are for production environments. In proof-of-concept deployments one NSX Manager node is sufficient. The NSX management cluster and load balancer are also optional for proof-of-concept deployments.

Prerequisites for an Automated NAT Deployment to VMware NSX

An unprepared environment is an NSX-T Data Center instance that you have not already configured for use with Tanzu Kubernetes Grid Integrated Edition. Tanzu Kubernetes Grid Integrated Edition Management Console helps you to complete the configuration of an unprepared environment on vSphere, but the environment must meet certain infrastructure prerequisites.

- 3 NSX Manager Nodes deployed
- NSX Management Cluster formed
- Virtual IP address assigned for the Management Cluster or load balancer

For information about the supported versions of NSX-T Data Center, see the [release notes](#).

General Requirements

- Edge Cluster with at least 2 NSX-T Data Center Edge Nodes deployed and connectivity to an uplink network configured and verified
- Overlay Transport Zone created, with the edge nodes included
- VLAN Transport Zone created, with the edge nodes included
- MTU of all transport nodes and physical interfaces configured to 1600 or more
- In NSX-T 3.0, you can create host transport nodes by installing NSX-T on a VDS switch. The NSX-T logical switch is created on the VDS as a distributed virtual port group.
- Obtain the following IP addresses for the uplink network to use:
 - Subnet, subnet mask, gateway, and VLAN ID of the uplink network
 - Addresses within the uplink subnet for the Tier 0 uplinks

- Address to use for the HA Virtual IP on the Tier-0 router
- Obtain the following IP additional addresses:
 - CIDR ranges to use for deployment, pods, and nodes. This range of IP addresses must not be in conflict with any other workloads.
 - IP addresses of DNS and NTP servers
 - A range of 5 available floating IP addresses
- If your NSX-T Data Center environment uses custom certificates, obtain the CA certificate for NSX Manager



Note: If NSX-T Data Center uses custom certificates and you do not provide the CA certificate for NSX Manager, Tanzu Kubernetes Grid Integrated Edition Management Console automatically generates one and registers it with NSX Manager. This can cause other services that are integrated with NSX Manager not to function correctly.

Proof-of-Concept Deployments

The requirements above are for production environments. In proof-of-concept deployments one NSX Manager node is sufficient. The NSX management cluster and load balancer are also optional for proof-of-concept deployments. One NSX-T Data Center Edge node is sufficient for proof-of-concept deployments.

Prerequisites for vSphere (Antrea and Flannel Networking)

To deploy Tanzu Kubernetes Grid Integrated Edition on vSphere with Antrea or Flannel container networking, you can select the option for Tanzu Kubernetes Grid Integrated Edition Management Console to provision an Antrea or Flannel CNI for you during Tanzu Kubernetes Grid Integrated Edition deployment on vSphere.

Obtain the following IP addresses to use for deployment to vSphere without an NSX-T network:

- DNS server, subnet, subnet mask, and gateway of the network on which to deploy Tanzu Kubernetes Grid Integrated Edition
- DNS server, subnet, subnet mask, and gateway of the vSphere without NSX-T service network
- Subnet range and subnet mask for the Kubernetes pod and Kubernetes service networks

Deploy the Tanzu Kubernetes Grid Integrated Edition Management Console

This topic describes how to deploy the VMware Tanzu Kubernetes Grid Integrated Edition Management Console from the OVA template.

If you have deployed a previous version of VMware Tanzu Kubernetes Grid Integrated Edition Management Console, you can use the management console to upgrade it to a newer version. For information about upgrading, see [Upgrade Tanzu Kubernetes Grid Integrated Edition Management](#)

Console.

Prerequisites

- Download the OVA template from <https://downloads.vmware.com>.
- Use an account with vSphere administrator privileges to log in to vSphere using the vSphere Client.
- The vCenter Server instance must be correctly configured for Tanzu Kubernetes Grid Integrated Edition Management Console deployment. For information about the vCenter Server requirements, see [Virtual Infrastructure Prerequisites](#).

Step 1: Deploy the OVA Template

To deploy the Tanzu Kubernetes Grid Integrated Edition Management Console to vSphere, the procedure is as follows:

1. In the vSphere Client, right-click an object in the vCenter Server inventory, select **Deploy OVF template**, select **Local file**, and click **Browse** to navigate to your download of the OVA template.
2. Follow the installer prompts to perform basic configuration of the management console and to select the vSphere resources for it to use.
 - ◊ Accept or modify the management console VM name
 - ◊ Select the destination datacenter or folder
 - ◊ Select the destination cluster or resource pool for the management console VM
 - ◊ Accept the end user license agreements (EULA)
 - ◊ Select the disk format and destination datastore for the management console VM
3. On the **Select Networks** page, select a network port group to which to connect the management console VM.



Important: If you intend to deploy Tanzu Kubernetes Grid Integrated Edition in a bring your own topology NSX-T Data Center environment, do not use the network on which you deploy the Tanzu Kubernetes Grid Integrated Edition Management Console VM as the network for the management plane when you deploy Tanzu Kubernetes Grid Integrated Edition. Using the same network for the management console VM and the management plane requires additional NSX-T Data Center configuration and is not recommended.

4. On the **Customize template** page, expand **Appliance Configuration**.
 - ◊ Set the root password for the management console VM. Setting the root password for the VM is mandatory.
 - ◊ Optionally uncheck the **Permit Root Login** checkbox.



Note: If you uncheck the checkbox, you can permit root login later by editing

the settings of the management console VM.

The root password is the only mandatory option. If you want to use auto-generated certificates, DHCP networking, and you do not want to integrate with VMware vRealize Log Insight, click **Next** to start the OVA deployment. Otherwise, complete the remaining steps in this procedure.

- Configure the management console VM certificate, that is used by all of the services that run in the management console VM to authenticate connections.

To use auto-generated, self-signed certificates, leave the **Appliance TLS Certificate**, **Appliance TLS Certificate Key**, and **Certificate Authority Certificate** text boxes blank.

To use a custom certificate:

Paste the contents of the server certificate PEM file in the **Appliance TLS Certificate** text box.

```
-----BEGIN CERTIFICATE-----
appliance_certificate_contents
-----END CERTIFICATE-----
```

Paste the contents of the certificate key in the **Appliance TLS Certificate Key** text box. The management console VM supports unencrypted PEM encoded formats for TLS private keys.

```
-----BEGIN PRIVATE KEY-----
appliance_private_key_contents
-----END PRIVATE KEY-----
```

Paste the contents of the Certificate Authority (CA) file in the **Certificate Authority Certificate** text box.

```
-----BEGIN CERTIFICATE-----
root_CA_certificate_contents
-----END CERTIFICATE-----
```

To use a certificate that uses a chain of intermediate CAs, paste into the **Certificate Authority Certificate** text box the contents of a certificate chain PEM file. The PEM file must include a chain of the intermediate CAs all the way down to the root CA.

```
-----BEGIN CERTIFICATE-----
intermediate_CA_certificate_contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
intermediate_CA_certificate_contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
root_CA_certificate_contents
-----END CERTIFICATE-----
```

- Expand **Networking Properties** and optionally configure the networking for the management console VM.

To use DHCP, leave these properties blank.

- ◊ To set a static IP address on the management console VM, set the **Network IP Address**, **Network Netmask**, and **Default Gateway** settings.
- ◊ To configure DNS servers, set the **Domain Name Servers**, and **Domain Search Path** settings.
- ◊ To specify a fully qualified domain name (FQDN) for the management console VM, set the **FQDN** setting.
- ◊ If necessary, update **Docker Container Network Subnet** and **Docker Container Network Gateway**.

Services in the management console VM are deployed as Docker containers on a Docker bridge network.



Warning: If the default subnet CIDR 172.18.0.0/16 and gateway address 172.18.0.1 for this bridge network conflict with existing networks, you must update these values.

7. (Optional) Enter the host name and port for VMware vRealize Log Insight in the **Log Insight Server Host/IP** and **Log Insight Server Port** text boxes.

vRealize Log Insight gathers logs from the Tanzu Kubernetes Grid Integrated Edition Management Console VM itself. For vRealize Log Insight to gather logs from your Tanzu Kubernetes Grid Integrated Edition deployments, you must configure the connection when you deploy Tanzu Kubernetes Grid Integrated Edition from Tanzu Kubernetes Grid Integrated Edition Management Console.

8. Click **Next** to review the settings that you have made.

9. Click **Finish** to deploy the Tanzu Kubernetes Grid Integrated Edition Management Console.

Use the Recent Tasks panel at the bottom of the vSphere Client to check the status of the OVA import and deployment of the management console VM. The management console VM takes a few minutes to deploy.

If the management console VM fails to deploy, see [Troubleshooting](#).

Step 2: Log In to Tanzu Kubernetes Grid Integrated Edition Management Console

When the OVA deployment has completed successfully, you can access the management console.

1. In the vSphere Client, right-click the management console VM and select **Power > Power On**.
2. When the management console VM has booted, go to the **Summary** tab for the VM and copy its IP address.
3. Enter the management console VM IP address in a browser.
4. At the VMware Tanzu Kubernetes Grid Integrated Edition log in page, enter username `root` and the root password that you set when you deployed the OVA template.

Next Steps

You can now use Tanzu Kubernetes Grid Integrated Edition Management Console to deploy or upgrade Tanzu Kubernetes Grid Integrated Edition instances, either by using the configuration wizard or by importing an existing YAML configuration file.

- [Deploy Tanzu Kubernetes Grid Integrated Edition from the management console](#)
- [Upgrade Tanzu Kubernetes Grid Integrated Edition Management Console](#)

Deploy Tanzu Kubernetes Grid Integrated Edition from the Management Console

You can deploy a new VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) instance on vSphere either by using the VMware Tanzu Kubernetes Grid Integrated Edition Management Portal configuration wizard to guide you through the configuration process, or by importing an existing YAML configuration file into the YAML editor.

- [Deploy Tanzu Kubernetes Grid Integrated Edition by Using the Configuration Wizard](#)
- [Deploy Tanzu Kubernetes Grid Integrated Edition by Importing a YAML Configuration File](#)

If you deploy TKGI with plans that use Windows worker nodes, further configuration is required. See [Enable Plans with Windows Worker Nodes](#) for information about how to install a Windows Server stemcell and other necessary configuration actions that you must perform after you deploy Tanzu Kubernetes Grid Integrated Edition.

Additional Ops Manager Configurations

The Management Console takes values entered into the configuration wizard and sets them in Ops Manager, an older component that underlies TKGI and has its own UI. Because of this:

- The configuration wizard often makes using the Ops Manager UI unnecessary
- Values set in the configuration wizard override values set in Ops Manager

But there are configuration fields in Ops Manager that are not exposed by the configuration wizard, and which you can apply to your TKGI deployment. You can use the Ops Manager UI to configure the following, and the management console will not override their values:

- **BOSH Director** tile fields, described in [Configuring BOSH Director on vSphere in the Ops Manager Documentation](#):
 - **Director Config** pane:
 - **Custom SSH Banner**
 - **Identification Tags**
 - **Health Monitor**
 - **CredHub Encryption Provider**
 - **Blobstore Location**
 - **Database Location**
 - **BOSH DNS Config** pane: all fields
 - **Syslog** pane: all fields

- Resource Config pane:
 - Master Compilation Job
- TKGI tile tile fields, described in [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#):
 - Networking pane:
 - Enable outbound internet access
 - Host Monitoring pane:
 - Enable Telegraf Outputs?
 - In-Cluster Monitoring pane:
 - Enable node exporter on workers
 - Errands pane:
 - NSX-T Validation errand
 - Run smoke tests
 - Delete all clusters errand
- Harbor tile fields, described in [Installing and Configuring VMware Harbor Registry](#) in the *VMware Harbor Registry* documentation:
 - General pane:
 - Static IP Address
 - Wait time for Harbor Tile migration complete
 - Credentials pane:
 - Admin Password to run smoke test
 - Image Scanners pane:
 - Install Trivy
 - Errands pane:
 - smoke-testing
 - deregister Harbor UAA client
- Resource Config pane:
 - smoke-testing

Deploy Tanzu Kubernetes Grid Integrated Edition by Using the Configuration Wizard

This topic describes how to use the configuration wizard to deploy Tanzu Kubernetes Grid Integrated Edition (TKGI) on vSphere.

- For information about how to deploy Tanzu Kubernetes Grid Integrated Edition from a YAML, see [Deploy Tanzu Kubernetes Grid Integrated Edition by Importing a YAML Configuration File](#).
- For information about how to upgrade an existing deployment to this version, see [Upgrade Tanzu Kubernetes Grid Integrated Edition Management Console](#).

Prerequisites

- Deploy the Tanzu Kubernetes Grid Integrated Edition Management Console to vCenter Server.
- The vCenter Server instance must be correctly configured for Tanzu Kubernetes Grid Integrated Edition deployment. For information about the vCenter Server requirements, see [Virtual Infrastructure Prerequisites](#).
- Depending on the type of networking you want to use, your infrastructure must meet the appropriate prerequisites. For information about networking prerequisites, see the following topics:
 - [Prerequisites for an Automated NAT Deployment to NSX-T Data Center](#)
 - [Prerequisites for a Bring Your Own Topology Deployment to NSX-T Data Center](#)
 - [Prerequisites for vSphere Without an NSX-T Network](#)
- Log in to Tanzu Kubernetes Grid Integrated Edition Management Console.

Step 0: Launch the Configuration Wizard

1. On the VMware Tanzu Kubernetes Grid Integrated Edition landing page, click **Install**.

Welcome to VMware Tanzu Kubernetes Grid Integrated Edition

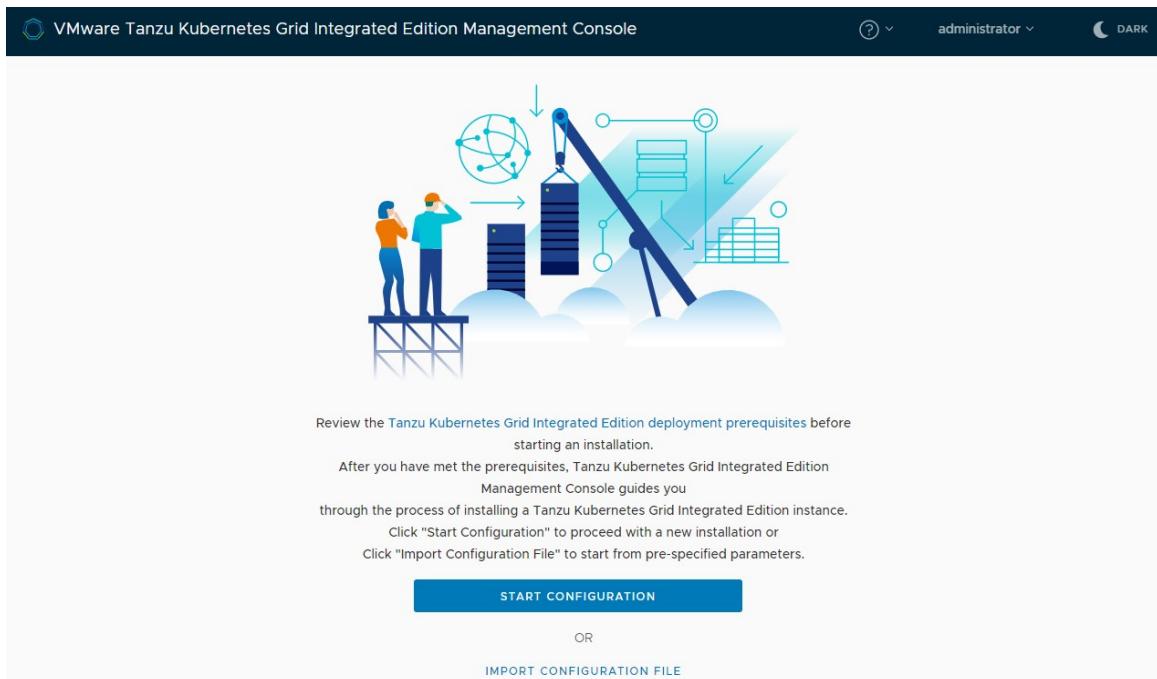
RESOURCE	VERSION
Ops Manager	2.10.5
Tanzu Kubernetes Grid Integrated Edition	1.10.0-build.22
Harbor	2.1.2-build.5

Do you want to install a new TKGI instance or upgrade an existing one?

INSTALL **UPGRADE**

[View a larger version of this image](#)

2. Click **Start Configuration**.



[View a larger version of this image](#)

To get help in the wizard at any time, click the ? icon at the top of the page and select **Help**, or click the **More Info...** links in each section to see help topics relevant to that section. Click the i icons for tips about how to fill in specific fields.

Step 1: Connect to vCenter Server

1. Enter the IP address or FQDN for the vCenter Server instance on which to deploy Tanzu Kubernetes Grid Integrated Edition.



Note: The FQDN for the vCenter Server cannot contain uppercase letters.

2. Enter the vCenter Single Sign On username and password for a user account that has vSphere administrator permissions.
3. Click **Connect**.
4. Select the datacenter in which to deploy Tanzu Kubernetes Grid Integrated Edition from the drop-down menu.



WARNING: Ideally, do not deploy TKGI from the management console to a datacenter that also includes TKGI instances that you deployed manually. If deploying management console and manual instances of TKGI to the same datacenter cannot be avoided, make sure that the TKGI instances that you deployed manually do not use the folder names `BoshVMFolder: pks_vms`, `BoshTemplateFolder: pks_templates`, `BoshDiskPath: pks_disk`. If a manual installation uses these folder names, the VMs that they contain will be deleted when you delete a TKGI instance from the management console.

5. Click **Next** to configure networking.

Step 2: Configure Networking

Provide connection information for the container networking interface to use with Tanzu Kubernetes Grid Integrated Edition. Tanzu Kubernetes Grid Integrated Edition Management Console provides 3 network configuration options for your Tanzu Kubernetes Grid Integrated Edition deployments. Each network configuration option has specific prerequisites.

- **Automated NAT deployment:** Deploy Tanzu Kubernetes Grid Integrated Edition to an existing NSX-T Data Center network that you have not fully set up, that Tanzu Kubernetes Grid Integrated Edition Management Console configures for you. See [Configure an Automated NAT Deployment to NSX-T Data Center](#) below for instructions.
- **Bring your own topology:** Deploy Tanzu Kubernetes Grid Integrated Edition to an existing NSX-T Data Center network that you have fully configured yourself. See [Configure a Bring Your Own Topology Deployment to NSX-T Data Center](#) below for instructions.
- **Antrea and Flannel:** Deploy Tanzu Kubernetes Grid Integrated Edition with either an Antrea or Flannel network that Tanzu Kubernetes Grid Integrated Edition Management Console provisions for you. See [Configure an Antrea or Flannel Network](#) below for instructions.

The screenshot shows the 'Networking' configuration step in the Tanzu Kubernetes Grid Integrated Edition Management Console. It lists four options: NSX-T Data Center (Automated NAT Deployment), NSX-T Data Center (Bring Your Own Topology), Flannel, and Antrea. A note at the bottom states: "Important: You cannot change the type of networking after you deploy Tanzu Kubernetes Grid Integrated Edition." A 'MORE INFO...' link is also present.

[View a larger version of this image.](#)



Important: You cannot change the type of networking after you deploy Tanzu Kubernetes Grid Integrated Edition.

Configure an Automated NAT Deployment to NSX-T Data Center

Provide information about an NSX-T Data Center network that you have not already configured for use with Tanzu Kubernetes Grid Integrated Edition. You provide information about your NSX-T Data Center setup, and Tanzu Kubernetes Grid Integrated Edition Management Console creates the necessary objects and configures them for you. Make sure that your NSX-T Data Center setup satisfies the [Prerequisites for an Automated NAT Deployment to NSX-T Data Center](#) before you begin.

1. Select the **NSX-T Data Center (Automated NAT Deployment)** radio button.
2. Configure the connection to NSX Manager.
 - Enter the IP address or FQDN of NSX Manager.
 - Enter the user name and password for an NSX administrator account.
3. Click **Connect**.

4. Enter information about the uplink network.

- ◊ **Uplink CIDR:** Enter a CIDR range within the uplink subnet for the Tier 0 uplinks, for example 10.40.206.0/24.
- ◊ **Gateway IP:** Enter the IP address for the gateway, for example 10.40.206.125.
- ◊ **VLAN ID:** Enter the VLAN ID within the range 0 to 4095, for example 1206.
- ◊ **Edge Node 1:** Select an Edge Node from the drop-down menu, for example `nsx-edge-1`.
- ◊ **TO Uplink 1 IP:** Enter the IP address of the Tier 0 uplink 1, for example 10.40.206.9.
- ◊ **Edge Node 2:** Select an Edge Node from the drop-down menu, for example `nsx-edge-2`. The second edge node is optional for proof-of-concept deployments, but it is strongly recommended for production deployments. To use only one edge node, set Edge Node two to `None`.
- ◊ **TO Uplink 2 IP:** Enter the IP address of the Tier 0 uplink 1, for example 10.40.206.11.
- ◊ **TO HA Virtual IP:** Enter the IP address for the HA Virtual IP, for example 10.40.206.24.

5. Optionally activate **Tier0 Active-Active Mode**.

By default, the management console sets the high availability (HA) mode of the tier 0 router to active-standby. You can optionally activate active-active mode on the tier 0 router, so that all NAT configuration moves from the tier 0 to the tier 1 router.

Uplink Network ⓘ

Uplink CIDR ⓘ 192.168.115.0/24	Gateway IP ⓘ 192.168.115.1	VLAN ID ⓘ 0
Edge Node 1 ⓘ nsxedge2.tkgi.vmware.local	TO Uplink 1 IP ⓘ 192.168.115.3	
Edge Node 2 ⓘ nsxedge3.tkgi.vmware.local	TO Uplink 2 IP ⓘ 192.168.115.4	TO HA Virtual IP ⓘ 192.168.115.2

The second edge node is optional, but it is recommended for production deployments. To use only one edge node, set Edge Node two to "None".

Tier0 Active Active Mode ⓘ

6. Enter information about the network resources for the Tanzu Kubernetes Grid Integrated Edition deployment to use.

- ◊ **Deployment CIDR:** Enter a CIDR range to use for Tanzu Kubernetes Grid Integrated Edition components, for example 10.192.182.1/22.
- ◊ **Deployment DNS:** Enter the IP address of the DNS server to use for deploying Tanzu Kubernetes Grid Integrated Edition components, for example 192.168.111.155.
- ◊ **NTP Server:** Enter the IP address of an NTP server.
- ◊ **Pod IP Block CIDR:** Enter a CIDR range to use for pods, with a maximum suffix of 24. For example 11.192.183.1/22.
- ◊ **Node IP Block CIDR:** Enter a CIDR range to use for nodes, with a maximum suffix of 22. For example 11.192.184.1/22.

- ❖ **Nodes DNS:** Enter the Domain Name Server used by the Kubernetes nodes.
- ❖ **Deployment Network Reserved IP Range:** Optionally enter a range of IP addresses in the **From** and **To** text boxes. No VMs are deployed in this range. You cannot modify reserved IP ranges after the initial deployment. You can specify additional reserved IP ranges by editing the YAML configuration for your deployment before you deploy it in [Step 10: Generate Configuration File and Deploy Tanzu Kubernetes Grid Integrated Edition](#).
- ❖ **Usable range of floating IPs:** Enter the floating IP range, for example **From** 192.168.160.100 **To** 192.168.160.199. Click **Add Range** to add more IP ranges.

Network Resources ⓘ

Deployment CIDR ⓘ 10.192.182.1/22	Deployment DNS ⓘ 192.168.111.155	NTP Server ⓘ 192.168.115.2
Pod IP Block CIDR ⓘ 11.192.183.1/22	Node IP Block CIDR ⓘ 11.192.184.1/22	Nodes DNS ⓘ 192.168.111.155
Deployment Network Reserved IP Range ⓘ From Optional		To Optional
Usable Range of Floating IPs ⓘ From 192.168.160.100		To 192.168.160.199
ADD RANGE		

7. Optionally activate **Manage certificates manually for NSX** if NSX Manager uses a custom CA certificate.

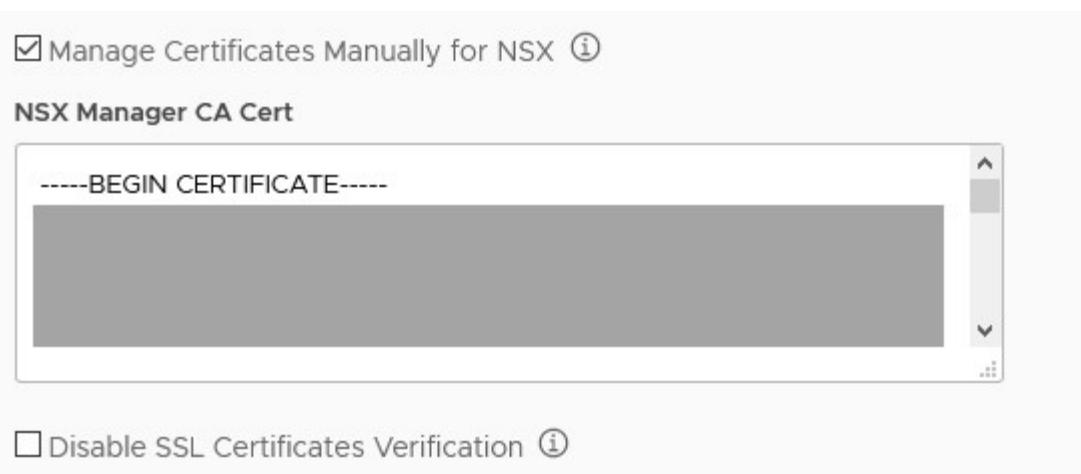


Important: If NSX-T Data Center uses custom certificates and you do not provide the CA certificate for NSX Manager, Tanzu Kubernetes Grid Integrated Edition Management Console automatically generates one and registers it with NSX Manager. This can cause other services that are integrated with NSX Manager not to function correctly. If you have manually deployed TKG instances to the same datacenter as the one to which you are deploying this instance, you must select **Manage certificates manually for NSX** and enter the current NSX-T manager CA certificate.

Enter the contents of the CA certificate in the **NSX Manager CA Cert** text box:

```
-----BEGIN CERTIFICATE-----
nsx_manager_CA_certificate_contents
-----END CERTIFICATE-----
```

If you do not select **Manage certificates manually for NSX**, the management console generates a certificate for you.



8. Optionally activate **Disable SSL certificates verification** to allow unsecured connections to NSX Manager.
9. Click **Next** to configure identity management.

For the next steps, see [Configure Identity Management](#).

Configure a Bring Your Own Topology Deployment to NSX-T Data Center

Provide information about an NSX-T Data Center network that you have already fully configured for use with Tanzu Kubernetes Grid Integrated Edition. Make sure that your NSX-T Data Center setup satisfies the [Prerequisites for a Bring Your Own Topology Deployment to NSX-T Data Center](#) before you begin.

1. Select the **NSX-T Data Center (Bring Your Own Topology)** radio button.
2. Configure the connection to NSX Manager.
 - Enter the IP address or FQDN of the NSX Manager.
 - Enter the user name and password for an NSX administrator account.
3. Click **Connect**.
4. Use the drop-down menus to select existing network resources for each of the following items.
 - **Network for TKGI Management Plane:** Select the name of an opaque network on an NSX-T Virtual Distributed Switch (N-VDS).



Important: Do not use the network on which you deployed the Tanzu Kubernetes Grid Integrated Edition Management Console VM as the network for the management plane. Using the same network for the management console VM and the management plane requires additional NSX-T Data Center configuration and is not recommended.

- **Pod IP Block ID:** Select the UUID for the IP block to use for Kubernetes pods.
- **Node IP Block ID:** Select the UUID for the IP block to use for Kubernetes nodes.
- **T0 Router ID:** Select the UUID for the Tier-0 Logical Router configured in NSX-T

Data Center.

- ◊ **Floating IP Pool ID:** Select the UUID for the Floating IP Pool.
5. Enter IP addresses for the following resources.
 - ◊ **Nodes DNS:** Enter the IP address for the DNS server to use for Kubernetes nodes and pods.
 - ◊ **Deployment DNS:** Enter the IP address for the DNS server to use for the TKGI control plane VMs, for example 192.168.111.155.
 - ◊ **NTP Server:** Enter the IP address of an NTP server.
 - ◊ **Deployment Network Reserved IP Range:** Optionally enter a range of IP addresses in the **From** and **To** text boxes. No VMs are deployed in this range. You cannot modify reserved IP ranges after the initial deployment. You can specify additional reserved IP ranges by editing the YAML configuration for your deployment before you deploy it in [Step 10: Generate Configuration File and Deploy Tanzu Kubernetes Grid Integrated Edition](#).

Network Resources		
Network for TKGI Management Plane	Pod IP Block ID	Node IP Block ID
Network 1	IP Block ip-blk-1	IP Block ip-blk-2
TO Router ID	Floating IP Pool ID	Nodes DNS
TO Router 002	floating-ip-pool	192.168.111.155
Deployment DNS	NTP Server	
192.168.111.155	192.168.115.2	
Deployment Network Reserved IP Range		
From Optional	To Optional	

6. If you are using the NSX Policy API, select this option. See [Considerations for Using the NSX Policy API with TKGI](#).
7. Optionally deactivate **NAT Mode** to implement a routable (No-NAT) topology.

Tanzu Kubernetes Grid Integrated Edition supports NAT topologies, No-NAT with logical switch (NSX-T) topologies, No-NAT with virtual switch (VSS/VDS) topologies, and multiple tier-0 routers for tenant isolation. For information about implementing a routable topology, see [No-NAT Topology in NSX-T Deployment Topologies for Tanzu Kubernetes Grid Integrated Edition](#).

8. If you left NAT mode activated, optionally activate **Hybrid NAT Mode**.

If you activate hybrid NAT mode, the Tanzu Kubernetes Grid Integrated Edition management plane runs on a routable subnet but the cluster node network uses a non-routable subnet.

9. Optionally activate **Manage certificates manually for NSX** if NSX Manager uses a custom CA certificate.



Important: If NSX-T Data Center uses custom certificates and you do not provide the CA certificate for NSX Manager, Tanzu Kubernetes Grid

Integrated Edition Management Console automatically generates one and registers it with NSX Manager. This can cause other services that are integrated with NSX Manager not to function correctly. If you have manually deployed TKGI instances to the same datacenter as the one to which you are deploying this instance, you must select **Manage certificates manually for NSX** and enter the current NSX-T manager CA certificate.

Enter the contents of the CA certificate in the **NSX Manager CA Cert** text box:

```
-----BEGIN CERTIFICATE-----
nsx_manager_CA_certificate_contents
-----END CERTIFICATE-----
```

If you do not select **Manage certificates manually for NSX**, the management console generates a certificate for you.

Manage Certificates Manually for NSX ⓘ

NSX Manager CA Cert

```
-----BEGIN CERTIFICATE-----
```



Disable SSL Certificates Verification ⓘ

10. Optionally activate **Disable SSL certificates verification** to allow unsecured connections to NSX Manager.
11. Click **Next** to configure identity management.

For the next steps, see [Configure Identity Management](#).

Configure an Antrea or Flannel Network

Provide networking information so that Tanzu Kubernetes Grid Integrated Edition Management Console can provision an Antrea or Flannel network for you during deployment. Make sure that you have the information listed in [Prerequisites for vSphere Without an NSX-T Network](#) before you begin.

1. Select either the **Antrea** or **Flannel** radio button.

The options for Antrea and Flannel networking are identical.

2. Configure the Deployment Network Resource options.
 - ◊ **Deployment Network:** Select a vSphere network on which to deploy Tanzu Kubernetes Grid Integrated Edition.
 - ◊ **Deployment Network CIDR:** Enter a CIDR range to use for Tanzu Kubernetes Grid Integrated Edition components, for example 10.192.182.1/22.

- ◊ **Deployment Network Gateway IP:** Enter the IP address for the gateway for the deployment network, for example 10.192.182.1.
- ◊ **Deployment DNS:** Enter the IP address for the deployment network DNS server, for example 192.168.111.155.
- ◊ **Deployment Network Reserved IP Range:** Optionally enter a range of IP addresses in the **From** and **To** text boxes. No VMs are deployed in this range. You cannot modify reserved IP ranges after the initial deployment. You can specify additional reserved IP ranges by editing the YAML configuration for your deployment before you deploy it in [Step 10: Generate Configuration File and Deploy Tanzu Kubernetes Grid Integrated Edition](#).

Deployment Network Resource ⓘ

Deployment Network ⓘ
Network 1
Deploy Network CIDR ⓘ
172.16.100.0/24
Deployment Network Gateway IP ⓘ
172.16.100.1
Deployment DNS ⓘ
192.168.111.155
Deployment Network Reserved IP Range ⓘ
From
Optional
To
Optional

3. Configure the Service Network Resource options.

- ◊ **Service Network:** Select a vSphere network to use as the service network.
- ◊ **Service Network CIDR:** Enter a CIDR range to use for the service network, for example 10.192.182.1/23.
- ◊ **Service Network Gateway IP:** Enter the IP address for the gateway for the service network.
- ◊ **Service DNS:** Enter the IP address for the service network DNS server, for example 192.168.111.155.
- ◊ **Service Network Reserved IP Range:** Optionally enter a range of IP addresses in the **From** and **To** text boxes. No VMs are deployed in this range. You cannot modify the reserved IP range after the initial deployment. You can specify additional reserved IP ranges by editing the YAML configuration for your deployment before you deploy it in [Step 10: Generate Configuration File and Deploy Tanzu Kubernetes Grid Integrated Edition](#).
- ◊ **NTP Server:** Enter the IP address of an NTP server.

Service Network Resource ⓘ

Service Network ⓘ
Network 1

Service Network CIDR ⓘ
10.192.182.1/23

Service Network Gateway IP ⓘ
10.0.0.1

Service DNS ⓘ
192.168.111.155

Service Network Reserved IP Range ⓘ
From: Optional
To: Optional

NTP Server ⓘ
192.168.115.2

4. Configure the Kubernetes network options.

- **Pod Network CIDR:** Enter a CIDR range to use for pods, for example 11.192.182.1/31.
- **Service Network CIDR:** Enter a CIDR range to use for the Kubernetes services, for example 10.192.182.1/23.

Kubernetes Pod Network ⓘ

Pod Network CIDR ⓘ
11.192.182.1/31

Kubernetes Service Network ⓘ

Service Network CIDR ⓘ
10.192.182.1/23

NEXT

5. Click **Next** to configure identity management.

Step 3: Configure Identity Management

Tanzu Kubernetes Grid Integrated Edition Management Console provides 3 identity management options for your Tanzu Kubernetes Grid Integrated Edition deployments.

- A local database of users. See [Use a Local Database](#) below for instructions.
- Connect to an external Active Directory or LDAP server. See [Use an External LDAP Server](#) below for instructions.
- Connect to a SAML identity provider. See [Use a SAML Identity Provider](#) below for instructions.

Use a Local Database

You can manage users by using a local database that is created during Tanzu Kubernetes Grid Integrated Edition deployment. After deployment, you can add users and groups to the database and assign roles to them in the Identity Management view of the Tanzu Kubernetes Grid Integrated Edition Management Console.

1. Select the **Local user database** radio button.
2. In the **TKGI API FQDN** text box, enter an address for the TKGI API Server VM, for example `api.tkgi.example.com`.



Note: The FQDN for the TKGI API cannot contain uppercase letters.

For the next steps, see [Optionally Configure UAA and Custom Certificates](#).

Use an External LDAP Server

Provide information about an existing external Active Directory or LDAP server.

1. Select the **AD/LDAP** radio button.
2. For **AD/LDAP Endpoint**, select **Idap** or **Idaps** from the drop-down menu and enter the IP address and port of the AD or LDAP server.
3. Enter the username and password to use to connect to the server.
4. Enter the remaining details for your server:
 - ◊ **User Search Base:** Enter the location in the AD/LDAP directory tree where user search begins. For example, a domain named `cloud.example.com` might use `ou=Users,dc=example,dc=com`.
 - ◊ **User Search Filter:** Enter a string to use for user search criteria. For example, the standard search filter `cn=Smith` returns all objects with a common name equal to `Smith`. Use `cn={0}` to return all LDAP objects with the same common name as the username.
 - ◊ **LDAP Referrals:** Select how to handle references to alternate locations in which AD/LDAP requests can be processed:
 - Automatically follow referrals
 - Ignore referrals
 - Abort authentication
 - ◊ **Group Search Base:** Optionally enter the location in the AD/LDAP directory tree where group search begins. For example, a domain named `cloud.example.com` might use `ou=Groups,dc=example,dc=com`.
 - ◊ **Group Search Filter:** Enter a string that defines AD/LDAP group search criteria, such as `member={0}`.
 - ◊ **Group Max Search Depth:** Enter the LDAP group search depth, such as `1`. Allowed values are between `1` and `10`. The default value is `1`, which limits queering under the searchBase to one subtree. Values greater than `1` activate nested group searching. If the searchBase in your LDAP groups includes more than one subtree, for example: `ou=Groups,ou=Client,ou=DE,dc=fs01,dc=vwf,dc=vwfs-ad`, increase the **Group Max**

Search Depth value to support searching all subtrees in your groups.



Note: Increasing the LDAP group search depth impacts performance.

- ◊ **External Groups Whitelist:** Optionally enter a comma-separated list of group patterns to be populated in the user's `id_token`.
 - ◊ **Email Attribute:** Enter the attribute name in the AD/LDAP directory that contains user email addresses. For example, `mail`.
 - ◊ **Email Domains:** Optionally enter a comma-separated list of the email domains for external users who can receive invitations to Tanzu Kubernetes Grid Integrated Edition.
 - ◊ **First Name Attribute:** Optionally enter the attribute name in the AD/LDAP directory that contains user first names, for example `cn`.
 - ◊ **Last Name Attribute:** Optionally enter the attribute name in the AD/LDAP directory that contains user last names. for example `sn`.
 - ◊ **Server SSL Certificate:** If you are using an LDAPS endpoint, paste the contents of the LDAP server certificate certificate into the text box.
5. Optionally click the **Test LDAP Server** button to test the connection that you have configured.
 6. In the **TKGI API FQDN** text box, enter an address for the TKGI API Server VM, for example `api.tkgi.example.com`.



Note: The FQDN for the TKGI API cannot contain uppercase letters.

3. Identity Provide the identity management endpoint settings for Kubernetes clusters

Identity Management Source

[MORE INFO...](#)

Local User Database AD/LDAP SAML Identity Provider

AD/LDAP Endpoint <small>①</small>	Username <small>①</small>	Password <small>①</small>
ldap // 10.192.168.55 : 389	admin
User Search Base <small>①</small>	User Search Filter <small>①</small>	LDAP Referrals <small>①</small>
ou=Users,dc=example,dc=	cn=(0)	Automatically Follow Any Referrals
Group Search Base <small>①</small>	Group Search Filter <small>①</small>	External Groups Whitelist <small>①</small>
ou=Groups,dc=example,dc=	member=(0)	Optional
Email Attribute <small>①</small>	Email Domains <small>①</small>	
mail	Optional	
First Name Attribute <small>①</small>	Last Name Attribute <small>①</small>	
cn	sn	

TEST LDAP SERVER

For the next steps, see [Optionally Configure UAA and Custom Certificates](#).

Use a SAML Identity Provider

You can configure Tanzu Kubernetes Grid Integrated Edition so that Kubernetes authenticates users

against a SAML identity provider. Before you configure a SAML identity provider, you must configure your identity provider to designate Tanzu Kubernetes Grid Integrated Edition as a service provider. For information about how to configure Okta and Azure Active Directory, see the following topics:

- [Configuring Okta as a SAML Identity Provider](#)
- [Configuring Azure Active Directory as a SAML Identity Provider](#)

After you have configured your identity provider, enter information about the provider in Tanzu Kubernetes Grid Integrated Edition Management Console.

1. Select the **SAML Identity Provider** radio button.
2. For **Provider Name**, enter a unique name you create for the Identity Provider. This name can include only alphanumeric characters, +, _, and -. You must not change this name after deployment because all external users use it to link to the provider.
3. For **Display Name**, enter a display name for your provider. The display name appears as a link on your login page.
4. Enter the metadata from your identity provider either as XML or as a URL.
 - ◊ Download your identity provider metadata and paste the XML into **Provider Metadata**.
 - ◊ If your identity provider exposes a metadata URL, enter it in **Provider Metadata URL**.
5. For **Name ID Format**, select the name identifier format for your SAML identity provider. This translates to `username` on TKGI. The default is `Email Address`.
6. For **First Name Attribute** and **Last Name Attribute**, enter the attribute names in your SAML database that correspond to the first and last names in each user record. These fields are case sensitive.
7. For **Email Attribute**, enter the attribute name in your SAML assertion that corresponds to the email address in each user record, for example, `EmailID`. This field is case sensitive.
8. For **External Groups Attribute**, enter the attribute name in your SAML database for your user groups. This field is case sensitive. To map the groups from the SAML assertion to admin roles in Tanzu Kubernetes Grid Integrated Edition, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an External LDAP Group](#).
9. By default, all SAML authentication requests from Tanzu Kubernetes Grid Integrated Edition are signed, but you can optionally deactivate **Sign Authentication Requests**. If you deactivate this option, you must configure your identity provider to verify SAML authentication requests.
10. To validate the signature for the incoming SAML assertions, activate **Required Signed Assertions**. If you activate this option, you must configure your Identity Provider to send signed SAML assertions.
11. For **Signature Algorithm**, choose an algorithm from the drop down to use for signed requests and assertions. The default value is SHA256.

12. In the **TKGI API FQDN** text box, enter an address for the TKGI API Server VM, for example `api.tkgi.example.com`.



Note: The FQDN for the TKGI API cannot contain uppercase letters.

3. Identity		Provide the identity management endpoint settings for Kubernetes clusters
Identity Management Source <input type="radio"/> Local User Database <input type="radio"/> AD/LDAP <input checked="" type="radio"/> SAML Identity Provider		
Provider Name ⓘ okta	Display Name ⓘ Log in with Okta IDP	Name ID Format ⓘ Email Address
Provider Metadata (or use metadata URL) <small>Optional</small>		Provider Metadata URL ⓘ https://dev-518997.okta.co
First Name Attribute ⓘ <small>Optional</small>	Last Name Attribute ⓘ <small>Optional</small>	Email Attribute ⓘ <small>Optional</small>
External Groups Attribute ⓘ <small>groups</small>		
<input checked="" type="checkbox"/> Sign Authentication Requests <input checked="" type="checkbox"/> Required Signed Assertions		
Signature Algorithm ⓘ <small>SHA256</small>		
PKS API FQDN ⓘ <small>api.pks.local</small>		

For the next steps, see [Optionally Configure UAA and Custom Certificates](#).

Optionally Configure UAA and Custom Certificates

However you manage identities, you can use OpenID Connect (OIDC) to instruct Kubernetes to verify end-user identities based on authentication performed by a User Account and Authentication (UAA) server. Using OIDC lets you set up an external IDP, such as Okta, to authenticate users who access Kubernetes clusters with `kubectl`. If you activate OIDC, administrators can grant namespace-level or cluster-wide access to Kubernetes end users. If you do not activate OIDC, you must use service accounts to authenticate `kubectl` users.



Note: You cannot activate OIDC if you intend to integrate Tanzu Kubernetes Grid Integrated Edition with VMware vRealize Operations Management Pack for Container Monitoring.

1. Optionally select **Configure created clusters to use UAA as the OIDC provider** and provide the following information.
 - ❖ **UAA OIDC Groups Claim:** Sets the `--oidc-groups-claim` flag on the kube-api server. Enter the name of your groups claim. This is used to set a user's group in the JSON Web Token (JWT) claim. The default value is `roles`.
 - ❖ **UAA OIDC Groups Prefix:** Sets the `--oidc-groups-prefix` flag. Enter a prefix for

your groups claim. This prevents conflicts with existing names. For example, if you enter the prefix `oidc:`, UAA creates a group name like `oidc:developers`.

- **UAA OIDC Username Claim:** Sets the `--oidc-username-claim` flag. Enter the name of your username claim. This is used to set a user's username in the JWT claim. The default value is `user_name`. Depending on your provider, admins can enter claims besides `user_name`, such as `email` or `name`.
- **UAA OIDC Username Prefix:** Sets the `--oidc-username-prefix` flag. Enter a prefix for your username claim. This prevents conflicts with existing names. For example, if you enter the prefix `oidc:`, UAA creates a username like `oidc:admin`.

Configure Created Clusters to Use UAA as the OIDC Provider

UAA OIDC Groups Claim ⓘ <input type="text" value="roles"/>	UAA OIDC Groups Prefix ⓘ <input type="text" value="oidc:"/>
UAA OIDC Username Claim ⓘ <input type="text" value="user_name"/>	UAA OIDC Username Prefix ⓘ <input type="text" value="oidc:"/>

Manage Certificates Manually for TKGI API ⓘ

NEXT

2. Optionally select **Manage Certificates Manually for TKGI API** to generate and upload your own certificates for the TKGI API Server.

If you do not select this option, the management console creates auto-generated, self-signed certificates.

Enter the contents of the certificate in the **TKGI API Certificate** text box:

```
-----BEGIN CERTIFICATE-----
tkgi_api_certificate_contents
-----END CERTIFICATE-----
```

Enter the contents of the certificate key in the **Private Key PEM** text box:

```
-----BEGIN RSA PRIVATE KEY-----
tkgi_api_private_key_contents
-----END RSA PRIVATE KEY-----
```

3. Click **Next** to configure availability zones.

Step 4: Configure Availability Zones

Availability zones specify the compute resources for Kubernetes cluster deployment. Availability zones are a BOSH construct, that in Tanzu Kubernetes Grid Integrated Edition deployments to

vSphere correspond to vCenter Server clusters, host groups, and resource pools. Availability zones allow you to provide high-availability and load balancing to applications. When you run more than one instance of an application, those instances are balanced across all of the availability zones that are assigned to the application. You must configure at least one availability zone. You can configure multiple additional availability zones.



Note: If you select a cluster as an availability zone, Tanzu Kubernetes Grid Integrated Edition Management Console sets the DRS VM-host affinity rule on that cluster to **MUST**. If you select a host group as an availability zone, Tanzu Kubernetes Grid Integrated Edition Management Console sets the DRS VM-host affinity rule on that group to **SHOULD**.

1. In the **Name** field, enter a name for the availability zone.
2. Optionally select **This is the management availability zone**.
The management availability zone is the availability zone in which to deploy the TKGI Management Plane. The management plane consists of the TKGI API VM, Ops Manager, BOSH Director, and Harbor Registry. You can only designate one availability zone as the management zone. If you do not designate an availability zone as the management zone, Tanzu Kubernetes Grid Integrated Edition Management Console selects the first one.
3. In the **Compute Resource** tree, select clusters, host groups, or resource pools for this availability zone to use.
4. Click **Save Availability Zone**.

The screenshot shows the '4. Availability Zones' configuration screen. The 'az1' availability zone is selected. Under 'Compute Resource', 'Respool 1' is checked as the management availability zone. A tree view shows nested clusters, folders, and resource pools. Buttons for 'SAVE AVAILABILITY ZONE', 'DELETE', 'NEXT', and 'ADD AVAILABILITY ZONE' are visible.

5. Optionally click **Add Availability Zone** to add another zone.
You can only select resources that are not already included in another zone. You can create multiple availability zones.
6. Click **Save Availability Zone** for every additional availability zone that you create.

7. Click **Next** to configure storage.

Step 5: Configure Resources and Storage

Resource Settings allow you to configure the resources that are allocated to the VM on which the Tanzu Kubernetes Grid Integrated Edition API and other component services, such as UAA, run. Allocate resources according to the workloads that TKGI will run. You can also activate High Availability for the TKGI Database and deploy multiple instances of the TKGI API VM.

Tanzu Kubernetes Grid Integrated Edition, the MySQL database runs on a separate VM to the Tanzu Kubernetes Grid Integrated Edition API and other components.

You must also designate the datastores to use for the different types of storage required by your Tanzu Kubernetes Grid Integrated Edition deployment.

- Ephemeral storage is used to contain the files for ephemeral VMs that Tanzu Kubernetes Grid Integrated Edition creates during installation, upgrade, and operation. Ephemeral VMs are automatically created and deleted as needed.
- Permanent storage is used for permanent Tanzu Kubernetes Grid Integrated Edition data.
- Kubernetes persistent volume storage is used to store Kubernetes persistent volumes, for use in stateful applications.

You can use different datastores for the storage of permanent and ephemeral data. If you deactivate the permanent storage option, Tanzu Kubernetes Grid Integrated Edition uses the ephemeral storage for permanent data. For information about when it is appropriate to share the ephemeral, permanent, and persistent volume datastores or use separate ones, see [PersistentVolume Storage Options on vSphere](#).

You can use VMware vSAN, Network File Share (NFS), or VMFS storage for ephemeral, permanent, and Kubernetes persistent storage. Datastores can only be selected if their minimum capacity is greater than 250GB.

1. Optionally toggle **TKGI Database** to activate database HA mode.
2. For **TKGI Database Persistent Disk Size**, select the size of the persistent disk for the Tanzu Kubernetes Grid Integrated Edition MySQL database VM.
Set the TKGI Database Persistent Disk Size according to the amount of data that you expect the cluster workload to store.
3. Use the **TKGI Database VM Type** drop-down menu to select from different combinations of CPU, RAM, and storage for the Tanzu Kubernetes Grid Integrated Edition MySQL database VM.
Choose the configuration for the TKGI Database VM depending on the volume of database operations that it will run.
4. Use the **TKGI API Instances** drop-down menu to select 1, 2, or 3 instances of the TKGI API VM.
5. For **TKGI API Persistent Disk Size**, select the size of the persistent disk for the Tanzu Kubernetes Grid Integrated Edition API VM.
Set the TKGI API Persistent Disk Size according to the number of pods that you expect the cluster workload to run continuously. It is recommended to allocate 10GB for every 500 pods. For example:

- ◊ For 1000 pods, allocate 20GB
 - ◊ For 10,000 pods, allocate 200GB
 - ◊ For 50,000 pods, allocate 1TB
6. Use the **TKGI API VM Type** drop-down menu to select from different combinations of CPU, RAM, and storage for the Tanzu Kubernetes Grid Integrated Edition API VM. Choose the configuration for the API VM depending on the expected CPU, memory, and storage consumption of the workloads that it will run. For example, some workloads might require a large compute capacity but relatively little storage, while others might require a large amount of storage and less compute capacity.

The screenshot shows the '5. Resources & Storage' configuration page. At the top, there's a header bar with the title 'Configure resources and choose ephemeral, permanent and Kubernetes storage resources'. Below this is a section titled 'Resource Settings' with the sub-instruction 'Configure resources for the Tanzu Kubernetes Grid Integrated Edition deployments'. There are two main sections: 'TKGI Database' and 'TKGI API Instances'. Under 'TKGI Database', there is a toggle switch for 'Enable Database HA Mode (BETA)' and a dropdown for 'Persistent Disk Size' set to '10 GB'. To its right is a dropdown for 'TKGI Database VM Type' set to 'large.disk (cpu: 2, ram: 8 GB, disk: 64 GB)'. Under 'TKGI API Instances', there is a dropdown for 'Instances' set to '2' and a dropdown for 'Persistent Disk Size' set to '10 GB'. To its right is a dropdown for 'TKGI API VM Type' set to 'large.disk (cpu: 2, ram: 8 GB, disk: 64 GB)'. A 'MORE INFO...' link is located at the top right of the page.

7. Under **Ephemeral Storage**, select one or more datastores for use as ephemeral storage, or use the search field on the right to find datastores by name.

Ephemeral Storage ⓘ

The screenshot shows the 'Ephemeral Storage' selection interface. It has a header 'Choose ephemeral storage resources for your availability zones' with a search icon. Below is a table with columns: Name, Capacity, Provisioned, and Free. Datastore 2 and Datastore 4 are selected (indicated by checked checkboxes). Datastore 3 is unselected. Datastore 1 and Datastore 5 have warning icons next to their free space values. A note at the bottom says 'Selected Ephemeral Storage: Datastore 4, Datastore 2'.

Name	Capacity	Provisioned	Free
<input checked="" type="checkbox"/> Datastore 2	1953.13 TB	1904.30 TB	97.66 TB
<input checked="" type="checkbox"/> Datastore 4	97.66 TB	106.97 TB	9.77 TB
<input type="checkbox"/> Datastore 3	39.06 TB	39.02 TB	1.95 TB
<input type="checkbox"/> Datastore 1	1.93 TB	2.03 TB	100.00 GB ⚠
<input type="checkbox"/> Datastore 5	1.93 TB	2.03 TB	100.00 GB ⚠

Selected Ephemeral Storage: Datastore 4, Datastore 2

8. Optionally activate **Specify Permanent Storage** to designate different datastores for ephemeral and permanent data.
9. If you activated permanent storage, under **Permanent Storage**, select one or more datastores for permanent storage, or use the search field to find datastores by name.

Specify Permanent Storage (default will use ephemeral storage for permanent storage)

Permanent Storage ①

Choose permanent storage resources for your availability zones

Name	Capacity	Provisioned	Free
<input checked="" type="checkbox"/> Datastore 2	1953.13 TB	1904.30 TB	97.66 TB
<input type="checkbox"/> Datastore 4	97.66 TB	106.97 TB	9.77 TB
<input checked="" type="checkbox"/> Datastore 3	39.06 TB	39.02 TB	1.95 TB
<input type="checkbox"/> Datastore 1	1.93 TB	2.03 TB	100.00 GB ⚠
<input type="checkbox"/> Datastore 5	1.93 TB	2.03 TB	100.00 GB ⚠

Selected Permanent Storage: Datastore 2, Datastore 3

Kubernetes Persistent Volume Storage ②

Choose persistent volume storage resource for use with Kubernetes clusters

Name	Capacity	Provisioned	Free
<input type="radio"/> Datastore 2	1953.13 TB	1904.30 TB	97.66 TB
<input type="radio"/> Datastore 4	97.66 TB	106.97 TB	9.77 TB
<input type="radio"/> Datastore 3	39.06 TB	39.02 TB	1.95 TB
<input checked="" type="radio"/> Datastore 1	1.93 TB	2.03 TB	100.00 GB
<input type="radio"/> Datastore 5	1.93 TB	2.03 TB	100.00 GB

Selected Datastore: Datastore 1

NEXT

- Click **Next** to configure plans.

Step 6: Configure Plans

A plan is a cluster configuration template that defines the set of resources for Tanzu Kubernetes Grid Integrated Edition to use when deploying Kubernetes clusters. A plan allows you to configure the numbers of control plane and worker nodes, select between Linux and Windows OS for worker nodes, specify the configuration of the control plane and worker VMs, set disk sizes, select availability zones for control plane and node VMs, and configure advanced settings.

Notes about Windows Worker Nodes

- You can use Windows worker nodes if you implement either vSphere with NSX-T or vSphere without NSX-T Data Center networking.
- You can create a maximum of 3 plans that implement Windows worker nodes in a given Tanzu Kubernetes Grid Integrated Edition deployment.
- If you use Windows worker nodes, certain options are not available, and the default values of other options change. See the option descriptions below for more information.

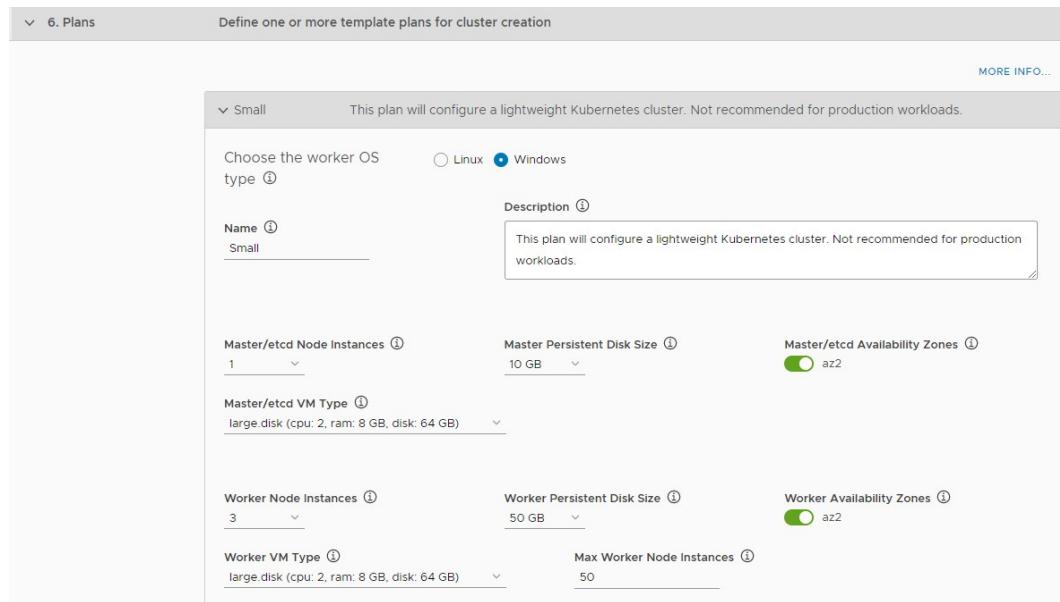
- If you use Windows worker nodes, by default one Linux worker node is deployed per Windows cluster. The Linux node provides cluster services to the Windows worker nodes. You can optionally make the cluster services Linux node highly available, in which case two Linux nodes are deployed.
- If you use Windows worker nodes, after you deploy Tanzu Kubernetes Grid Integrated Edition, you must use Operations Manager to manually install a Windows Server Stemcell in BOSH. For information about how to install a Windows Server Stemcell and other steps to perform after you deploy Tanzu Kubernetes Grid Integrated Edition with Windows worker nodes, see [Enable Plans with Windows Worker Nodes](#).

Tanzu Kubernetes Grid Integrated Edition Management Console provides preconfigured default plans, for different sizes of Kubernetes clusters. You can change the default configurations, or you can activate the plans as they are. You must activate at least one plan configuration because when you use the TKGI CLI to create a Kubernetes cluster, you must specify the plan on which you are basing the Kubernetes cluster. If no plans are activated, you cannot create Kubernetes clusters.

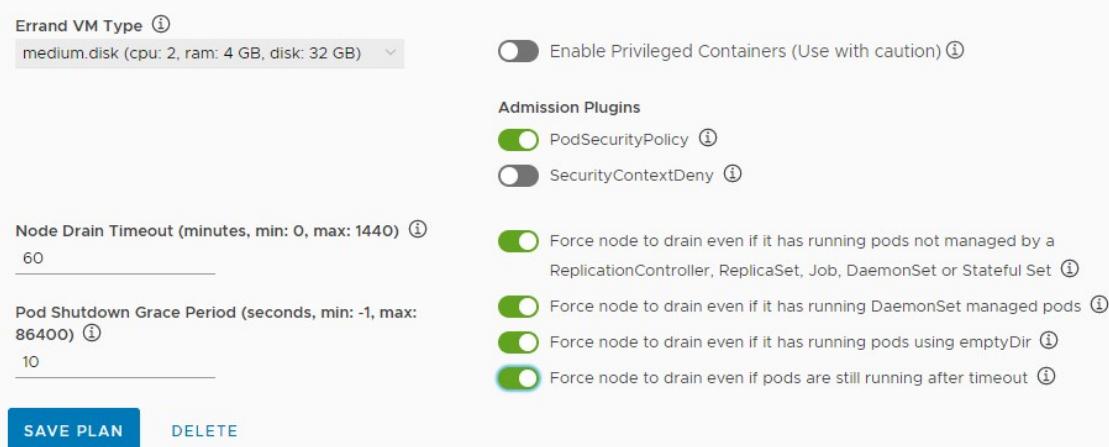
Tanzu Kubernetes Grid Integrated Edition plans support privileged containers and three admission control plugins. For information about privileged containers and the supported admission plugins, see [Privileged mode for pods](#) in the Kubernetes documentation. For information about admission plugins, see [Using Admission Control Plugins for Tanzu Kubernetes Grid Integrated Edition Clusters](#).

After you have deployed Tanzu Kubernetes Grid Integrated Edition, when you use the management console to create clusters, you can override some of the values that you define in plans by using [Compute Profiles](#).

1. To use the preconfigured plans as they are, click **Save Plan** for each of the **small**, **medium**, and **large** plans.
2. Optionally use the drop-down menus and buttons to change the default configurations of the preconfigured plans.
 - Select **Linux** or **Windows** to set the OS for the worker nodes.
 - Enter a name and a description for the plan in the **Name** and **Description** text boxes.
 - **Master/etc_d Node Instances:** Select **1** (small), **3** (medium), or **5** (large).
 - **Master Persistent Disk Size:** Select the size of the control plane persistent disk.
 - **Master/etc_d Availability Zones:** Activate one or more availability zones for the control plane nodes.
 - **Master/etc_d VM Type:** Select the size of the Control Plane VM. If you use Windows worker nodes, this option defaults to **large.disk**.
 - **Worker Node Instances:** Specify the number of worker nodes. For a small deployment, 3 is suggested.
 - **Worker Persistent Disk Size:** Select the size of the worker node persistent disk.
 - **Worker Availability Zones:** Activate one or more available availability zones for the worker nodes.
 - **Worker VM Type:** Select a configuration for worker nodes. If you use Windows worker nodes, this option defaults to **large.disk**.
 - **Max Worker Node Instances:** Select the maximum number of worker nodes.

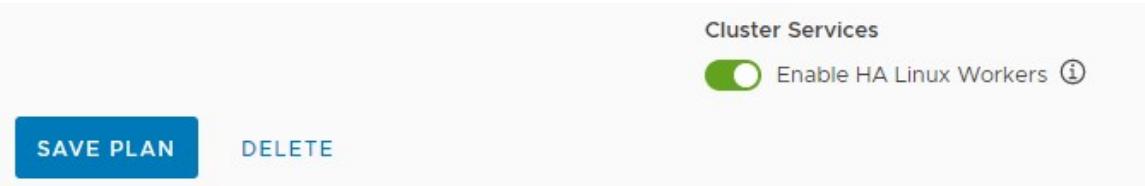


- ❖ **Errand VM Type:** Select the size of the VM to run BOSH errand tasks.
 - ❖ **Enable Privileged Containers:** Optionally activate privileged container mode. Use with caution. If you use Windows worker nodes, this option is not available.
 - ❖ **Admission Plugins:** Optionally activate admission plugins. Admission plugins, provide a higher level of access control to the Kubernetes API server and should be used with caution.
 - `PodSecurityPolicy`
 - `SecurityContextDeny`
-  **Note:** To use PodSecurityPolicy features, you must use Ops Manager v2.10.17 or later.
- ❖ **Node Drain Timeout:** Enter the timeout in minutes for the node to drain pods. If you set this value to 0, the node drain does not terminate. If you use Windows worker nodes, the node drain options are not available. To configure when the nodes drain, optionally activate the following:
 - **Force node to drain even if it has running pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or Stateful Set**
 - **Force node to drain even if it has running DaemonSet managed pods**
 - **Force node to drain even if it has running pods using emptyDir**
 - **Force node to drain even if pods are still running after timeout**
 - ❖ **Pod Shutdown Grace Period:** Enter a timeout in seconds for the node to wait before it forces the pod to terminate. If you set this value to -1, the default timeout is set to the one specified by the pod. If you use Windows worker nodes, this option is not available.



3. If you use Windows worker nodes, optionally activate the **Enable HA Linux Workers** option to deploy two Linux worker nodes per Windows cluster instead of one.

The Linux nodes provide cluster services to the Windows clusters.



4. Click **Save Plan** for each plan that you edit.
5. Optionally click **Add Plan** to create a new plan, configure it as described above, and click **Save Plan**.

You can create a maximum of 10 Linux plans and a maximum of 3 Windows plans.

6. Optionally delete any plans that you do not need.
7. Click **Next** to configure integrations.

Step 7: Configure Integrations

If your infrastructure includes existing deployments of VMware Tanzu Mission Control, Wavefront by VMware, VMware vRealize Operations Management Pack for Container Monitoring, or VMware vRealize Log Insight, you can configure TKGI to connect to those services. You can also configure TKGI to forward logs to a Syslog server.

Configure a Connection to VMware Tanzu Mission Control

Tanzu Mission Control integration lets you monitor and manage Tanzu Kubernetes Grid Integrated Edition clusters from the Tanzu Mission Control console, making the Tanzu Mission Control console a single point of control for all Kubernetes clusters.

For more information about Tanzu Mission Control, see the [VMware Tanzu Mission Control home page](#).

1. Select the **Enable** toggle to activate the Tanzu Mission Control Integration.
2. For **API URL**, enter the API URL of your Tanzu Mission Control subscription, without a trailing

slash (/).

3. For **Cluster Group Name**, enter the name of a Tanzu Mission Control cluster group.
 - ◊ The name can be `default` or another value, depending on your role and access policy:
 - `Org Member` users in VMware cloud services have a `service.admin` role in Tanzu Mission Control. These users:
 - By default, can only create and attach clusters in the `default` cluster group.
 - Can create new cluster groups after an `organization.admin` user grants them the `clustergroup.admin` or `clustergroup.edit` role.
 - VMware cloud services `Org Owner` users have `organization.admin` permissions in Tanzu Mission Control. These users:
 - Can create cluster groups.
 - Can grant `clustergroup` roles to `service.admin` users through the Tanzu Mission Control Access Policy view.
 - ◊ **Tanzu Mission Control Cluster Name Prefix:** Enter a name prefix for identifying the TKGI clusters in Tanzu Mission Control.
4. For **API token**, Enter your API token to authenticate with VMware Cloud Services APIs. Retrieve this token by logging into **VMware Cloud Services** and viewing your account information.
5. For **Cluster Name Prefix**, enter a name prefix for identifying the TKGI clusters in Tanzu Mission Control. This name prefix cannot contain uppercase letters. For more information, see the see [Cluster Group Name Limitation for Tanzu Mission Control Integration](#) in the Release Notes.

The screenshot shows the configuration interface for Tanzu Mission Control. At the top, there's a header with a dropdown menu and the status "Enabled". Below it is a "MORE INFO..." link. The main form has several input fields: "Enable" (checkbox checked), "API URL" (value: https://tanzumc.example.com), "Cluster Group Name" (value: default), "Cluster Name Prefix" (value: tkgi-), and "API Token" (value: my-token). A descriptive text on the right explains that when you integrate Tanzu Kubernetes Grid Integrated Edition with Tanzu Mission Control, any clusters that you deploy are visible for monitoring and management in the Tanzu Mission Control dashboard. At the bottom is a blue "SAVE" button.

6. Click **Save**.
7. Configure integrations with other applications, or click **Next** to install Harbor.

Configure a Connection to Wavefront

By connecting your Tanzu Kubernetes Grid Integrated Edition deployment to an existing deployment of Wavefront by VMware, you can obtain detailed metrics about Kubernetes clusters and pods. Before you configure Wavefront integration, you must have an active Wavefront account and access

to a Wavefront instance. For more information, including about how to generate a Wavefront access token, see [VMware TKGI Integration](#) and [VMware TKGI Integration Details](#) in the Wavefront by VMware documentation.

1. Select the **Enable** toggle to activate a connection to Wavefront.
2. Enter the address of your Wavefront instance in the **Wavefront URL** text box.
3. Enter the Wavefront API token in the **Wavefront Access Token** text box.
4. In the **HTTP Proxy for TKGI** text box, enter the address of the proxy server to use when it is not possible for the Tanzu Kubernetes Grid Integrated Edition Wavefront component to connect to an outside address over HTTP. For example, `http://your.proxy.com:8080` or `https://your.proxy.com:443`.
5. Click **Save**.
6. Configure integrations with other applications, or click **Next** to install Harbor.

Configure a Connection to VMware vRealize Operations Management Pack for Container Monitoring

You can connect your Tanzu Kubernetes Grid Integrated Edition deployment to an existing instance of VMware vRealize Operations Management Pack for Container Monitoring. vRealize Operations Management Pack for Container Monitoring provides detailed monitoring of your Kubernetes clusters. vRealize Operations Management Pack for Container Monitoring must be installed, licensed, running, and available in your environment before you activate the option. For more information, see the [vRealize Operations Management Pack for Container Monitoring](#) documentation.

If you activate the option to integrate Tanzu Kubernetes Grid Integrated Edition with VMware vRealize Operations Management Pack for Container Monitoring, the management console creates a **cAdvisor** container in your Tanzu Kubernetes Grid Integrated Edition deployment.

1. Select the **Enable** toggle to activate a connection to vRealize Operations Management Pack for Container Monitoring.
2. Click **Save**.
3. Configure integrations with other applications, or click **Next** to install Harbor.

Configure a Connection to VMware vRealize Log Insight

You can configure Tanzu Kubernetes Grid Integrated Edition deployment so that an existing deployment of VMware vRealize Log Insight pulls logs from all BOSH jobs and containers running in the cluster, including node logs from core Kubernetes and BOSH processes, Kubernetes event logs, and POD stdout and stderr.

vRealize Log Insight must be installed, licensed, running, and available in your environment before you activate the option. For instructions and additional information, see the [vRealize Log Insight documentation](#).

1. Select the **Enable** toggle to activate a connection to vRealize Log Insight.
2. Enter the address of your vRealize Log Insight instance in the **Host** text box.
3. Optionally deactivate **Enable SSL**.

4. Optionally deactivate **Disable SSL certificate validation**.

4. vRealize Log Insight Enabled

Enable Log Insight

Host ⓘ vrli.example.com

Enable SSL ⓘ

Disable SSL Certificate Validation ⓘ

SAVE

MORE INFO...

vRealize Log Insight delivers heterogeneous and highly scalable log management with intuitive, actionable dashboards, sophisticated analytics and broad third-party extensibility.

5. Click **Save**.
6. Configure integrations with other applications, or click **Next** to install Harbor.



Note: If you activate integration with vRealize Log Insight, Tanzu Kubernetes Grid Integrated Edition Management Console generates a unique vRealize Log Insight agent ID for the management console. You must provide this agent ID to vRealize Log Insight so that it can pull the appropriate logs from the management console VM. For information about how to obtain the agent ID, see [Obtain the VMware vRealize Log Insight Agent ID for TKGI Management Console](#) in *Troubleshooting Tanzu Kubernetes Grid Integrated Edition Management Console*.

Configure a Connection to Syslog

You can configure your Tanzu Kubernetes Grid Integrated Edition deployment so that it sends logs for BOSH-deployed VMs, Kubernetes clusters, and namespaces to an existing Syslog server.

1. Select the **Enable** toggle to activate a connection to Syslog.
2. Enter the address of your Syslog server in the **Address and port** text boxes.
3. Select **TCP**, **UDP**, or **RELP** from the **Transport protocol** drop-down menu.
4. Optionally select **Enable TLS**.
5. Enter a permitted peer ID.

5. Other Logging Enabled

Enable Syslog for TKGI ⓘ

Address and Port ⓘ
syslog.example : 514

Transport Protocol ⓘ
UDP

Enable TLS ⓘ

Permitted Peer ⓘ
*example.com

SAVE

MORE INFO...

6. Click **Save**.

7. Click **Next** to install Harbor.

Step 8: Configure Harbor

Harbor is an enterprise-class registry server that you can use to store and distribute container images. Harbor allows you to organize image repositories in projects, and to set up role-based access control to those projects to define which users can access which repositories. Harbor also provides rule-based replication of images between registries, optionally implements Content Trust with Notary and vulnerability scanning of stored images with Clair, and provides detailed logging for project and user auditing.



Note: In-product support for the [Clair](#) image scanner is deprecated with Harbor tile v2.2.1. To use Clair you can install it separately from the Harbor tile VM. For more information, see [Breaking Changes](#) in the Release Notes.

Harbor provides a Notary server that allows you to implement Content Trust by signing and verifying the images in the registry. When Notary content trust is activated, users can only push and pull images that have been signed and verified to or from the registry.

Harbor uses Clair to perform vulnerability and security scanning of images in the registry. You can set thresholds that prevent users from running images that exceed those vulnerability thresholds. Once an image is uploaded into the registry, Harbor uses Clair to check the various layers of the image against known vulnerability databases and reports any issues found.

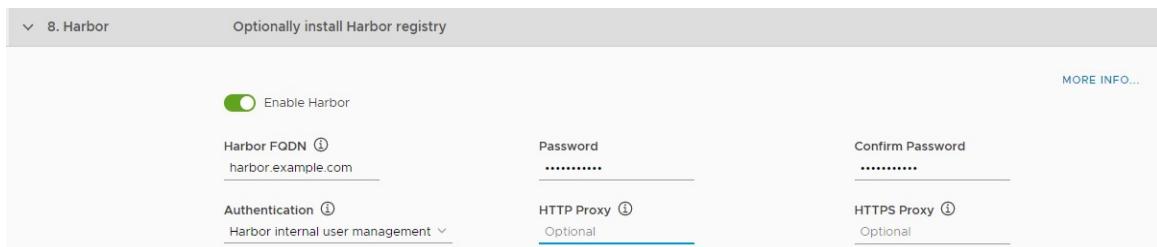
1. Optionally select the **Enable** toggle to deploy Harbor when you deploy Tanzu Kubernetes Grid Integrated Edition.
2. In the **Harbor FQDN** text box, enter a name for the Harbor VM, for example harbor.tkgi.example.com.

This is the address at which you access the Harbor administration UI and registry service. Before you set the host name, you must check for potential host name conflicts between TKGI and Harbor. - If the host name might resolve to an IP address that is not one that you want it to, clear the DNS entry manually to avoid conflicts in subsequent use. - If the host name can be resolved to an IP address that you have intentionally created beforehand, be aware that the IP address in the DNS entry might not be the same as the reachable IP address that TKGI Management Console uses, resulting in network issues. If you must use a pre-created DNS entry, after the TKGI deployment finishes, check the IP address that TKGI Management Console uses for Harbor and update the DNS entry accordingly.

3. Enter and confirm a password for the Harbor VM.
4. Select the method to use for authenticating connections to Harbor.
 - ◊ **Harbor internal user management:** Create a local database of users in the Harbor VM.
 - ◊ **Log in Harbor with LDAP users:** Use AD or LDAP to manage users. You configure the connection to the LDAP server in Harbor after deployment.
 - ◊ **UAA in Pivotal Container Service:** Use the same UAA as you use for Tanzu Kubernetes Grid Integrated Edition.

5. If your environment does not allow Harbor components to access the external network on which Tanzu Kubernetes Grid Integrated Edition Management Console is running, provide proxy addresses.
 - ◊ In the **HTTP Proxy** field, enter the proxy server to use when it is not possible for Harbor to connect to an outside address over HTTP. For example, `http://your.proxy.com:8080` or `https://your.proxy.com:443`.
 - ◊ In the **HTTPS Proxy** field, enter the proxy server to use when it is not possible for Harbor to connect to an outside address over HTTPS. For example, `http://your.proxy.com:8080` or `https://your.proxy.com:443`.

These proxies allow Clair to obtain updates from its vulnerability database.



The screenshot shows the configuration page for Harbor. At the top, there is a link to "Optional install Harbor registry". Below that, there is a section titled "Enable Harbor" with a green toggle switch. Under "Harbor FQDN", the value "harbor.example.com" is listed. There are fields for "Password" and "Confirm Password", both containing masked text. Under "Authentication", it says "Harbor internal user management". At the bottom, there are optional fields for "HTTP Proxy" and "HTTPS Proxy", both also containing masked text. A "MORE INFO..." link is located in the top right corner.

[View a larger version of this image.](#)

6. Optionally select **Manage Certificates Manually** for Harbor to use custom certificates with Harbor.

To use custom certificates with Harbor:

1. Paste the contents of the server certificate PEM file in the **SSL Certificate PEM** text box:

```
-----BEGIN CERTIFICATE-----
ssl_certificate_contents
-----END CERTIFICATE-----
```

2. Paste the contents of the certificate key in the **SSL Key PEM** text box:

```
-----BEGIN PRIVATE KEY-----
ssl_private_key_contents
-----END PRIVATE KEY-----
```

3. Paste the contents of the Certificate Authority (CA) file in the **Certificate Authority** text box:

```
-----BEGIN CERTIFICATE-----
CA_certificate_contents
-----END CERTIFICATE-----
```

4. Apply the configuration to update the TKGI Management Console database with the revised Harbor certificates.



Note: If you use the TKGI Management Console and Harbor and rotate Harbor certificates within the Harbor tile, you must activate the **Manage Certificates Manually For Harbor** option and configure the new Harbor certificates.

7. Select the location in which to store image repositories.
 - ◊ **Local file system:** Stores images in the Harbor VM. No configuration required.
 - ◊ **Remote NFS server:** Provide the IP address and path to an NFS share point.

Container Registry Storage Configuration ⓘ

Specify the File Storage Used for Storing Container Images

Remote NFS Server ▾

NFS Server Address
10.192.188.55://path/to/s

- ◊ **AWS S3:** Provide the connection details for your Amazon S3 account.
 - **Access Key:** Enter your access key ID.
 - **Secret Key:** Enter the secret access key for your access key ID.
 - **Region:** The region in which your bucket is located.
 - **Bucket Name:** Enter the name of your S3 bucket.
 - **Root Directory in the Bucket:** Enter the root directory of the bucket.
 - **Chunk Size:** The default is 5242880 but you can change it if necessary.
 - **Endpoint URL of your S3-compatible file store:** Enter the URL of your S3-compatible filestore.
 - **Enable v4auth:** Access to the S3 bucket is authenticated by default. Deselect this checkbox for anonymous access.
 - **Secure mode:** Access to your S3 bucket is secure by default. Deselect this checkbox to deactivate secure mode.

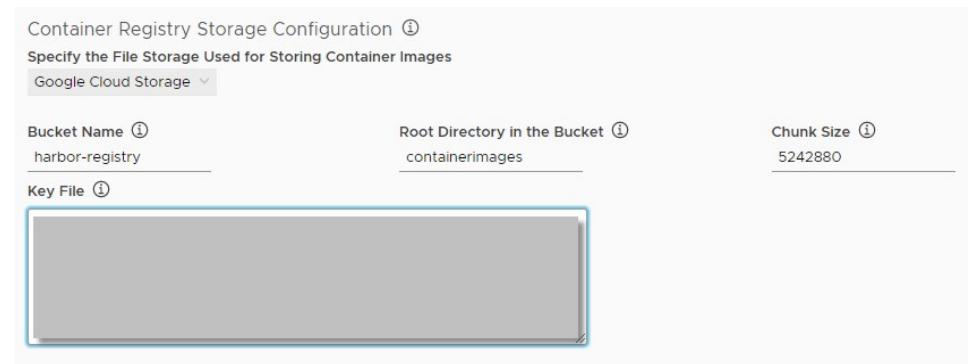
Container Registry Storage Configuration ⓘ

Specify the File Storage Used for Storing Container Images

AWS S3 ▾

Access Key ⓘ my-access-key	Secret Key ⓘ	Region ⓘ us-west-1
Bucket Name ⓘ harbor-registry	Root Directory in the Bucket ⓘ container-images	Chunk Size ⓘ 5242880
Endpoint URL of Your S3-compatible File Store ⓘ Endpoint URL		

- ◊ **Google Cloud Storage:** Provide the connection details for your Google Cloud Storage account.
 - **Bucket Name:** Enter the name of the GCS bucket.
 - **Root Directory in the Bucket:** Enter the root directory of the bucket.
 - **Chunk Size:** The default is 5242880 but you can change it if necessary.
 - **Key File:** Enter the service account key for your bucket.



8. Select the configuration for the Harbor VM from the **VM Type for Harbor-App** drop-down menu.
9. Select the size of the disk for the Harbor VM from the **Disk Size for Harbor-App** drop-down menu.

VM Type for Harbor-App	Disk Size for Harbor-App
large.disk (cpu: 2, ram: 8 GB, disk: 64 GB)	100 GB

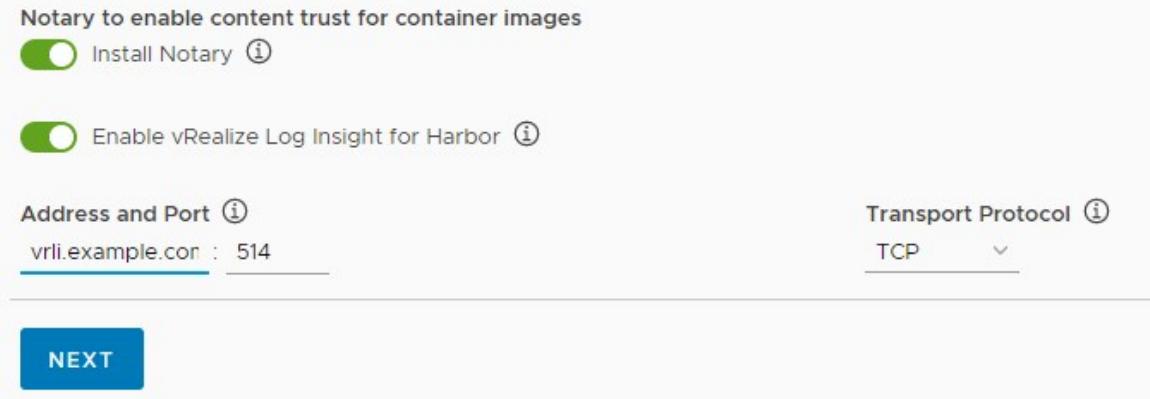
10. Optionally activate Clair by enabling the **Install Clair** toggle.
11. In the **Update Interval** field, specify when Clair will update its CVE databases for the registered sources.

When the updater interval expires, Clair will update its CVE databases. The default updater interval is 0, which means Clair will never update its CVE databases. If you set the updater interval to 24, Clair updates its CVE databases every 24 hours.

Clair Settings
<input checked="" type="checkbox"/> Install Clair
Update Interval
24

12. Optionally activate Notary by enabling the **Install Notary** toggle.
13. Optionally send Harbor logs to vRealize Log Insight by enabling the **Enable vRealize Log Insight for Harbor** toggle.

If you activate vRealize Log Insight, provide the address and port of your vRealize Log Insight service, and select either UDP or TCP for the transport protocol.



- Click **Next** to complete the configuration wizard.

Step 9: Configure CEIP

VMware's Customer Experience Improvement Program (CEIP) provides VMware with information to improve the products and services, fix problems, and advise you on how best to deploy and use our products. As part of the CEIP program, VMware collects technical information about your organization's use of Tanzu Kubernetes Grid Integrated Edition Management Console.

To configure VMware's Customer Experience Improvement Program (CEIP), do the following:

- Click **CEIP**.
- Review the information about the CEIP.

About the CEIP Program

VMware's Customer Experience Improvement Program ("CEIP") provides VMware with information that enables VMware to improve its products and services, to fix problems, and advise you on how best to deploy and use our products. As part of the CEIP, VMware collects technical information about your organization's use of VMware products and services on a regular basis in association with your organization's VMware license key(s). This information does not personally identify any individual.

Customers who participate in the CEIP receive proactive support benefits that include a weekly report based on telemetry data. Contact your Customer Success Manager to subscribe to this report. You can view a sample report at <https://via.vmware.com/TKGI>.

Additional information regarding the data collected through CEIP and the purposes for which it is used by VMware is set forth in the Trust & Assurance Center at <http://www.vmware.com/trustvmware/ceip.html>. If you prefer not to participate in VMware's CEIP for this product, you should uncheck the box below. You may join or leave VMware's CEIP for this product at any time.

Join the VMware Customer Experience Improvement Program

Please select how you will be using this TKGI Installation

- Demo or Proof-of-concept
- Development or Pre-production
- Production
- I do not wish to provide this information

NEXT

[View a larger version of this image.](#)

- If you wish to participate in CEIP, select the **Join the VMware Customer Experience Improvement Program** checkbox.
- Enter the following information in the fields:
 - Your entitlement account number or Tanzu customer number. If you are a VMware customer, you can find your entitlement account number in your **Account Summary** on my.vmware.com. If you are a Pivotal customer, you can find your Pivotal Customer Number in your Pivotal Order Confirmation email.
 - A descriptive name for your TKGI installation. The label you assign to this installation will be used in the reports to identify the environment.
- To provide information about the purpose for this installation, select an option.

Please select how you will be using this TKGI Installation

- Demo or Proof-of-concept
- Development or Pre-production
- Production
- I do not wish to provide this information

6. Click Save.



Note: If you join the CEIP Program for Tanzu Kubernetes Grid Integrated Edition, open your firewall to allow outgoing access to <https://vcsa.vmware.com/ph> on port 443.



Note: Even if you do not wish to participate in CIEP, Tanzu Kubernetes Grid Integrated Edition-provisioned clusters send usage data to the TKGI control plane. However, this data is not sent to VMware and remains on your Tanzu Kubernetes Grid Integrated Edition installation.

Step 10: Generate Configuration File and Deploy Tanzu Kubernetes Grid Integrated Edition

When all of the sections of the wizard are green, you can generate a YAML configuration file and deploy Tanzu Kubernetes Grid Integrated Edition.

1. Click **Generate Configuration** to see the generated YAML file.

The screenshot shows the management console interface. At the top, it displays the title "VMware Tanzu Kubernetes Grid Integrated Edition Management Console". On the right, there are user and theme settings. Below the header, a sidebar titled "TKGI Configuration" lists nine configuration items with their respective details. At the bottom of the page are two buttons: "GENERATE CONFIGURATION" and "EXIT".

Section	Details
1. vCenter Account	VC: vsphere.local, Username: admin@vsphere.local
2. Networking	NSX Manager: nsx.local, Username: admin@nsx.local
3. Identity	Identity Management Source: Local user database
4. Availability Zones	Configured Availability Zones: az
5. Resources & Storage	Ephemeral: Datastore 2; Permanent: Datastore 2; Kubernetes: Datastore 2
6. Plans	Small, Medium, Large
7. Integrations	Tanzu Mission Control, Wavefront, Syslog enabled
8. Harbor	Optionally install Harbor registry
9. CEIP & Telemetry	VMware's Customer Experience Improvement Program and Pivotal's Telemetry Program

- (Optional) Click **Export YAML** to save a copy of the YAML file for future use. This is recommended. The manifest is exported as the file `PksConfiguration.yaml`.
- (Optional) Specify an FQDN address for the Ops Manager VM by editing the YAML directly in the YAML editor.



WARNING: You cannot change the Ops Manager FQDN of Tanzu Kubernetes Grid Integrated Edition once it has already deployed.

To specify an FQDN address for the Ops Manager VM, update the YAML as follows:

1. Locate the `opsman_fqdn:` entry in the YAML file.
2. Update the `opsman_fqdn:` entry with the Ops Manager VM FQDN:

```
opsman_fqdn: "myopsman.example.com"
```

3. Make sure that the FQDN is mapped to the following IP address:
 - For vSphere with NSX-T deployments map it to the first address in the floating IP range.
 - For vSphere without NSX-T deployments, map it to the first address in the deployment network, excluding the gateway, deployment DNS, and reserved IP range.

If you start the deployment and you have not mapped the FQDN to an IP address, the deployment fails with an error. If this happens, configure the mapping as above, return to the YAML editor, and start the deployment again.

4. (Optional) To use a custom certificate for Ops Manager, edit the YAML directly in the YAML editor.

1. Generate a private key and root certificate for Ops Manager, by using `openssl`. For example:

```
openssl genrsa -out opsman.key 2048
```

```
openssl req -key opsman.key -new -x509 -days 365 -sha256 -extensions v3_ca -out opsman_ca.crt -subj "/C=US/ST=CA/L=Palo Alto/O=Vmware/OU=Eng/CN=Sign By Vmware.Inc"
```

2. Locate and update the `opsman_private_key:` entry in the YAML file.

```
opsman_private_key: -----BEGIN RSA PRIVATE KEY-----
MIIEpaIBAAKCAQ [...]
-----END RSA PRIVATE KEY-----
```

3. Locate and update the `opsman_root_cert:` entry in the YAML file.

```
opsman_root_cert: -----BEGIN CERTIFICATE-----
MIIDtTCCAp2 [...]
-----END CERTIFICATE-----
```

5. (Optional) Edit the YAML directly in the YAML editor to specify additional reserved IP ranges in the deployment network or service network.

No VMs will be deployed in the reserved ranges that you specify. To specify additional reserved IP ranges, update the YAML as follows:

1. Locate the `additional_dep_reserved_ip_range:` and `additional_svc_reserved_ip_range:` entries in the YAML file.
2. Update the `additional_dep_reserved_ip_range:` and `additional_svc_reserved_ip_range:` entries to specify reserved IP ranges in the

deployment and service networks:

- Deployment network:

```
additional_dep_reserved_ip_range: "172.16.100.2,172.16.100.3-172.1
6.100.10"
```

- Service network (vSphere without NSX-T only):

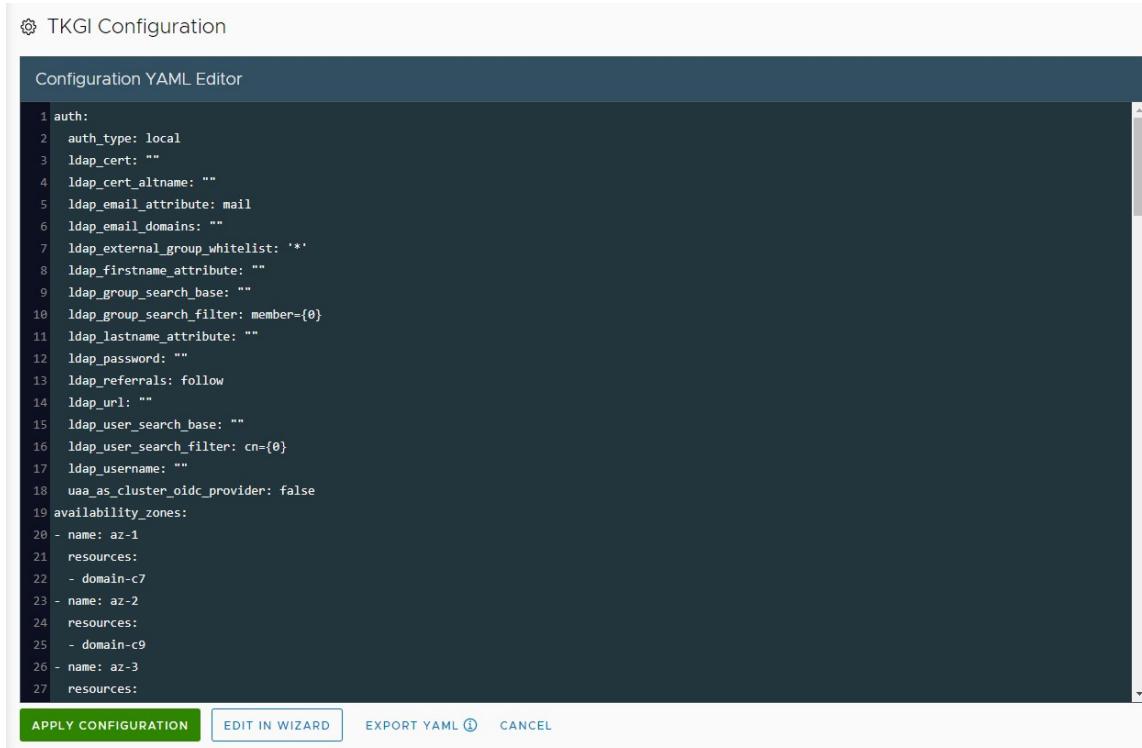
```
additional_svc_reserved_ip_range: ""
```

6. (Optional) Edit the YAML directly in the YAML editor to specify TKGI Operation Timeout. To specify the TKGI Operation Timeout, update the YAML as follows:

1. Locate the `nsx_feign_client_read_timeout`: entry in the YAML file.
2. Update the `nsx_feign_client_read_timeout` value, in milliseconds, to revise the TKGI-API operation timeout.

In large-scale NSX-T environments, increase the `nsx_feign_client_read_timeout` value to avoid timeouts during cluster deletion. To determine the optimal Operation Timeout setting, see [Cluster Deletion Fails in General Troubleshooting](#).

7. Click **Apply Configuration** then **Continue** to deploy Tanzu Kubernetes Grid Integrated Edition.



8. On the TKGI Configuration page, follow the progress of the deployment.
9. When the deployment has completed successfully, click **Continue** to monitor and manage your deployment.

The screenshot shows the VMware Tanzu Kubernetes Grid Integrated Edition Management Console. On the left, there's a sidebar with navigation links for TKG Integrated Edition, Quotas, Network Profiles, Administration (Identity Management, Configuration, Deployment Metadata), TKGI Configuration, TKGI Instance Upgrade, and TKGI Component Patch. The main content area is titled 'TKGI Configuration' and shows a table of deployment status. The table has columns for Operations, State, and Status. All entries show a green checkmark in the State column and 'Completed' in the Status column. The operations listed are NSX, Ops Manager, BOSH, PKS, and Harbor.

Operations	State	Status
NSX	✓ Success	Completed
Ops Manager	✓ Success	Completed
BOSH	✓ Success	Completed
PKS	✓ Success	Completed
Harbor	✓ Success	Completed

Next Steps

You can now access the Tanzu Kubernetes Grid Integrated Edition control plane and begin deploying Kubernetes clusters. For information about how to deploy clusters directly from the management console, see [Create and Manage Clusters in the Management Console](#).

For information about how you can use Tanzu Kubernetes Grid Integrated Edition Management Console to monitor and manage your Tanzu Kubernetes Grid Integrated Edition deployment, see [Monitor and Manage Tanzu Kubernetes Grid Integrated Edition in the Management Console](#).



Important: If you deployed Tanzu Kubernetes Grid Integrated Edition with plans that use Windows worker nodes, see [Enable Plans with Windows Worker Nodes](#) for information about how to install a Windows Server stemcell and other necessary configuration actions that you must perform. Plans that use Linux worker nodes are available immediately, but plans that use Windows worker nodes are ignored until you install the Windows Server stemcell.

If Tanzu Kubernetes Grid Integrated Edition fails to deploy, see [Troubleshooting](#).

Deploy Tanzu Kubernetes Grid Integrated Edition by Importing a YAML Configuration File

If you have an existing YAML configuration file from a previous deployment of VMware Tanzu Kubernetes Grid Integrated Edition, you can use the VMware Tanzu Kubernetes Grid Integrated Edition Management Console to deploy a new Tanzu Kubernetes Grid Integrated Edition instance from that file.

You can import a YAML from an earlier supported version of Tanzu Kubernetes Grid Integrated Edition Management Console. In this case, after you import the YAML, open the configuration in the wizard and configure any missing settings that are new in this version.

For information about how to deploy Tanzu Kubernetes Grid Integrated Edition by using the configuration wizard, see [Deploy Tanzu Kubernetes Grid Integrated Edition by Using the Configuration Wizard](#).



WARNING: Ideally, do not deploy TKGI from the management console to a datacenter that also includes TKGI instances that you deployed manually. If deploying management console and manual instances of TKGI to the same datacenter cannot be avoided, make sure that the TKGI instances that you deployed manually do not use the folder names `BoshVMFolder: pks_vms`, `BoshTemplateFolder: pks_templates`, `BoshDiskPath: pks_disk`. If a manual installation uses these folder names, the VMs that they contain will be deleted when you delete a TKGI instance from the management console.

If NSX-T Data Center uses custom certificates and you do not provide the CA certificate for NSX Manager, Tanzu Kubernetes Grid Integrated Edition Management Console automatically generates one and registers it with NSX Manager. This can cause other services that are integrated with NSX Manager not to function correctly. If you have manually deployed TKGI instances to the same datacenter as the one to which you are deploying this instance, you must manage NSX-T certificates manually by specifying the `nsx_ca_crt` option.

For information about how to upgrade an existing deployment, see [Upgrade Tanzu Kubernetes Grid Integrated Edition Management Console](#).

YAML Files and Passwords

When Tanzu Kubernetes Grid Integrated Edition Management Console generates the content of the YAML file for the YAML editor, it masks the passwords for NSX Manager, vCenter Server, and Harbor so that they do not appear in plain text. In the generated YAML files, the password fields look like the following example:

```
admin_password: <hidden:f065be51-84e9-4ca7-972d-ed46f7273123>
```

The `<hidden>` tag includes a GUID that refers to a database entry for the password that was entered into the configuration wizard. If you import a YAML file from an instance of Tanzu Kubernetes Grid Integrated Edition Management Console that is deployed in a different vSphere environment, the GUID provided in the hidden tag will not correspond to an entry in the database of the environment in which you are importing the YAML. As a consequence, if you import a YAML from a different vSphere environment, you must manually update the passwords for NSX Manager, vCenter Server, and Harbor in the YAML editor. If you are importing a YAML file from the same environment, the correct passwords are held in the database and no action is required.

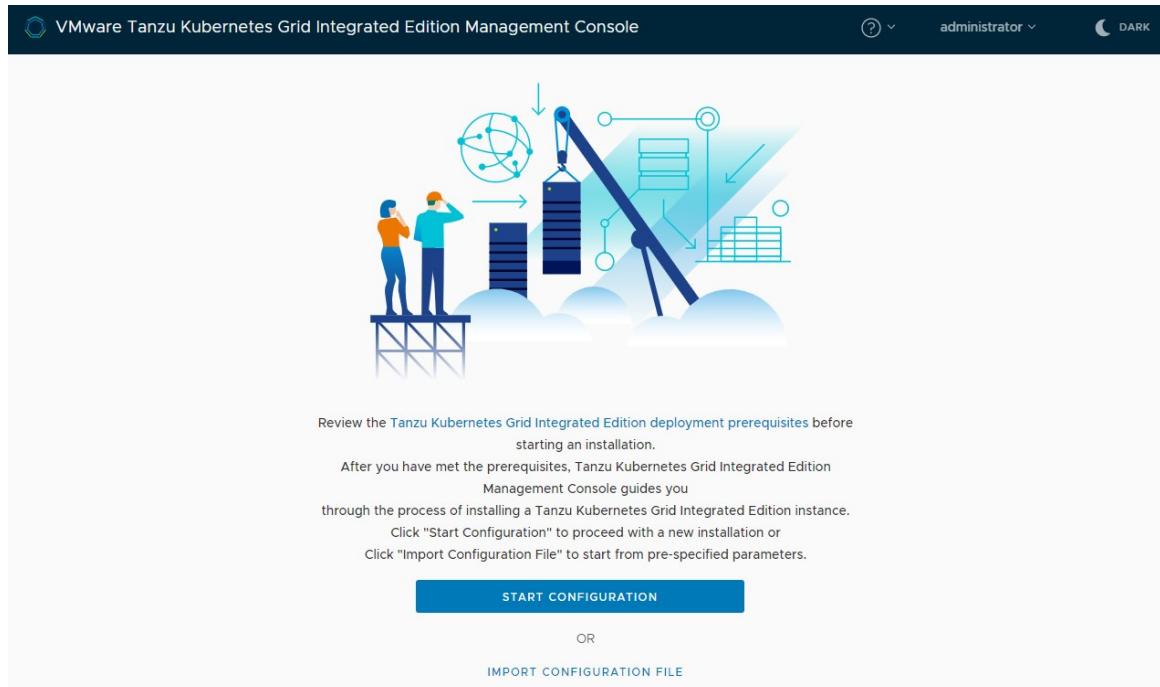
Prerequisites

- Deploy the Tanzu Kubernetes Grid Integrated Edition Management Console to vCenter Server.
- The vCenter Server instance must be correctly configured for Tanzu Kubernetes Grid Integrated Edition deployment. For information about the vCenter Server requirements, see [Virtual Infrastructure Prerequisites](#).

- Depending on the type of networking to use, your infrastructure must meet the appropriate prerequisites. For information about networking prerequisites, see the following topics:
 - Prerequisites for a Bring Your Own Topology Deployment to NSX-T Data Center
 - Prerequisites for an Automated NAT Deployment to NSX-T Data Center
 - Prerequisites for vSphere Without an NSX-T Network
- Log in to Tanzu Kubernetes Grid Integrated Edition Management Console.
- You have an existing YAML configuration file that you exported during a previous Tanzu Kubernetes Grid Integrated Edition deployment from Tanzu Kubernetes Grid Integrated Edition Management Console.
- For information about how to set the networking parameters in the YAML file, see [Networking Options in the YAML File](#) below.

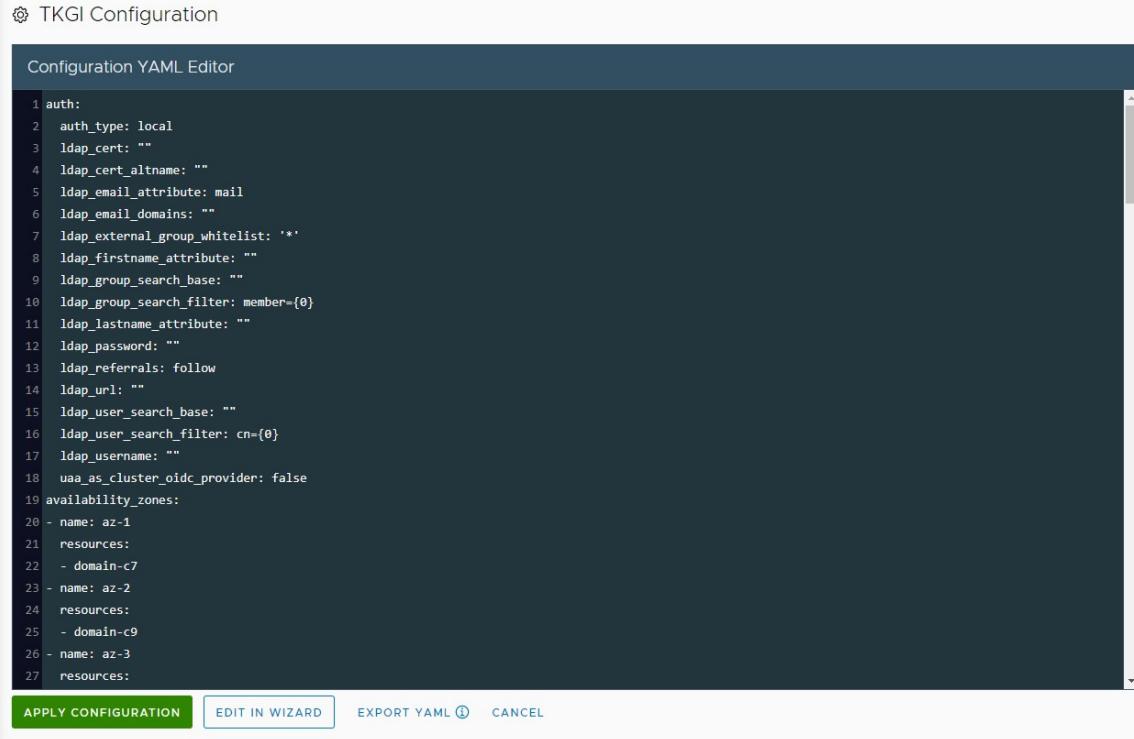
Import a YAML Configuration File

1. On the VMware Tanzu Kubernetes Grid Integrated Edition landing page, click **Install** then **Import Configuration File**.



[View a larger version of this image](#)

2. Drag the YAML file into the Import Configuration File window, or click **Browse** to navigate to it.
3. In the Configuration File editor, modify the contents of the YAML file appropriately for the new instance of Tanzu Kubernetes Grid Integrated Edition that you want to deploy.



[View a larger version of this image](#)

If the YAML was generated by an instance of the management console that is running in a different vSphere environment, update the passwords for NSX Manager, vCenter Server, and Harbor.

You can click the **Edit in Wizard** button, to open the imported configuration in the wizard and modify it there. For example, if you have imported a YAML that was generated by a previous version of Tanzu Kubernetes Grid Integrated Edition Management Console, open it in the wizard so that you can configure any options that are new in this version.

To abandon this YAML and start again, click **Import YAML** to upload the YAML again or to import a new one.

4. When you have finished editing the YAML in the Configuration File editor, click **Export YAML** to save a copy of your updated YAML configuration.
5. Click **Apply Configuration** and **Continue** to deploy Tanzu Kubernetes Grid Integrated Edition from this configuration file.
6. On the TKGI Configuration page, follow the progress of the deployment.
7. When the deployment has completed successfully, click **Continue** to monitor and manage your deployment.

The screenshot shows the VMware Tanzu Kubernetes Grid Integrated Edition Management Console. On the left, a sidebar menu includes 'TKG Integrated Edition', 'Quotas', 'Network Profiles', 'ADMINISTRATION' (with 'Identity Management' and 'Configuration' expanded), 'Deployment Metadata', 'TKGI Configuration', 'TKGI Instance Upgrade', and 'TKGI Component Patch'. The main content area is titled 'TKGI Configuration' and shows a 'DEPLOYMENT STATUS' card. It states 'Current configuration status of VMware Tanzu Kubernetes Grid Integrated Edition' and 'This installation was successfully deployed and all components are up and running.' Below this is a table with columns 'Operations', 'State', and 'Status' for components: NSX, Ops Manager, BOSH, PKS, and Harbor, all listed as 'Success' and 'Completed'. At the bottom is a 'CONTINUE' button.

[View a larger version of this image](#)

Networking Options in the YAML File

The networking parameters for the three types of Tanzu Kubernetes Grid Integrated Edition networking are all included in the `network:` section of the YAML file. When you edit the YAML file, you only need to set those parameters that apply to your type of networking.

The following table lists the parameters to set for each type of networking.

Unprepared NSX-T Data Center

Prepared NSX-T Data Center

vSphere without NSX-T

active_t0_edge_node	additional_dep_reserved_ip_range	additional_dep_reserved_ip_range
active_t0_edge_node_ip	autoprovision_nsx	additional_svc_reserved_ip_range
additional_dep_reserved_ip_range	dep_dns	dep_dns
autoprovision_nsx	dep_network_name	dep_network_cidr
dep_dns	dep_reserved_ip_range_from	dep_network_gateway
dep_network_cidr	dep_reserved_ip_range_to	dep_network_name
dep_reserved_ip_range_from	ntp_servers	dep_reserved_ip_range_from
dep_reserved_ip_range_to	nsx_ca_crt	dep_reserved_ip_range_to
external_portgroup_gateway	nsx_dns	flannel_pod_network_cidr
external_portgroup_netmask	nsx_fip_id	flannel_service_network_cidr
external_portgroup_subnet	nsx_host	ntp_servers
external_vlan_id	nsx_manual_ssl_certs	opsman_fqdn
floating_ips_range	nsx_nat_mode	svc_dns
nsx_manual_ssl_certs	nsx_node_ip_block_id	svc_network_name
nsx_ca_crt	nsx_password	svc_network_cidr
nsx_dns	nsx_pod_ip_block_id	svc_network_gateway
nsx_host	nsx_t0_id	svc_reserved_ip_range_from
nsx_node_cidr	nsx_username	svc_reserved_ip_range_to
nsx_password	nsx_verify_ssl_certs	use_antrea
nsx_pod_cidr	opsman_fqdn	use_nsx
nsx_username	use_nsx	
nsx_verify_ssl_certs		
ntp_servers		
opsman_fqdn		
standby_t0_edge_node		
standby_t0_edge_node_ip		
t0_edge_node_lb_ip		
t0_ha_mode_active_active		
use_nsx		

Next Steps

You can now access the Tanzu Kubernetes Grid Integrated Edition control plane and begin deploying Kubernetes clusters. For information about how to deploy clusters directly from the management console, see [Create and Manage Clusters in the Management Console](#).

For information about how you can use Tanzu Kubernetes Grid Integrated Edition Management Console to monitor and manage your Tanzu Kubernetes Grid Integrated Edition deployment, see [Monitor and Manage Tanzu Kubernetes Grid Integrated Edition in the Management Console](#).



Important: If you deployed Tanzu Kubernetes Grid Integrated Edition with plans that use Windows worker nodes, see [Enable Plans with Windows Worker Nodes](#) for information about how to install a Windows Server stemcell and other necessary configuration actions that you must perform. Plans that use Linux worker nodes are available immediately, but plans that use Windows worker nodes are ignored until you install the Windows Server stemcell.

If Tanzu Kubernetes Grid Integrated Edition fails to deploy, see [Troubleshooting](#).

Enable Plans with Windows Worker Nodes

If you used Tanzu Kubernetes Grid Integrated Edition Management Console to deploy Tanzu

Kubernetes Grid Integrated Edition on vSphere, and you created Plans that implement Windows worker nodes, you must use Operations Manager to provide BOSH with a vSphere stemcell for Windows Server. See the [release notes](#) for the correct version of vSphere stemcell for this release.

Tanzu Kubernetes Grid Integrated Edition Management Console does not provide a mechanism for the automatic upload and installation of the Windows stemcell. Because Operations Manager and BOSH Director for vSphere are deployed when you deploy Tanzu Kubernetes Grid Integrated Edition from Tanzu Kubernetes Grid Integrated Edition Management Console, you can only install the stemcell after you have deployed Tanzu Kubernetes Grid Integrated Edition.

After you deploy Tanzu Kubernetes Grid Integrated Edition from the management console, any plans that use Windows worker nodes are ignored until you install a Windows Stemcell and configure the management console to use it.

Prerequisites

- During Tanzu Kubernetes Grid Integrated Edition deployment, you configured a plan that implements Windows worker nodes. For information about creating a plan with Windows worker nodes, see [Configure Plans](#) in *Deploy Tanzu Kubernetes Grid Integrated Edition by Using the Configuration Wizard*.
- If you did not create a plan that uses Windows worker nodes when you deployed Tanzu Kubernetes Grid Integrated Edition, or if you have upgraded Tanzu Kubernetes Grid Integrated Edition Management Console from a version that did not support Windows worker nodes, you can use Tanzu Kubernetes Grid Integrated Edition Management Console to reconfigure the plans of your existing deployment. For information about reconfiguring Tanzu Kubernetes Grid Integrated Edition in the management console, see [Reconfigure Your Tanzu Kubernetes Grid Integrated Edition Deployment](#).

Step 1: Create a Windows Server Stemcell

vSphere stemcells for Windows Server version 2019 are not available on the [Tanzu Network](#). You must create Windows Server stemcells for vSphere by using Stembuild and your own Windows Server ISO.

Create a vSphere stemcell for Windows Server version 2019 by following the instructions in [Creating a Windows Stemcell for vSphere Using Stembuild](#).

Step 2: Install the Windows Server Stemcell in Operations Manager

After you have created your stemcell, you must upload it to Operations Manager and install it in BOSH. You can use either the Operations Manager interface or the Operations Manager CLI. These instructions describe how to upload and install the stemcell by using the Operations Manager interface.

1. Log in to Operations Manager. For information about how to log in to Operations Manager and the credentials to use, see [Log In to the Operations Manager UI](#).
2. Select **Stemcell Library**.
3. Click **Import Stemcell** and navigate to the location on your local machine where you saved

the stemcell `.tgz` file in [Step 1](#) above.

4. Click **Save**.

Step 3: Reconfigure Your Tanzu Kubernetes Grid Integrated Edition Deployment to Use the Stemcell

After you have created a Windows stemcell and used Operations Manager to install it in BOSH Director for vSphere, you must redeploy your Tanzu Kubernetes Grid Integrated Edition with the Windows stemcell installed.

1. In Tanzu Kubernetes Grid Integrated Edition Management Console, expand **Configuration** and select **TKGI Configuration**.
2. If you did not already create a plan that uses Windows worker nodes, or if you have upgraded this Tanzu Kubernetes Grid Integrated Edition instance from a version that did not support Windows worker nodes, expand the **Plans** section and create or reconfigure a plan that uses Windows worker nodes.
3. If necessary, reconfigure any other options that show a red status bar or that you want to change.
4. Click **Generate Configuration** to view the generated YAML file.
5. Click **Apply Configuration** to redeploy this Tanzu Kubernetes Grid Integrated Edition instance.
6. When the deployment finishes, go to the **TKG Integrated Edition** view and verify that the Windows Stemcell Status is **INSTALLED**.

You can now deploy Kubernetes clusters in plans that implement Windows worker nodes.

Install Tanzu Kubernetes Grid Integrated Edition on vSphere with VMware NSX Using Ops Manager

This topic lists the procedures to follow to install Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T networking manually, using Ops Manager.



Note: The recommended method for installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T is to use the Tanzu Kubernetes Grid Integrated Edition Management Console. For information, see [Install on vSphere with the Management Console](#).

Step 1: Prepare to Install Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T

In preparation for installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T Data Center, review all of the topics in the subsection [Preparing to Install Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T Data Center](#).

Step 2: Install and Configure NSX-T Data Center for Tanzu Kubernetes Grid Integrated Edition

NSX-T Data Center must be installed and configured before you install Tanzu Kubernetes Grid Integrated Edition.

For instructions, see one of the following:

- [Installing and Configuring NSX-T Data Center v3.0 for Tanzu Kubernetes Grid Integrated Edition](#)
- [Installing and Configuring NSX-T Data Center v2.5 for Enterprise PKS in the TKGI v1.7 documentation](#).
 - Follow the sequence of NSX-T v2.5 installation topics in the v1.7 documentation, through [Provisioning a Load Balancer for the NSX-T Management Cluster](#). Then return here to the TKGI v1.9 docs and continue with the next step.

Step 3: Create the Management Plane for Tanzu Kubernetes Grid Integrated Edition

Prepare the vSphere and NSX-T infrastructure for the Tanzu Kubernetes Grid Integrated Edition Management Plane where Ops Manager, BOSH Director, Tanzu Kubernetes Grid Integrated Edition components, and Harbor Registry are deployed. This includes creating a vSphere resource pool for Tanzu Kubernetes Grid Integrated Edition management components, an NSX Tier-1 (T1) Logical Switch, an NSX Tier-1 Logical Router and Port, and NAT rules (if you are using NAT mode).

For instructions, see [Create Management Plane in *Installing and Configuring NSX-T Data Center v3.0 for TKGI*](#).

Step 4: Create the Compute Plane for Tanzu Kubernetes Grid Integrated Edition

Create vSphere Resource Pools for the Availability Zones where you will deploy Kubernetes clusters. These resource pools map to the AZs you will create when you configure BOSH Director and reference when you install the Tanzu Kubernetes Grid Integrated Edition tile.

Create IP blocks for the [node networks](#) and the [pod networks](#). Typically the initial subnets for both nodes and pods will have a size of 256 (/16).

Create a [Floating IP Pool](#) from which to assign routable IP addresses to components. This network provides your load balancing address space for each Kubernetes cluster created by Tanzu Kubernetes Grid Integrated Edition. The network also provides IP addresses for Kubernetes API access and Kubernetes exposed services.

For instructions, see [Create IP Blocks and Pool for Compute Plane in *Installing and Configuring NSX-T Data Center v3.0 for TKGI*](#).

Step 5: Deploy Ops Manager for Tanzu Kubernetes Grid Integrated Edition with NSX-T

Deploy a supported version of Ops Manager on the NSX-T Management Plane network.

For instructions, see [Deploying Ops Manager with NSX-T for Tanzu Kubernetes Grid Integrated Edition](#).

Step 6: Generate the NSX-T Management Cluster Root CA Certificate and Key

Generate the CA Cert for the NSX Manager and import the certificate to NSX Manager.

For instructions, see [Generate and Register the NSX-T Management SSL Certificate and Private Key](#).

Step 7: Configure BOSH Director for vSphere with NSX-T

Create BOSH availability zones (AZs) that map to the Management and Compute resource pools in vSphere, and the Management and Control plane networks in NSX-T.

For instructions, see [Configuring BOSH Director with NSX-T for Tanzu Kubernetes Grid Integrated Edition](#).

Step 8: Generate and Register the NSX-T Management Cluster Super User Principal Identity Certificate and Key

Generate the NSX Manager Super User Principal Identity Certificate and register it with the NSX Manager using the NSX API.

For instructions, see [Generating and Registering the NSX Manager Superuser Principal Identity Certificate and Key](#).

Step 9: Install Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T

At this point your NSX-T environment is prepared for Tanzu Kubernetes Grid Integrated Edition installation using the Tanzu Kubernetes Grid Integrated Edition tile in Ops Manager.

For instructions, see [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T](#).

Step 10: Install VMware Harbor Registry for Tanzu Kubernetes Grid Integrated Edition

The VMware Harbor Registry is recommended for Tanzu Kubernetes Grid Integrated Edition. Install Harbor in the NSX Management Plane with other Tanzu Kubernetes Grid Integrated Edition components, such as the TKGI API and TKGI database, Ops Manager, and BOSH.

If you are using the [NAT deployment topology](#), create a DNAT rule that maps the private Harbor IP address to a routable IP address from the floating IP pool on the TKGI management network. See [Create DNAT Rule](#).

For instructions, see [Installing VMware Harbor Registry](#).

Step 11: Install the TKGI and Kubectl CLIs

See [Installing the TKGI CLI](#) and [Installing the Kubernetes CLI](#).

Step 12: Create Admin Users for Tanzu Kubernetes Grid Integrated Edition

See [Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on vSphere](#)

Step 13: Verify the Installation of Tanzu Kubernetes Grid Integrated Edition

Create a Kubernetes cluster using the TKGI CLI. For instructions, see [Create a Kubernetes Cluster](#).

Deploy a simple workload to the Kubernetes cluster. For instructions, see [Deploy Workloads on vSphere with NSX-T](#).

Step 14: Perform Desired Post-Installation Configurations

After you have installed Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T, refer to the following subsection for topics describing additional NSX-T configuration options: [Advanced Configurations for Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T Data Center](#)

Step 15: Create Network Profiles to Customize Cluster Deployments

Network profiles let you provide customized deployment templates for Kubernetes clusters. See [Network Profiles \(NSX-T Only\)](#) for details.

Preparing to Install Tanzu Kubernetes Grid Integrated Edition on vSphere with VMware NSX

This topic lists the sections to follow when installing VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) on vSphere with NSX-T Data Center.

Preparing to Install Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T

In preparation for installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T, review the following documentation:

- [Version Requirements for Tanzu Kubernetes Grid Integrated Edition](#)
- [Hardware Requirements for Tanzu Kubernetes Grid Integrated Edition](#)
- [VMware Ports and Protocols on the VMware site](#)
- [NSX-T Deployment Topologies for Tanzu Kubernetes Grid Integrated Edition](#)
- [NSX-T Cluster Objects Created for Tanzu Kubernetes Grid Integrated Edition](#)
- [Network Planning Tanzu Kubernetes Grid Integrated Edition](#)

- Considerations for Using the NSX-T Policy API with TKGI

vSphere with NSX-T Version Requirements

This topic describes the version requirements for installing VMware Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T integration.

For prerequisites and resource requirements for installing Tanzu Kubernetes Grid Integrated Edition on vSphere without NSX-T integration, see [vSphere Prerequisites and Resource Requirements](#).

For hardware and resource requirements for deploying Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T in production environments, see [Hardware Requirements for Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T](#).

Tanzu Kubernetes Grid Integrated Edition supports air-gapped deployments on vSphere with or without NSX-T integration.

You can also configure integration with the Harbor tile, an enterprise-class registry server for container images. For more information, see [VMware Harbor Registry](#) in the *VMware Partner documentation*.

vSphere Version Requirements

For Tanzu Kubernetes Grid Integrated Edition on vSphere version requirements, refer to the [VMware Product Interoperability Matrices](#).

NSX-T Integration Component Version Requirements

Refer to the [Tanzu Kubernetes Grid Integrated Edition Release Notes](#) for supported NSX-T versions.

Hardware Requirements for Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T

This topic provides hardware requirements for production deployments of VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) on vSphere with NSX-T.

vSphere Clusters for Tanzu Kubernetes Grid Integrated Edition

A vSphere cluster is a collection of ESXi hosts and associated virtual machines (VMs) with shared resources and a shared management interface. Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T requires the following vSphere clusters:

- [Tanzu Kubernetes Grid Integrated Edition Management Cluster](#)
- [Tanzu Kubernetes Grid Integrated Edition Edge Cluster](#)
- [Tanzu Kubernetes Grid Integrated Edition Compute Cluster](#)

For more information on creating vSphere clusters, see [Creating Clusters](#) in the vSphere documentation.

Tanzu Kubernetes Grid Integrated Edition Management Cluster

The Tanzu Kubernetes Grid Integrated Edition Management Cluster on vSphere comprises the following components:

- vCenter Server
- NSX-T Manager v3.0 or later (quantity 3)

For more information, see [Installing and Configuring NSX-T Data Center v3.0 for TKGI](#).

Tanzu Kubernetes Grid Integrated Edition Edge Cluster

A TKGI Edge Cluster on vSphere with NSX-T is comprised of two or more NSX-T Edge Nodes. The maximum supported number of Edge Nodes per TKGI Edge Cluster is 10.

For more information, see [Installing and Configuring NSX-T Data Center v3.0 for TKGI](#).

Tanzu Kubernetes Grid Integrated Edition Compute Cluster

The Tanzu Kubernetes Grid Integrated Edition Compute Cluster on vSphere comprises the following components:

- Kubernetes control plane nodes (quantity 3)
- Kubernetes worker nodes

For more information, see [Installing and Configuring NSX-T Data Center v3.0 for TKGI](#).

Tanzu Kubernetes Grid Integrated Edition Management Plane Placement Considerations

The Tanzu Kubernetes Grid Integrated Edition Management Plane comprises the following components:

- Ops Manager
- BOSH Director
- TKGI Control Plane
- VMware Harbor Registry

Depending on your design choice, TKGI management components can be deployed in the Tanzu Kubernetes Grid Integrated Edition Management Cluster on the standard vSphere network or in the Tanzu Kubernetes Grid Integrated Edition Compute Cluster on the NSX-T-defined virtual network.

For more information, see [NSX-T Deployment Topologies for Tanzu Kubernetes Grid Integrated Edition](#).

Configuration Requirements for vSphere Clusters for Tanzu Kubernetes Grid Integrated Edition

For each vSphere cluster defined for Tanzu Kubernetes Grid Integrated Edition, the following configurations are required to support production workloads:

- The vSphere Distributed Resource Scheduler (DRS) is enabled. For more information, see [Creating a DRS Cluster](#) in the vSphere documentation.

- The DRS custom automation level is set to **Partially Automated** or **Fully Automated**. For more information, see [Set a Custom Automation Level for a Virtual Machine](#) in the vSphere documentation.
- vSphere high-availability (HA) is enabled. For more information, see [Creating and Using vSphere HA Clusters](#) in the vSphere documentation.
- vSphere HA Admission Control (AC) is configured to support one ESXi host failure. For more information, see [Configure Admission Control](#) in the vSphere documentation.

Specifically:

- Host failure: Restart VMs
- Admission Control: Host failures cluster tolerates = 1

RPD for Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T

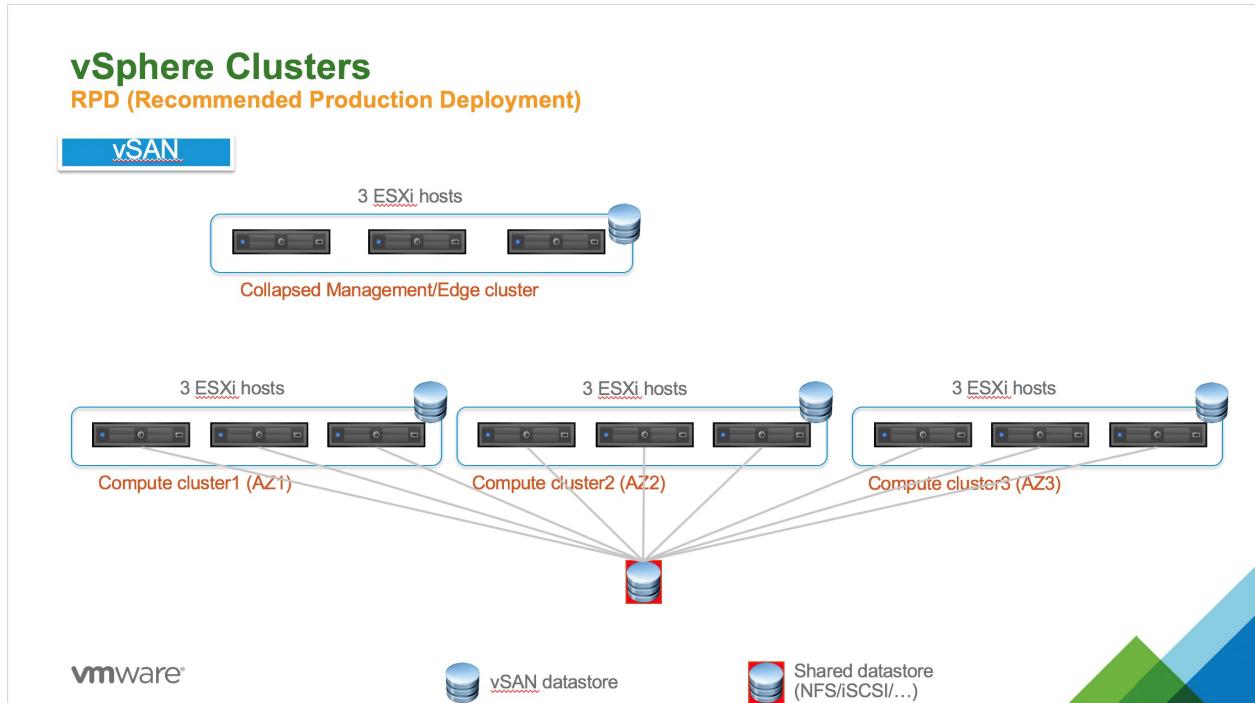
The recommended production deployment (RPD) topology represents the VMware-recommended configuration to run production workloads in Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T.



Note: The RPD differs depending on whether you are using vSAN or not.

RPD for Tanzu Kubernetes Grid Integrated Edition with vSAN

The RPD for Tanzu Kubernetes Grid Integrated Edition with vSAN storage requires 12 ESXi hosts. The diagram below shows the topology for this deployment.



The following subsections describe configuration details for the RPD with vSAN topology.

Management/Edge Cluster

The RPD with vSAN topology includes a Management/Edge Cluster with the following characteristics:

- Collapsed Management/Edge Cluster with three ESXi hosts.
- Each ESXi host runs one NSX-T Manager. The NSX-T Control Plane has three NSX-T Managers total.
- Two NSX-T Edge Nodes are deployed across two different ESXi hosts.

Compute Clusters

The RPD with vSAN topology includes three Compute Clusters with the following characteristics:

- Each Compute cluster has three ESXi hosts and is bound by a distinct availability zone (AZ) defined in BOSH Director.
 - Compute cluster1 (AZ1) with three ESXi hosts.
 - Compute cluster2 (AZ2) with three ESXi hosts.
 - Compute cluster3 (AZ3) with three ESXi hosts.
- Each Compute cluster runs one instance of an Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes cluster with three control plane nodes per cluster and a per-plan number of worker nodes.

Storage (vSAN)

The RPD with vSAN topology requires the following storage configuration:

- Each Compute Cluster is backed by a vSAN datastore
- An external shared datastore (using NFS or iSCSI, for instance) must be provided to store Kubernetes Pod PV (Persistent Volumes).
- Three ESXi hosts are required per Compute cluster because of the vSAN cluster requirements. For data protection, vSAN creates two copies of the data and requires one witness.

For more information on using vSAN with Tanzu Kubernetes Grid Integrated Edition, see [PersistentVolume Storage Options on vSphere](#).

Future Growth

The RPD with vSAN topology can be scaled as follows to accommodate future growth requirements:

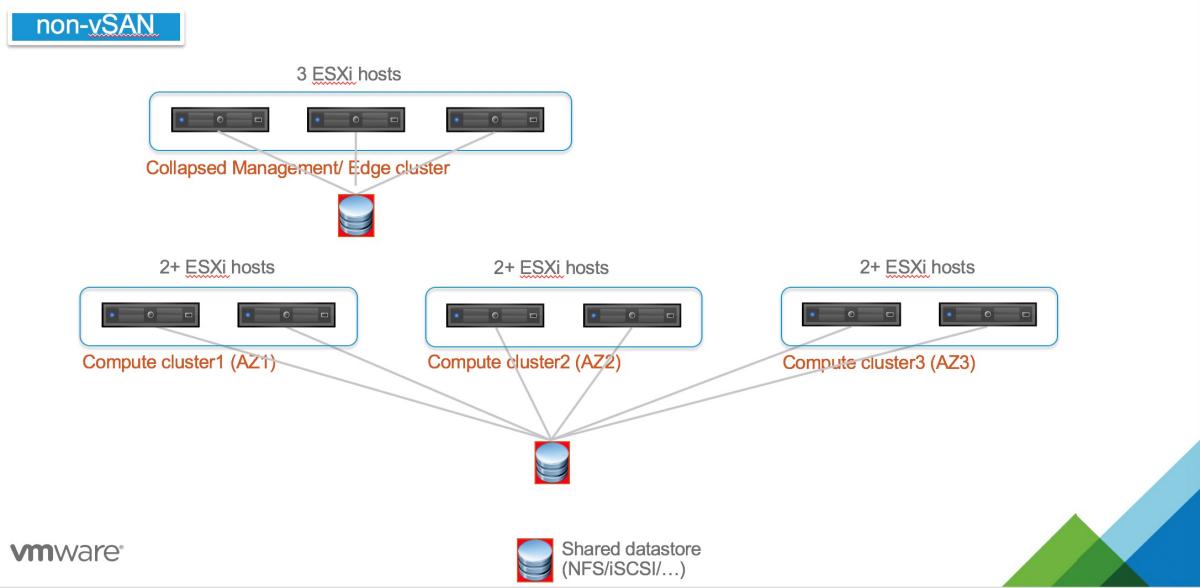
- The collapsed Management/Edge Cluster can be expanded to include up to 64 ESXi hosts.
- Each Compute Cluster can be expanded to include up to 64 ESXi hosts.

RPD for Tanzu Kubernetes Grid Integrated Edition without vSAN

The RPD for Tanzu Kubernetes Grid Integrated Edition without vSAN storage requires nine ESXi hosts. The diagram below shows the topology for this deployment.

vSphere Clusters

RPD (Recommended Production Deployment)



The following subsections describe configuration details for the RPD of Tanzu Kubernetes Grid Integrated Edition without vSAN.

Management/Edge Cluster

The RPD without vSAN includes a Management/Edge Cluster with the following characteristics:

- Collapsed Management/Edge Cluster with three ESXi hosts.
- Each ESXi host runs one NSX-T Manager. The NSX-T Control Plane has three NSX-T Managers total.
- Two NSX-T Edge Nodes are deployed across two different ESXi hosts.

Compute Clusters

The RPD without vSAN topology includes three Compute Clusters with the following characteristic:

- Each Compute cluster has two ESXi hosts and is bound by a distinct availability zone (AZ) defined in BOSH Director.
 - Compute cluster1 (AZ1) with two ESXi hosts.
 - Compute cluster2 (AZ2) with two ESXi hosts.
 - Compute cluster3 (AZ3) with two ESXi hosts.
- Each Compute cluster runs one instance of a Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes cluster with three control plane nodes per cluster and a per-plan number of worker nodes.

Storage (non-vSAN)

The RPD without vSAN topology requires the following storage configuration:

- All Compute Clusters are connected to same shared datastore that is used for persistent VM

disks for Tanzu Kubernetes Grid Integrated Edition components and Persistent Volumes (PVs) for Kubernetes pods.

- All datastores can be collapsed to single datastore, if needed.

Future Growth

The RPD without vSAN topology can be scaled as follows to accommodate future growth requirements:

- The collapsed Management/Edge Cluster can be expanded to include up to 64 ESXi hosts.
- Each Compute Cluster can be expanded to include up to 64 ESXi hosts.

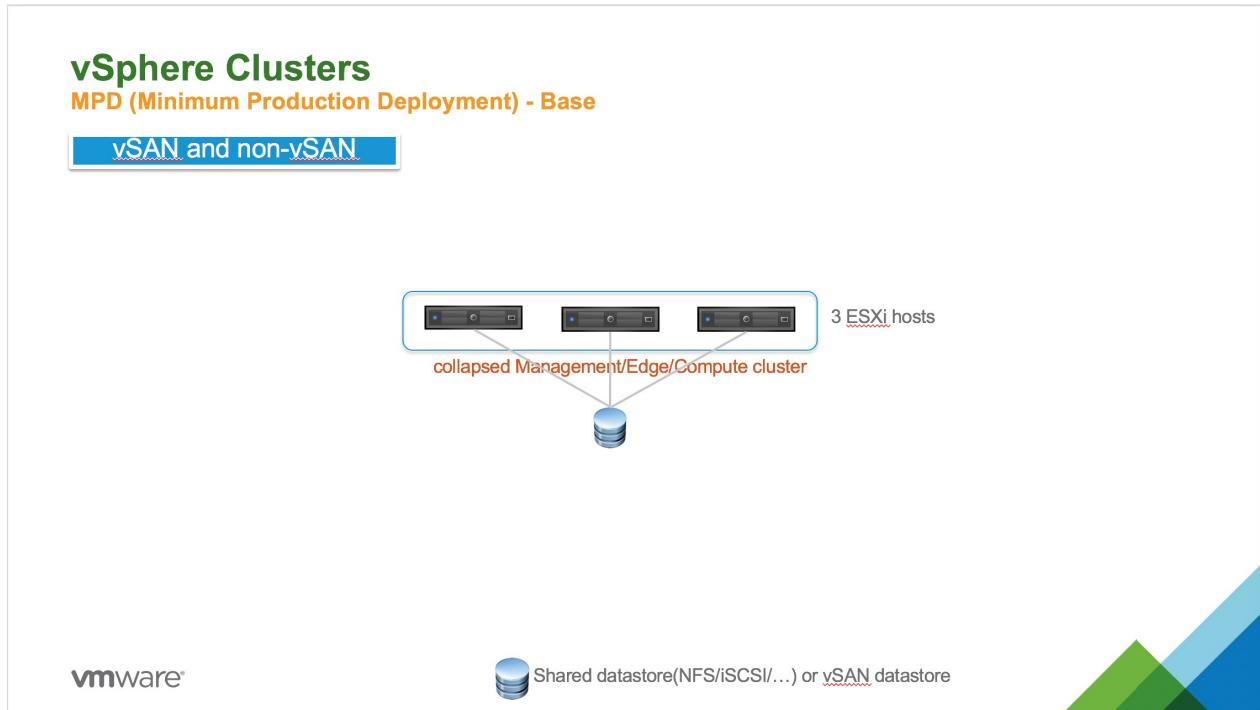
MPD for Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T

The minimum production deployment (MPD) topology represents the baseline requirements for running Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T.



Note: The MPD topology for Tanzu Kubernetes Grid Integrated Edition applies to both vSAN and non-vSAN environments.

The diagram below shows the topology for this deployment.



The following subsections describe configuration details for an MPD of Tanzu Kubernetes Grid Integrated Edition.

MPD Topology

The MPD topology for Tanzu Kubernetes Grid Integrated Edition requires the following minimum configuration:

- A single collapsed Management/Edge/Compute cluster running three ESXi hosts in total.
- Each ESXi host runs one NSX-T Manager. The NSX-T Control Plane has three NSX-T Managers in total.
- Each ESXi host runs one Kubernetes control plane node. Each Kubernetes cluster has three control plane nodes in total.
- Two NSX-T edge nodes are deployed across two different ESXi hosts.
- The shared datastore (NFS or iSCSI, for instance) or vSAN datastore is used for persistent VM disks for Tanzu Kubernetes Grid Integrated Edition components and Persistent Volumes (PVs) for Kubernetes pods.
- The collapsed Management/Edge/Compute cluster can be expanded to include up to 64 ESXi hosts.



Note: For an MPD deployment, each ESXi host must have four physical network interface controllers (PNICs). In addition, while a Tanzu Kubernetes Grid Integrated Edition deployment requires a minimum of three nodes, Tanzu Kubernetes Grid Integrated Edition upgrades require four ESXi hosts to ensure full survivability of the NSX-T Manager appliance.

MPD Configuration Requirements

When configuring vSphere for an MPD topology for Tanzu Kubernetes Grid Integrated Edition, keep in mind the following requirements:

- When deploying the NSX-T Manager to each ESXi host, create a vSphere distributed resource scheduler (DRS) anti-affinity rule of type “separate virtual machines” for each of the three NSX-T Managers.
- When deploying the NSX-T Edge Nodes across two different ESXi hosts, create a DRS anti-affinity rule of type “separate virtual machines” for both Edge Node VMs.
- After deploying the Kubernetes cluster, you must manually make sure each control plane node is deployed to a different ESXi host by tuning the DRS anti-affinity rule of type “separate virtual machines.”

For more information on defining DRS anti-affinity rules, see [Virtual Machine Storage DRS Rules](#) in the vSphere documentation.

MPD Considerations

When planning an MPD topology for Tanzu Kubernetes Grid Integrated Edition, keep in mind the following:

- Leverage vSphere resource pools to allocate proper hardware resources for the Tanzu Kubernetes Grid Integrated Edition Management Plane components and tune reservation and resource limits accordingly.
- There is no fault tolerance for the Kubernetes cluster because Tanzu Kubernetes Grid Integrated Edition Availability Zones are not fully leveraged with this topology.
- At a minimum, the Tanzu Kubernetes Grid Integrated Edition AZ should be mapped to a

vSphere Resource Pool.

For more information, see [Create Management Plane](#) and [Create IP Blocks and Pool for Compute Plane](#) in *Installing and Configuring NSX-T Data Center v3.0 for TKGI*.

VM Inventory and Sizes

The following tables list the VMs and their sizes for deployments of Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T.

Management Plane VMs and Sizes

The following table lists the resource requirements for NSX-T infrastructure and Tanzu Kubernetes Grid Integrated Edition Management Plane VMs.

VM	CPU Cores	Memory (GB)	Ephemeral Disk (GB)
BOSH Director	2	8	103
Harbor Registry	2	8	167
Ops Manager	1	8	160
TKGI API	2	8	64
TKGI Database	2	8	64
NSX-T Manager 1	6	24	200
NSX-T Manager 2	6	24	200
NSX-T Manager 3	6	24	200
vCenter Appliance	4	16	290
TOTAL	31	128	1.38 TB



Note: The NSX-T Manager resource requirements are based on the medium size VM, which is the minimum recommended form factor for NSX-T v3.0 and later. For more information, see [NSX Manager VM and Host Transport Node System Requirements](#).

Storage Requirements for Large Numbers of Pods

If you expect the cluster workload to run a large number of pods continuously, then increase the size of persistent disk storage allocated to the TKGI Database VM as follows:

Number of Pods	Persistent Disk Requirements (GB)
1,000 pods	20
5,000 pods	100
10,000 pods	200
50,000 pods	1,000

NSX-T Edge Node VMs and Sizes

The following table lists the resource requirements for each VM in the Edge Cluster.

VM	CPU Cores	Memory (GB)	Ephemeral Disk (GB)
NSX-T Edge Node 1	8	32	120
NSX-T Edge Node 2	8	32	120
TOTAL	16	64	240



Note: NSX-T 3.0 Edge Nodes can be deployed on Intel and AMD-based hosts.

Kubernetes Cluster Nodes VMs and Sizes

The following table lists sizing information for Kubernetes cluster node VMs. The size and resource consumption of these VMs are configurable in the **Plans** section of the Tanzu Kubernetes Grid Integrated Edition tile.

VM	CPU Cores	Memory (GB)	Ephemeral Disk (GB)	Persistent Disk
Control Plane Nodes	1 to 16	1 to 64	8 to 256	1 GB to 32 TB
Worker Nodes	1 to 16	1 to 64	8 to 256	1 GB to 32 TB

For illustrative purposes, consider the following example control plane node and a worker node requirements:

VM	CPU Cores	Memory (GB)	Ephemeral Disk (GB)	Persistent Disk (GB)
Control Plane Node	2	8	64	128
Worker Node	4	16	64	256

The following are the requirements for an example environment consisting of two Kubernetes clusters, each cluster consisting of three of the control plane nodes and five of the worker nodes described above:

VM	CPU Cores	Memory (GB)	Ephemeral Disk	Persistent Disk
Control Plane Nodes (3 nodes)	6	24	192 GB	384 GB
Worker Nodes (5 nodes)	20	80	320 GB	1280 GB
TOTAL (2 clusters)	52	208	1.0 TB	3.4 TB

Hardware Requirements

The following tables list the hardware requirements for RDP and MPD topologies for Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T.

RPD Hardware Requirements

The following table lists the hardware requirements for the RPD with vSAN topology.

VM	VM Count	CPU Cores (with HT)	Memory (GB)	NICS	Shared Persistent Disk (TB)
Management/Edge Cluster	3	16	98	2x 10GbE	1.5
Compute cluster1 (AZ1)	3	6	48	2x 10GbE	192
Compute cluster2 (AZ2)	3	6	48	2x 10GbE	192
Compute cluster3 (AZ3)	3	6	48	2x 10GbE	192



Note: The **CPU Cores** values assume the use of hyper-threading (HT).

The following table lists the hardware requirements for the RPD without vSAN topology.

VM	VM Count	CPU Cores (with HT)	Memory (GB)	NICS	Shared Persistent Disk (TB)
Management/Edge Cluster	3	16	98	2x 10GbE	1.5
Compute cluster1 (AZ1)	2	10	70	2x 10GbE	192
Compute cluster2 (AZ2)	2	10	70	2x 10GbE	192
Compute cluster3 (AZ3)	2	10	70	2x 10GbE	192



Note: The **CPU Cores** values assume the use of hyper-threading (HT).

MPD Hardware Requirements

The following table lists the hardware requirements for the MPD topology with a single (collapsed) cluster for all Management, Edge, and Compute nodes.

VM	VM Count	CPU Cores (with HT)	Memory (GB)	NICS	Shared Persistent Disk (TB)
Collapsed Cluster	3	32	236	4x 10GbE or 2x pNICs*	5.9

*If necessary, you can use two pNICs. For more information, see [Deploy a Fully Collapsed vSphere Cluster NSX-T on Hosts Running N-VDS Switches](#) in the NSX-T Data Center documentation.

Adding Hardware Capacity

To add hardware capacity to your Tanzu Kubernetes Grid Integrated Edition environment on

vSphere, do the following:

1. Add one or more ESXi hosts to the vSphere compute cluster. For more information, see the [VMware vSphere documentation](#).
2. Prepare each newly added ESXi host so that it becomes an ESXi transport node for NSX-T. For more information, see [Configure vSphere Networking for ESXi Hosts](#) in *Installing and Configuring NSX-T Data Center v3.0 for TKGI*.

Firewall Ports and Protocols Requirements for vSphere with NSX-T

This topic describes the firewall ports and protocols requirements for using VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) on vSphere with NSX-T integration.

If you are not using TKGI on vSphere with NSX-T, see one of the follow topics instead:

- [Firewall Ports and Protocols Requirements for vSphere \(Antrea and Flannel Networking\)](#)
- [Firewall Ports and Protocols Requirements \(Antrea and Flannel Networking\)](#)

Overview

Apps frequently require the ability to pass internal communication between system components on different networks.

Firewalls and Kubernetes Pod Security Policy are used to filter traffic and limit access in environments with strict inter-network access control policies and your apps require one or more conduits through a secured environment's firewalls.

VMware recommends that rather than using a Kubernetes Pod Security Policy to filter traffic between networks and TKGI system components and clusters that you instead use one of the following methods:

- Enable access to apps through standard Kubernetes load-balancers and ingress controller types. This enables you to designate specific ports and protocols as a firewall conduit.
- Enable access using the NSX-T load balancer and ingress. This enables you to configure external addresses and ports that are automatically mapped and resolved to internal/local addresses and ports.

For more information about vSphere with NSX-T port and protocol requirements, see [VMware Ports and Protocols](#) on the VMware site.

Consult the following tables when configuring port settings to install or upgrade TKGI or configure a Kubernetes cluster:

- [TKGI Users Ports and Protocols](#)
- [TKGI Core Ports and Protocols](#)
- [VMware Virtual Infrastructure Ports and Protocols](#)
- [VMware Optional Integration Ports and Protocols](#)



Note: To control which groups access deploying and scaling your organization's Tanzu Kubernetes Grid Integrated Edition-deployed Kubernetes clusters, configure your firewall settings as described on the Operator → TKGI API server lines below.

TKGI Ports and Protocols

The following tables list ports and protocols required for network communications between Tanzu Kubernetes Grid Integrated Edition v1.5.0 and later, and vSphere 6.7 and NSX-T 2.4.0.1 and later.

TKGI Users Ports and Protocols

The following table lists ports and protocols used for network communication between TKGI user interface components.

Source Component	Destination Component	Destination Protocol	Destination Port	Service
Admin/Operator Console	All System Components	TCP	22	ssh
Admin/Operator Console	All System Components	TCP	80	http
Admin/Operator Console	All System Components	TCP	443	https
Admin/Operator Console	BOSH Director	TCP	25555	bosh director rest api
Admin/Operator Console	NSX-T API VIP	TCP	443	https
Admin/Operator Console	Ops Manager	TCP	22	ssh
Admin/Operator Console	Ops Manager	TCP	443	https
Admin/Operator Console	TKGI Controller	TCP	9021	tkgi api server
Admin/Operator Console	vCenter Server	TCP	443	https
Admin/Operator Console	vCenter Server	TCP	5480	vami
Admin/Operator Console	vSphere ESXI Hosts Mgmt. vmknic	TCP	902	ideafarm-door
Admin/Operator and Developer Consoles	Harbor Private Image Registry	TCP	80	http
Admin/Operator and Developer Consoles	Harbor Private Image Registry	TCP	443	https
Admin/Operator and Developer Consoles	Harbor Private Image Registry	TCP	4443	notary
Admin/Operator and Developer Consoles	Kubernetes App Load-Balancer Svc	TCP/UDP	Varies	varies with apps

Source Component	Destination Component	Destination Protocol	Destination Port	Service
Admin/Operator and Developer Consoles	Kubernetes Cluster API Server -LB VIP	TCP	8443	httpsca
Admin/Operator and Developer Consoles	Kubernetes Cluster Ingress Controller	TCP	80	http
Admin/Operator and Developer Consoles	Kubernetes Cluster Ingress Controller	TCP	443	https
Admin/Operator and Developer Consoles	Kubernetes Cluster Worker Node	TCP/UDP	30000-32767	kubernetes nodeport
Admin/Operator and Developer Consoles	TKGI Controller	TCP	8443	httpsca
All User Consoles (Operator, Developer, Consumer)	Kubernetes App Load-Balancer Svc	TCP/UDP	Varies	varies with apps
All User Consoles (Operator, Developer, Consumer)	Kubernetes Cluster Ingress Controller	TCP	80	http
All User Consoles (Operator, Developer, Consumer)	Kubernetes Cluster Ingress Controller	TCP	443	https
All User Consoles (Operator, Developer, Consumer)	Kubernetes Cluster Worker Node	TCP/UDP	30000-32767	kubernetes nodeport



Note: The `type:NodePort` Service type is not supported for TKGI deployments on vSphere with NSX-T. Only `type:LoadBalancer` and Services associated with Ingress rules are supported on vSphere with NSX-T.

TKGI Core Ports and Protocols

The following table lists ports and protocols used for network communication between core TKGI components.

Source Component	Destination Component	Destination Protocol	Destination Port	Service
All System Components	Corporate Domain Name Server	TCP/UDP	53	dns
All System Components	Network Time Server	UDP	123	ntp
All System Components	vRealize LogInsight	TCP/UDP	514/1514	syslog/tls syslog
All System Control Plane Components	AD/LDAP Directory Server	TCP/UDP	389/636	ldap/ldaps
Ops Manager	Admin/Operator Console	TCP	22	ssh
Ops Manager	BOSH Director	TCP	6868	bosh agent http
Ops Manager	BOSH Director	TCP	8443	httpsca

Source Component	Destination Component	Destination Protocol	Destination Port	Service
Ops Manager	BOSH Director	TCP	8844	credhub
Ops Manager	BOSH Director	TCP	25555	bosh director rest api
Ops Manager	Harbor Private Image Registry	TCP	22	ssh
Ops Manager	Kubernetes Cluster Control Plane/Etcd Node	TCP	22	ssh
Ops Manager	Kubernetes Cluster Worker Node	TCP	22	ssh
Ops Manager	NSX-T API VIP	TCP	443	https
Ops Manager	NSX-T Manager/Controller Node	TCP	22	ssh
Ops Manager	NSX-T Manager/Controller Node	TCP	443	https
Ops Manager	TKGI Controller	TCP	22	ssh
Ops Manager	TKGI Controller	TCP	8443	httpsca
Ops Manager	vCenter Server	TCP	443	https
Ops Manager	vSphere ESXI Hosts Mgmt. vmknic	TCP	443	https
BOSH Director	NSX-T API VIP	TCP	443	https
BOSH Director	vCenter Server	TCP	443	https
BOSH Director	vSphere ESXI Hosts Mgmt. vmknic	TCP	443	https
BOSH Compilation Job VM	BOSH Director	TCP	4222	bosh nats server
BOSH Compilation Job VM	BOSH Director	TCP	25250	bosh blobstore
BOSH Compilation Job VM	BOSH Director	TCP	25923	health monitor daemon
BOSH Compilation Job VM	Harbor Private Image Registry	TCP	443	https
BOSH Compilation Job VM	Harbor Private Image Registry	TCP	8853	bosh dns health
TKGI Controller	BOSH Director	TCP	4222	bosh nats server
TKGI Controller	BOSH Director	TCP	8443	httpsca
TKGI Controller	BOSH Director	TCP	25250	bosh blobstore
TKGI Controller	BOSH Director	TCP	25555	bosh director rest api
TKGI Controller	BOSH Director	TCP	25923	health monitor daemon

Source Component	Destination Component	Destination Protocol	Destination Port	Service
TKGI Controller	Kubernetes Cluster Control Plane/Etcd Node	TCP	8443	httpsca
TKGI Controller	TKGI Database VM	TCP	3306	tkgi db proxy
TKGI Controller	NSX-T API VIP	TCP	443	https
TKGI Controller	vCenter Server	TCP	443	https
Harbor Private Image Registry	BOSH Director	TCP	4222	bosh nats server
Harbor Private Image Registry	BOSH Director	TCP	25250	bosh blobstore
Harbor Private Image Registry	BOSH Director	TCP	25923	health monitor daemon
Harbor Private Image Registry	IP NAS Storage Array	TCP	111	nfs rpc portmapper
Harbor Private Image Registry	IP NAS Storage Array	TCP	2049	nfs
Harbor Private Image Registry	Public CVE Source Database	TCP	443	https
kube-system pod/telemetry-agent	TKGI Controller	TCP	24224	fluentd out_forward
Kubernetes Cluster Ingress Controller	NSX-T API VIP	TCP	443	https
Kubernetes Cluster Control Plane/Etcd Node	BOSH Director	TCP	4222	bosh nats server
Kubernetes Cluster Control Plane/Etcd Node	BOSH Director	TCP	25250	bosh blobstore
Kubernetes Cluster Control Plane/Etcd Node	BOSH Director	TCP	25923	health monitor daemon
Kubernetes Cluster Control Plane/Etcd Node	Kubernetes Cluster Control Plane/Etcd Node	TCP	2379	etcd client
Kubernetes Cluster Control Plane/Etcd Node	Kubernetes Cluster Control Plane/Etcd Node	TCP	2380	etcd server
Kubernetes Cluster Control Plane/Etcd Node	Kubernetes Cluster Control Plane/Etcd Node	TCP	8443	httpsca
Kubernetes Cluster Control Plane/Etcd Node	Kubernetes Cluster Control Plane/Etcd Node	TCP	8853	bosh dns health
Kubernetes Cluster Control Plane/Etcd Node	Kubernetes Cluster Worker Node	TCP	4194	cadvisor
Kubernetes Cluster Control Plane/Etcd Node	Kubernetes Cluster Worker Node	TCP	10250	kubelet api
Kubernetes Cluster Control Plane/Etcd Node	Kubernetes Cluster Worker Node	TCP	31194	cadvisor

Source Component	Destination Component	Destination Protocol	Destination Port	Service
Kubernetes Cluster Control Plane/Etcd Node	NSX-T API VIP	TCP	443	https
Kubernetes Cluster Control Plane/Etcd Node	TKGI Controller	TCP	8443	httpsca
Kubernetes Cluster Control Plane/Etcd Node	TKGI Controller	TCP	8853	bosh dns health
Kubernetes Cluster Control Plane/Etcd Node	vCenter Server	TCP	443	https
Kubernetes Cluster Worker Node	BOSH Director	TCP	4222	bosh nats server
Kubernetes Cluster Worker Node	BOSH Director	TCP	25250	bosh blobstore
Kubernetes Cluster Worker Node	BOSH Director	TCP	25923	health monitor daemon
Kubernetes Cluster Worker Node	Harbor Private Image Registry	TCP	443	https
Kubernetes Cluster Worker Node	Harbor Private Image Registry	TCP	8853	bosh dns health
Kubernetes Cluster Worker Node	IP NAS Storage Array	TCP	111	nfs rpc portmapper
Kubernetes Cluster Worker Node	IP NAS Storage Array	TCP	2049	nfs
Kubernetes Cluster Worker Node	Kubernetes Cluster Control Plane/Etcd Node	TCP	8443	httpsca
Kubernetes Cluster Worker Node	Kubernetes Cluster Control Plane/Etcd Node	TCP	8853	bosh dns health
Kubernetes Cluster Worker Node	Kubernetes Cluster Control Plane/Etcd Node	TCP	10250	kubelet api
pks-system pod/cert-generator	TKGI Controller	TCP	24224	fluentd out_forward
pks-system pod/fluent-bit	TKGI Controller	TCP	24224	fluentd out_forward

VMware Ports and Protocols

The following tables list ports and protocols required for network communication between VMware components. For additional information, see [VMware Ports and Protocols](#).

VMware Virtual Infrastructure Ports and Protocols

The following table lists ports and protocols used for network communication between VMware virtual infrastructure components.

Source Component	Destination Component	Destination Protocol	Destination Port	Service
vCenter Server	NSX-T Manager/Controller Node	TCP	8080	http alt
vCenter Server	vSphere ESXI Hosts Mgmt. vmknic	TCP	443	https
vCenter Server	vSphere ESXI Hosts Mgmt. vmknic	TCP	8080	http alt
vCenter Server	vSphere ESXI Hosts Mgmt. vmknic	TCP	9080	io filter storage
vSphere ESXI Hosts Mgmt. vmknic	NSX-T Manager/Controller Node	TCP	443	https
vSphere ESXI Hosts Mgmt. vmknic	NSX-T Manager/Controller Node	TCP	1235	netcpa
vSphere ESXI Hosts Mgmt. vmknic	NSX-T Manager/Controller Node	TCP	5671	amqp traffic
vSphere ESXI Hosts Mgmt. vmknic	NSX-T Manager/Controller Node	TCP	8080	http alt
vSphere ESXI Hosts Mgmt. vmknic	vCenter Server	UDP	902	ideafarm-door
vSphere ESXI Hosts Mgmt. vmknic	vCenter Server	TCP	9084	update manager
vSphere ESXI Hosts Mgmt. vmknic	vSphere ESXI Hosts Mgmt. vmknic	TCP	8182	vsphere ha
vSphere ESXI Hosts Mgmt. vmknic	vSphere ESXI Hosts Mgmt. vmknic	UDP	8182	vsphere ha
vSphere ESXI Hosts vMotion vmknic	vSphere ESXI Hosts vMotion vmknic	TCP	8000	vmotion
vSphere ESXI Hosts IP Storage vmknic	IP NAS Storage Array	TCP	111	nfs rpc portmapper
vSphere ESXI Hosts IP Storage vmknic	IP NAS Storage Array	TCP	2049	nfs
vSphere ESXI Hosts IP Storage vmknic	IP NAS Storage Array	TCP	3260	iscsi
vSphere ESXI Hosts vSAN vmknic	vSphere ESXI Hosts vSAN vmknic	TCP	2233	vsan transport
vSphere ESXI Hosts vSAN vmknic	vSphere ESXI Hosts vSAN vmknic	UDP	12321	unicast agent

Source Component	Destination Component	Destination Protocol	Destination Port	Service
vSphere ESXI Hosts vSAN vmknic	vSphere ESXI Hosts vSAN vmknic	UDP	12345	vsan cluster svc
vSphere ESXI Hosts vSAN vmknic	vSphere ESXI Hosts vSAN vmknic	UDP	23451	vsan cluster svc
vSphere ESXI Hosts TEP vmknic	vSphere ESXI Hosts TEP vmknic	UDP	3784	bfd
vSphere ESXI Hosts TEP vmknic	vSphere ESXI Hosts TEP vmknic	UDP	3785	bfd
vSphere ESXI Hosts TEP vmknic	vSphere ESXI Hosts TEP vmknic	UDP	6081	geneve
vSphere ESXI Hosts TEP vmknic	NSX-T Edge TEP vNIC	UDP	3784	bfd
vSphere ESXI Hosts TEP vmknic	NSX-T Edge TEP vNIC	UDP	3785	bfd
vSphere ESXI Hosts TEP vmknic	NSX-T Edge TEP vNIC	UDP	6081	geneve
vSphere ESXI Hosts TEP vmknic	NSX-T Manager/Controller	TCP	1234	NSX messaging
NSX-T Manager/Controller Node	NSX-T API VIP	TCP	443	https
NSX-T Manager/Controller Node	NSX-T Manager/Controller Node	TCP	443	https
NSX-T Manager/Controller Node	NSX-T Manager/Controller Node	TCP	5671	amqp traffic
NSX-T Manager/Controller Node	NSX-T Manager/Controller Node	TCP	8080	http alt
NSX-T Manager/Controller Node	NSX-T Manager/Controller Node	TCP	9000	loginsight ingestion api
NSX-T Manager/Controller Node	Traceroute Destination	UDP	33434-33523	traceroute
NSX-T Manager/Controller Node	vCenter Server	TCP	80	http
NSX-T Manager/Controller Node	vCenter Server	TCP	443	https
NSX-T Manager/Controller Node	vSphere ESXI Hosts Mgmt. vmknic	TCP	443	https
NSX-T Edge Management	NSX-T Edge Management	TCP	1167	DHCP backend
NSX-T Edge Management	NSX-T Edge Management	TCP	2480	Nestdb
NSX-T Edge Management	NSX-T Edge Management	UDP	3784	bfd

Source Component	Destination Component	Destination Protocol	Destination Port	Service
NSX-T Edge Management	NSX-T Edge Management	UDP	50263	high-availability
NSX-T Edge Management	NSX-T Manager/Controller Node	TCP	443	https
NSX-T Edge Management	NSX-T Manager/Controller Node	TCP	1235	netcpa
NSX-T Edge Management	NSX-T Manager/Controller Node	TCP	5671	amqp traffic
NSX-T Edge Management	NSX-T Manager/Controller Node	TCP	8080	http alt
NSX-T Edge Management	Traceroute Destination	UDP	33434-33523	traceroute
NSX-T Edge TEP vNIC	NSX-T Edge TEP vNIC	UDP	3784	bfd
NSX-T Edge TEP vNIC	NSX-T Edge TEP vNIC	UDP	3785	bfd
NSX-T Edge TEP vNIC	NSX-T Edge TEP vNIC	UDP	6081	geneve
NSX-T Edge TEP vNIC	NSX-T Edge TEP vNIC	UDP	50263	high-availability
NSX-T Edge TEP vNIC	vSphere ESXI Hosts TEP vmknic	UDP	3784	bfd
NSX-T Edge TEP vNIC	vSphere ESXI Hosts TEP vmknic	UDP	3785	bfd
NSX-T Edge TEP vNIC	vSphere ESXI Hosts TEP vmknic	UDP	6081	geneve
NSX-T Edge Tier-0 Uplink IP(s) / HA VIP	Physical Network Router	TCP	179	bgp
NSX-T Edge TEP vNIC	NSX-T Manager/Controller	TCP	1234	NSX messaging
NSX-T Tier-1 Router	Kubernetes cluster Pods and Worker Nodes	TCP	80	http
NSX-T Tier-1 Router	Kubernetes cluster Pods and Worker Nodes	TCP	443	https
NSX-T Tier-1 Router	Kubernetes cluster Pods and Worker Nodes	TCP	8443	httpsca
Physical Network Router	NSX-T Edge Tier-0 Uplink IP(s) / HA VIP	TCP	179	bgp

VMware Optional Integration Ports and Protocols

The following table lists ports and protocols used for network communication between optional VMware integrations.

Source Component	Destination Component	Destination Protocol	Destination Port	Service
------------------	-----------------------	----------------------	------------------	---------

Admin/Operator Console	vRealize Operations Manager	TCP	443	https
vRealize Operations Manager	Kubernetes Cluster API Server -LB VIP	TCP	8443	httpsca
vRealize Operations Manager	NSX-T API VIP	TCP	443	https
vRealize Operations Manager	TKGI Controller	TCP	8443	httpsca
vRealize Operations Manager	Kubernetes Cluster API Server -LB VIP	TCP	8443	httpsca
Admin/Operator Console	vRealize LogInsight	TCP	443	https
Kubernetes Cluster Ingress Controller	vRealize LogInsight	TCP	9000	ingestion api
Kubernetes Cluster Control Plane/Etcd Node	vRealize LogInsight	TCP	9000	ingestion api
Kubernetes Cluster Control Plane/Etcd Node	vRealize LogInsight	TCP	9543	ingestion api - tls
Kubernetes Cluster Worker Node	vRealize LogInsight	TCP	9000	ingestion api
Kubernetes Cluster Worker Node	vRealize LogInsight	TCP	9543	ingestion api - tls
NSX-T Manager/Controller Node	vRealize LogInsight	TCP	9000	ingestion api
TKGI Controller	vRealize LogInsight	TCP	9000	ingestion api
Admin/Operator and Developer Consoles	Wavefront SaaS APM	TCP	443	https
kube-system pod/wavefront-proxy	Wavefront SaaS APM	TCP	443	https
kube-system pod/wavefront-proxy	Wavefront SaaS APM	TCP	8443	httpsca
pks-system pod/wavefront-collector	TKGI Controller	TCP	24224	fluentd out_forward
Admin/Operator Console	vRealize Network Insight Platform	TCP	443	https
Admin/Operator Console	vRealize Network Insight Proxy	TCP	22	ssh
vRealize Network Insight Proxy	Kubernetes Cluster API Server -LB VIP	TCP	8443	httpsca
vRealize Network Insight Proxy	NSX-T API VIP	TCP	22	ssh
vRealize Network Insight Proxy	NSX-T API VIP	TCP	443	https
vRealize Network Insight Proxy	TKGI Controller	TCP	8443	httpsca
vRealize Network Insight Proxy	TKGI Controller	TCP	9021	tkgi api server

Creating VMware NSX Objects for Tanzu Kubernetes Grid

Integrated Edition

Installing VMware Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T requires the creation of NSX IP blocks for Kubernetes node and pod networks, as well as a Floating IP Pool from which you can assign routable IP addresses to cluster resources.

Create separate NSX-T IP Blocks for the [node networks](#) and the [pod networks](#). The subnets for both nodes and pods should have a size of 256 (/16). For more information, see [Plan IP Blocks](#) and [Reserved IP Blocks](#). For more information about NSX-T IP Blocks, see [Advanced IP Address Management](#) in the *VMware NSX-T Data Center* documentation.

- **NODE-IP-BLOCK** is used by Tanzu Kubernetes Grid Integrated Edition to assign address space to Kubernetes control plane and worker nodes when new clusters are deployed or a cluster increases its scale.
- **POD-IP-BLOCK** is used by the NSX-T Container Plug-in (NCP) to assign address space to Kubernetes pods through the Container Networking Interface (CNI).

In addition, create a Floating IP Pool from which to assign routable IP addresses to components. This network provides your load balancing address space for each Kubernetes cluster created by Tanzu Kubernetes Grid Integrated Edition. The network also provides IP addresses for Kubernetes API access and Kubernetes exposed services. For example, [10.172.2.0/24](#) provides 256 usable IPs. This network is used when creating the virtual IP pools, or when the services are deployed. You enter this network in the **Floating IP Pool ID** field in the **Networking** pane of the Tanzu Kubernetes Grid Integrated Edition tile.

Complete the following instructions to create the required NSX-T network objects.

Create the Nodes IP Block

1. In NSX-T Manager, go to **Advanced Networking & Security** > **Networking** > **IPAM**.
2. Add a new IP Block for Kubernetes Nodes. For example:
 - **Name:** NODES-IP-BLOCK
 - **CIDR:** 192.168.0.0/16

New IP Block

Name *	<input type="text" value="nodes-ip-block"/>
Description	nodes-ip-block 40.0.0.0/16 NAT-mode (non-routable)
CIDR *	<input type="text" value="40.0.0.0/16"/>

ADD

3. Verify creation of the Nodes IP Block.

New IP Block

Name* PKS-POD-IP-BLOCK

Description

CIDR* 172.16.0.0/16

CANCEL **ADD**

4. Record the UUID of the Nodes IP Block object. You use this UUID when you install Tanzu Kubernetes Grid Integrated Edition with NSX-T.

	ID	CIDR
<input type="checkbox"/>	b5d3...2e56	10.40.14.0/24
<input checked="" type="checkbox"/>	b91093ee-2df8-4e12-8070-3cee338807d0	192.168.0.0/16
<input type="checkbox"/>	84c6...c42e	172.16.0.0/16
<input type="checkbox"/>	e133...9540	172.16.0.0/16

Create the Pods IP Block

1. In NSX-T Manager, go to **Advanced Networking & Security > Networking > IPAM**.
2. Add a new IP Block for Pods. For example:
 - **Name:** TKGI-PODS-IP-BLOCK
 - **CIDR:** 172.16.0.0/16

New IP Block

Name*	PKS-POD-IP-BLOCK
Description	
CIDR*	172.16.0.0/16
<input type="button" value="CANCEL"/> <input type="button" value="ADD"/>	

3. Verify creation of the Pods IP Block.

The screenshot shows the NSX-T IPAM interface under the Advanced Networking & Security tab. The left sidebar is collapsed. The main area displays a table of IP blocks. A green success message at the top right states "IP Block 'pods-ip-block' was successfully created." The table has columns for ID, IP Blocks, and CIDR. It lists three entries: "nodes-ip-block" (ID: c5f0...ebd2, CIDR: 40.0.0.0/16), "pods-ip-block" (ID: fead...c8d4, CIDR: 40.1.0.0/16), and the newly created "pods-ip-block" (ID: 84c6e69b-0361-460f-8c1a-a7e0cf4cc42e, CIDR: 172.16.0.0/16).

4. Record the UUID of the Pods IP Block object. You use this UUID when you install Tanzu Kubernetes Grid Integrated Edition with NSX-T.

The screenshot shows the NSX IPAM interface under the IPAM tab. The left sidebar is expanded, showing categories like Networking, IPAM, and Tools. The main area displays a table of IP blocks. The "PKS-POD-IP-BLOCK" entry is selected, highlighted with a blue border. Its details are shown in the bottom right: ID: 84c6e69b-0361-460f-8c1a-a7e0cf4cc42e, CIDR: 172.16.0.0/16.

Create Floating IP Pool

- In NSX-T Manager, go to **Advanced Networking & Security > Inventory > Groups > IP Pool**.

ID	Subnets	Allocations
2e08...78ae	1	0 of 128
64fa...b337	1	0 of 128
104f...615c	1	6 of 10
86a7...411d	3	24 of 90

- Add a new Floating IP Pool. For example:

- ❖ **Name:** TKGI-FLOATING-IP-POOL
- ❖ **IP Ranges:** 10.40.14.10 - 10.40.14.253
- ❖ **Gateway:** 10.40.14.254
- ❖ **CIDR:** 10.40.14.0/24

IP Ranges*	Gateway	CIDR*	DNS Servers	DNS Suffix
10.40.14.10 - 10.40.14.253	10.40.14.254	10.40.14.0/24		

- Verify creation of the Floating IP Pool.

	ID	Subnets	Allocations
<input type="checkbox"/>	PKS-FLOATING-IP-POOL	78c6...f709	1 0 of 244
<input type="checkbox"/>	SI_Destination_IP_Pool	2e08...78ae	1 0 of 128
<input type="checkbox"/>	SI_Source_IP_Pool	64fa...b337	1 0 of 128
<input type="checkbox"/>	TEP-ESXI-POOL	104f...615c	1 6 of 10
<input type="checkbox"/>	ip-pool-vips	86a7...411d	3 24 of 90

- Get the UUID of the Floating IP Pool object. You use this UUID when you install Tanzu Kubernetes Grid Integrated Edition with NSX-T.

Next Step

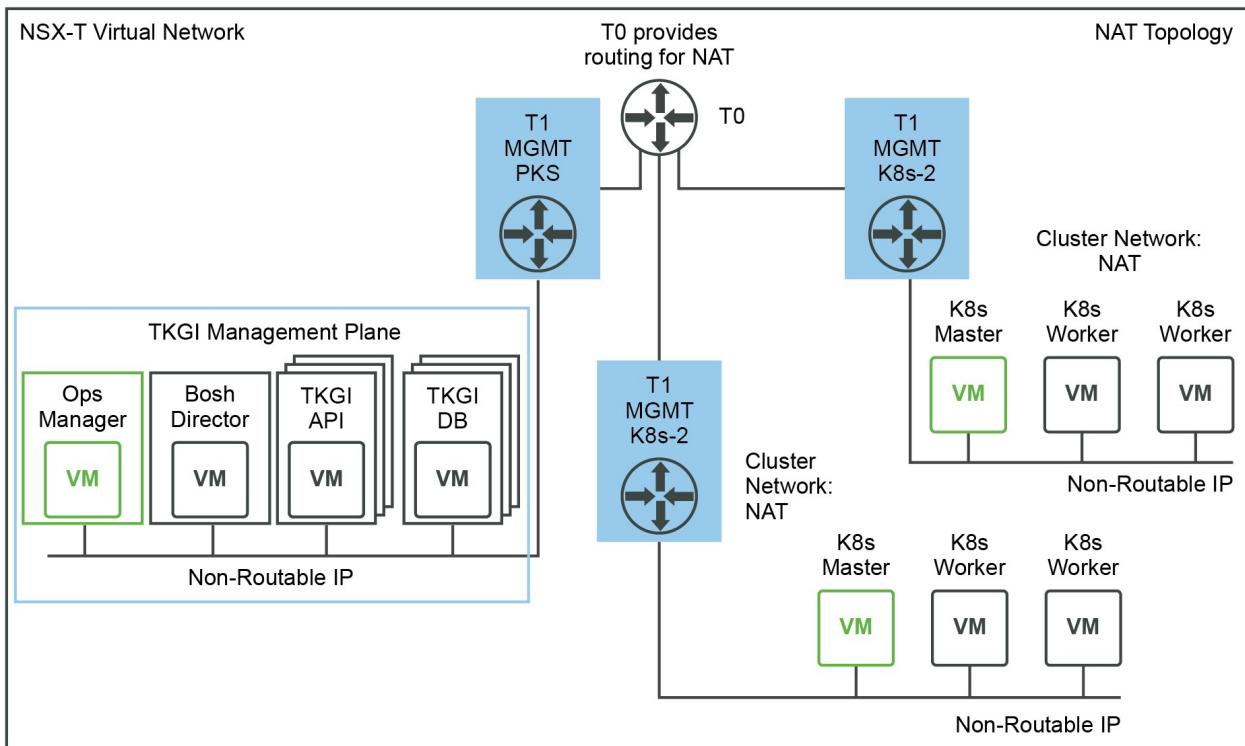
After you complete this procedure, follow the instructions in [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T](#).

NSX-T Deployment Topologies for Tanzu Kubernetes Grid Integrated Edition

This topic describes supported topologies for deploying VMware Tanzu Kubernetes Grid Integrated Edition with NSX-T.

NAT Topology

The following figure shows a Network Address Translation (NAT) deployment:



[View a larger version of this image.](#)

This topology has the following characteristics:

- TKGI Management Plane (Ops Manager, BOSH Director, and Tanzu Kubernetes Grid Integrated Edition VMs such as the TKGI API and TKGI Database VMs) components are all located on a logical switch that has undergone Network Address Translation on a TO.
- Kubernetes cluster control plane and worker nodes are located on a logical switch that has undergone Network Address Translation on a TO. This requires DNAT rules to allow access to Kubernetes APIs.

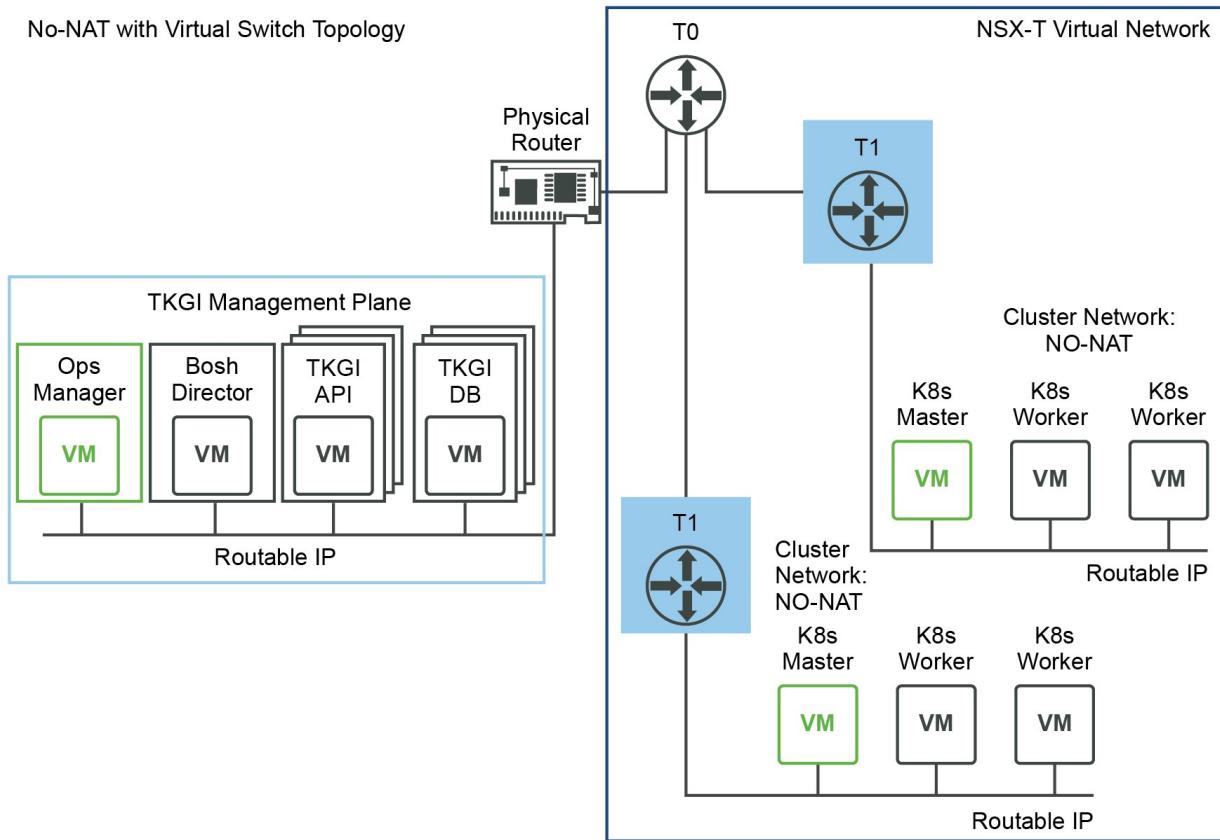
No-NAT Topology

A No-NAT topology uses a routable IP subnet for the TKGI Management network and for Kubernetes nodes.

There are two flavors of No-NAT topology: No-NAT with Virtual Switch or No-NAT with Logical Switch.

No-NAT with Virtual Switch (VSS/VDS) Topology

The following figure shows a No-NAT with Virtual Switch (VSS/VDS) deployment:



[View a larger version of this image.](#)

This topology has the following characteristics:

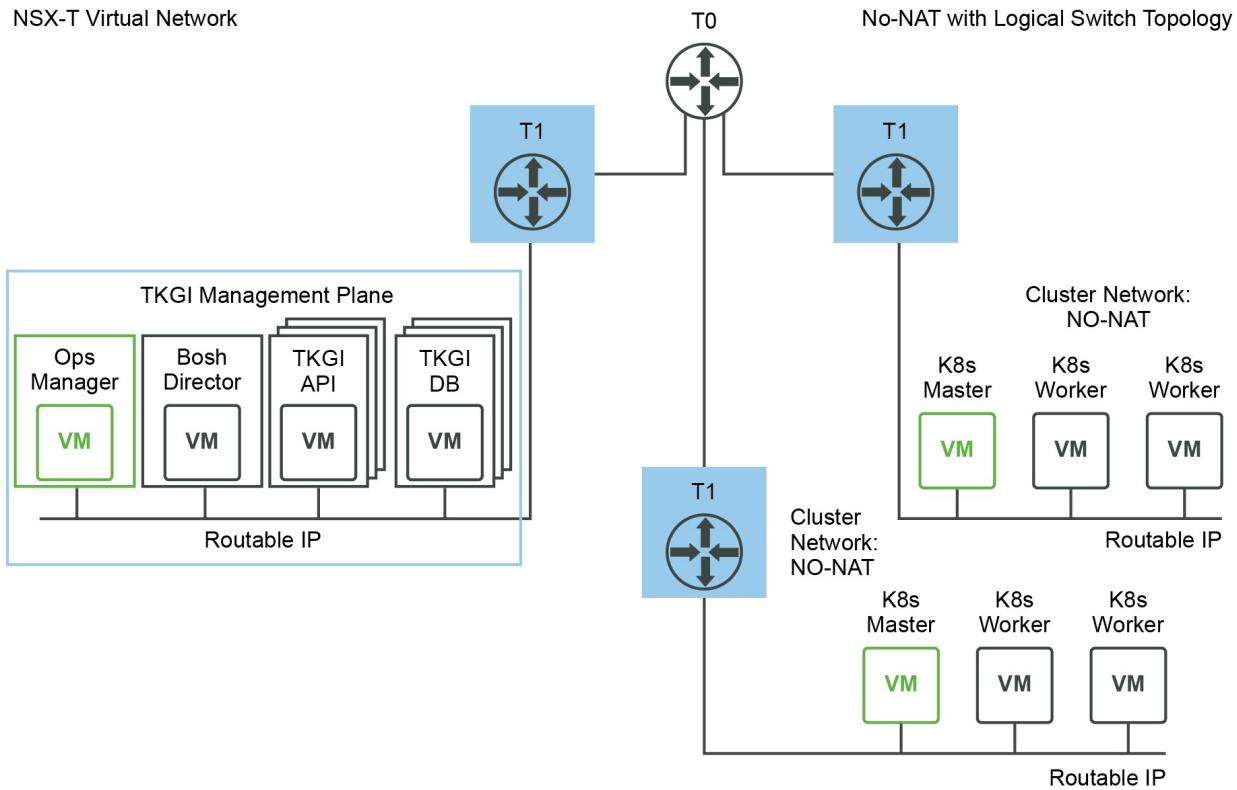
- TKGI Management Plane (Ops Manager, BOSH Director, and Tanzu Kubernetes Grid Integrated Edition VMs such as the TKGI API and TKGI Database VMs) components are using corporate routable IP addresses.
- Kubernetes cluster control plane and worker nodes are using corporate routable IP addresses.
- The TKGI Management Plane is deployed outside of the NSX-T network and the Kubernetes clusters are deployed and managed within the NSX-T network. Since BOSH needs routable access to the Kubernetes Nodes to monitor and manage them, the Kubernetes Nodes need routable access.
- (Optional) You can use multiple vCenter Servers to separate management plane components.

Consider the following caveats before using multiple vCenter Servers:

- This configuration is only supported through Ops Manager.
- Workload clusters must all use the same vCenter Server.

No-NAT with Logical Switch (NSX-T) Topology

The following figure shows a No-NAT with Logical Switch (NSX-T) deployment:



[View a larger version of this image.](#)

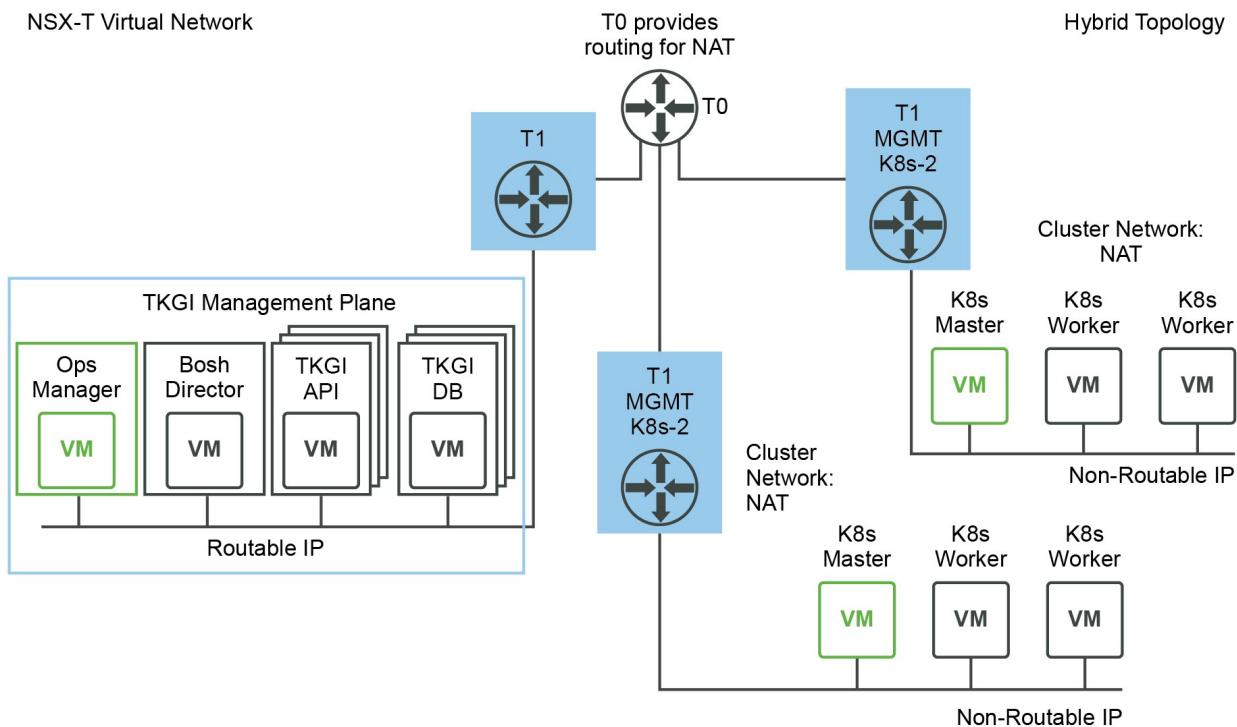
This topology has the following characteristics:

- TKGI Management Plane (Ops Manager, BOSH Director, and Tanzu Kubernetes Grid Integrated Edition VMs such as the TKGI API and TKGI Database VMs) components are using corporate routable IP addresses.
- Kubernetes cluster control plane and worker nodes are using corporate routable IP addresses.
- The TKGI Management Plane is deployed inside of the NSX-T network. Both the TKGI Management Plane components (VMs) and the Kubernetes Nodes use corporate routable IP addresses.

Hybrid Topology

With a hybrid topology, the TKGI Management Network is on a routable subnet, while the Kubernetes Nodes Network uses a non-routable subnet (NAT mode is checked in the TKGI tile).

The following figure shows a hybrid topology deployment:



[View a larger version of this image.](#)

This topology has the following characteristics:

- TKGI Management Plane (Ops Manager, BOSH Director, and Tanzu Kubernetes Grid Integrated Edition VMs such as the TKGI API and TKGI Database VMs) components are using corporate routable IP addresses.
- Kubernetes cluster control plane and worker nodes are located on a logical switch that has undergone Network Address Translation on a T0. This requires DNAT rules to allow access to Kubernetes APIs.

vSAN Stretched Cluster Topologies

A vSAN stretched cluster topology runs across two sites to support highly resilient workloads. vSAN stretched cluster topologies include:

- Topology 1: Dedicated vSphere clusters
- Topology 2: Fully collapsed vSphere clusters

For more information about vSAN stretched cluster topologies for TKGI, see [Solution Guide for Enabling Highly Resilient Kubernetes Workloads Using vSAN Stretched Clusters](#).

vSphere with VMware NSX Cluster Objects

This topic lists and describes the vSphere VMs and NSX-T objects that VMware Tanzu Kubernetes Grid Integrated Edition creates when you create a Kubernetes cluster. When you delete a Kubernetes cluster, Tanzu Kubernetes Grid Integrated Edition removes these objects.

For information about creating a Kubernetes cluster using Tanzu Kubernetes Grid Integrated Edition, see [Creating Clusters](#). For information about deleting a Kubernetes cluster using Tanzu Kubernetes Grid Integrated Edition, see [Deleting Clusters](#).

vSphere Virtual Machines

When a new Kubernetes cluster is created, Tanzu Kubernetes Grid Integrated Edition creates the following virtual machines (VMs) in the designated vSphere cluster:

Object Number	Object Description
1 or 3	Kubernetes control plane nodes. The number depends on the plan used to create the cluster.
1 or more	Kubernetes worker nodes. The number depends on the plan used to create the cluster, or the number specified during cluster creation.



Note: For production clusters, three control plane nodes are required, and a minimum of three worker nodes are required. See [Requirements for Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T](#) for more information.

NSX-T Logical Switches

When a new Kubernetes cluster is created, Tanzu Kubernetes Grid Integrated Edition creates the following [NSX-T logical switches](#):

Object Number	Object Description
1	Logical switch for Kubernetes control plane and worker nodes.
1	Logical switch for each Kubernetes namespace: <code>default, kube-public, kube-system, pks-infrastructure</code> .
1	Logical switch for the NSX-T load balancer associated with the Kubernetes cluster.

NSX-T Tier-1 Logical Routers

When a new Kubernetes cluster is created, Tanzu Kubernetes Grid Integrated Edition creates the following [NSX-T Tier-1 logical routers](#):

Object Number	Object Description
1	Tier-1 router for Kubernetes control plane and worker nodes. Name: <code>cluster-router</code> .
1	Tier-1 router for each Kubernetes namespace: <code>default, kube-public, kube-system, pks-infrastructure</code> .
1	Tier-1 router for the NSX-T load balancer associated with the Kubernetes cluster.

NSX-T Load Balancers

For each Kubernetes cluster created, Tanzu Kubernetes Grid Integrated Edition creates a single instance of a small [NSX-T load balancer](#). This load balancer contains the objects listed in the following table:

Object Number	Object Description
1	Virtual Server (VS) to access Kubernetes control plane API on port 8443.
1	Server Pool containing the 3 Kubernetes control plane nodes.
1	VS for HTTP Ingress Controller.
1	VS for HTTPS Ingress Controller.

The IP address allocated to each VS is derived from the **Floating IP Pool** that was created for use with Tanzu Kubernetes Grid Integrated Edition. The VS for the HTTP Ingress Controller and the VS for the HTTPS Ingress Controller use the same IP address.

NSX-T DDI/IPAM

For each Kubernetes cluster created, Tanzu Kubernetes Grid Integrated Edition extracts and allocates the following NSX-T subnets from the **IP blocks** created in preparation for installing Tanzu Kubernetes Grid Integrated Edition with NSX-T:

Object Number	Object Description
1	A /24 subnet from the Nodes IP Block will be extracted and allocated for the Kubernetes control plane and worker nodes.
1	A /24 subnet from the Pods IP Block will be extracted and allocated for each Kubernetes namespace: <code>default, kube-public, kube-system, pks-infrastructure</code> .

NSX-T Tier-0 Logical Routers

For each Kubernetes cluster created, Tanzu Kubernetes Grid Integrated Edition defines the following **NSX-T NAT rules** on the Tier-0 logical router:

Object Number	Object Description
1	SNAT rule created for each Kubernetes namespace: <code>default, kube-public, kube-system, pks-infrastructure</code> using 1 IP from the Floating IP Pool as translated IP address.
1	(NAT topology only) SNAT rule created for each Kubernetes cluster using 1 IP from the Floating IP Pool as translated IP address. The Kubernetes cluster subnet is derived from the Nodes IP Block using a /24 netmask.

NSX-T Distributed Firewall (DFW) Rules

For each Kubernetes cluster created, Tanzu Kubernetes Grid Integrated Edition defines the following **NSX-T distributed firewall rules**:

Object Amount	Object Description

-
- ```
1 DFW rule for kube-dns, applied to CoreDNS pod logical port: Source=Kubernetes worker node (hosting
the DNS Pod); Destination=Any; Port: TCP/8080; Action: allow
1 DFW rule for Validator in pks-system namespace, applied to Validator pod logical port: Source=Kubernetes
worker node (hosting the DNS Pod) IP Address; Destination=Any; Port: TCP/9000; Action: allow
```
- 

## Network Planning for Installing Tanzu Kubernetes Grid Integrated Edition with VMware NSX

This topic describes how to plan your environment before installing VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) on VMware vSphere with NSX integration.

## Overview

Before installing VMware Tanzu Kubernetes Grid Integrated Edition on VMware vSphere with NSX integration, plan your environment as described in the following sections:

- [Prerequisites](#)
- [Understand Component Interactions](#)
- [Plan Deployment Topology](#)
- [Plan Network CIDRs](#)
- [Gather Other Required IP Addresses](#)

## Prerequisites

Familiarize yourself with the following VMware documentation:

- [vSphere, vCenter, vSAN, and ESXi documentation](#)
- [VMware NSX Documentation](#)
- [NSX Container Plugin \(NCP\) documentation](#)

Familiarize yourself with the following related documentation:

- [Ops Manager documentation](#)
- [BOSH documentation](#)
- [Kubernetes documentation](#)
- [Docker documentation](#)
- [containerd documentation](#)

Review the following Tanzu Kubernetes Grid Integrated Edition documentation:

- [VMware vSphere with NSX Version Requirements](#)

- [Hardware Requirements for Tanzu Kubernetes Grid Integrated Edition on VMware vSphere with NSX](#)
- [VMware Ports and Protocols](#) on the VMware site.
- [Network Objects Created by NSX for Tanzu Kubernetes Grid Integrated Edition](#)

## Understand Component Interactions

Tanzu Kubernetes Grid Integrated Edition on VMware vSphere with NSX requires the following component interactions:

- vCenter, NSX Manager Nodes, NSX Edge Nodes, and ESXi hosts must be able to communicate with each other.
- The BOSH Director VM must be able to communicate with vCenter and the NSX Management Cluster.
- The BOSH Director VM must be able to communicate with all nodes in all Kubernetes clusters.
- Each Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes cluster deploys the NSX Node Agent and the Kube Proxy that run as BOSH-managed processes on each worker node.
- NCP runs as a BOSH-managed process on the Kubernetes control plane node. In a multi-control plane node deployment, the NCP process runs on all control plane nodes, but is active only on one control plane node. If the NCP process on an active control plane node is unresponsive, BOSH activates another NCP process.

## Plan Deployment Topology

Review the [Deployment Topologies](#) for Tanzu Kubernetes Grid Integrated Edition on VMware vSphere with NSX. The most common deployment topology is the [NAT topology](#). Decide which deployment topology you will implement, and plan accordingly.

## Plan Network CIDRs

Before you install Tanzu Kubernetes Grid Integrated Edition on VMware vSphere with NSX, you should plan for the CIDRs and IP blocks that you are using in your deployment.

Plan for the following network CIDRs in the IPv4 address space according to the instructions in [VMware NSX documentation](#).

- **VTEP CIDRs:** One or more of these networks host your GENEVE Tunnel Endpoints on your NSX Transport Nodes. Size the networks to support all of your expected Host and Edge

Transport Nodes. For example, a CIDR of `192.168.1.0/24` provides 254 usable IPs.

- **TKGI MANAGEMENT CIDR:** This small network is used to access Tanzu Kubernetes Grid Integrated Edition management components such as Ops Manager, BOSH Director, and Tanzu Kubernetes Grid Integrated Edition VMs as well as the Harbor Registry VM if deployed. For example, a CIDR of `10.172.1.0/28` provides 14 usable IPs. For the [No-NAT deployment topologies](#), this is a corporate routable subnet /28. For the [NAT deployment topology](#), this is a non-routable subnet /28, and DNAT needs to be configured in NSX to access the Tanzu Kubernetes Grid Integrated Edition management components.
- **TKGI LB CIDR:** This network provides your load balancing address space for each Kubernetes cluster created by Tanzu Kubernetes Grid Integrated Edition. The network also provides IP addresses for Kubernetes API access and Kubernetes exposed services. For example, `10.172.2.0/24` provides 256 usable IPs. This network is used when creating the `ip-pool-vips` described in [Creating VMware NSX Objects for Tanzu Kubernetes Grid Integrated Edition](#), or when the services are deployed. You enter this network in the **Floating IP Pool ID** field in the **Networking** pane of the Tanzu Kubernetes Grid Integrated Edition tile.

## Plan IP Blocks

When you install Tanzu Kubernetes Grid Integrated Edition on VMware NSX, you are required to specify the **Pods IP Block ID** and **Nodes IP Block ID** in the **Networking** pane of the Tanzu Kubernetes Grid Integrated Edition tile.

**Pods IP Block ID** and **Nodes IP Block ID** IDs map to the two IP blocks you must configure in VMware NSX: the Pods IP Block for Kubernetes pods, and the Node IP Block for Kubernetes nodes (VMs).

To configure **Pods IP Block ID** and **Nodes IP Block ID**:

- [Plan IP Blocks](#)
- [Pods IP Block](#)
- [Nodes IP Block](#)
- [Reserved IP Blocks](#)

For more information, see the [Networking](#) section of *Installing Tanzu Kubernetes Grid Integrated Edition on VMware vSphere with NSX Integration*.

---

NAT mode

Pods IP Block ID \*

78384e39-6bc6-4cc0-a8e2-8d70b727003f

Nodes IP Block ID \*

ad51f33b-e7ae-45f5-81dd-fd481177f1dc

 Enter the UUID of the IP Block to be used for Kubernetes Nodes

## Pods IP Block

Each time a Kubernetes namespace is created, a subnet from the **Pods IP Block** is allocated. The subnet size carved out from this block is /24, which means a maximum of 256 pods can be created per namespace.

When a Kubernetes cluster is deployed by Tanzu Kubernetes Grid Integrated Edition, by default 3 namespaces are created. Often additional namespaces will be created by operators to facilitate cluster use. As a result, when creating the **Pods IP Block**, you must use a CIDR range larger than /24 to ensure that NSX has enough IP addresses to allocate for all pods. The recommended size is /16. For more information, see [Creating VMware NSX Objects for Tanzu Kubernetes Grid Integrated Edition](#).



**Note:** By default, **Pods IP Block** is a block of non-routable, private IP addresses. After you deploy Tanzu Kubernetes Grid Integrated Edition, you can define a network profile that specifies a routable IP block for your pods. The routable IP block overrides the default non-routable **Pods IP Block** when a Kubernetes cluster is deployed using that network profile. For more information, see [Routable Pods in Using Network Profiles \(VMware NSX Only\)](#).

## ip-block-pks-pods-snat

**Overview** Subnets

▼ Summary | [EDIT](#)

|              |                                              |
|--------------|----------------------------------------------|
| Name         | ip-block-pks-pods-snat                       |
| ID           | 78384e39-6bc6-4cc0-a8e2-8d70b727003f         |
| Description  |                                              |
| CIDR         | 172.16.0.0/16                                |
| Created      | 5/11/2018, 2:12:50 PM by admin               |
| Last Updated | 7/16/2018, 8:43:42 AM by pks-nsx-t-superuser |

➤ Tags | [MANAGE](#)

## Nodes IP Block

Each Kubernetes cluster deployed by Tanzu Kubernetes Grid Integrated Edition owns a /24 subnet.

To deploy multiple Kubernetes clusters, set the **Nodes IP Block ID** in the **Networking** pane of the Tanzu Kubernetes Grid Integrated Edition tile to larger than /24. The recommended size is /16. For more information, see [Creating VMware NSX Objects for Tanzu Kubernetes Grid Integrated Edition](#).



**Note:** You can use a smaller nodes block size for no-NAT environments with a limited number of routable subnets. For example, /20 allows up to 16 Kubernetes clusters to be created.

## ip-block-pks-nodes-snat

- [Overview](#)
- [Subnets](#)

---

▼ Summary
|
[EDIT](#)

|              |                                              |
|--------------|----------------------------------------------|
| Name         | ip-block-pks-nodes-snat                      |
| ID           | ad51f33b-e7ae-45f5-81dd-fd481177f1dc         |
| Description  |                                              |
| CIDR         | 172.15.0.0/16                                |
| Created      | 5/21/2018, 11:53:50 AM by admin              |
| Last Updated | 7/16/2018, 8:43:32 AM by pks-nsx-t-superuser |

> Tags
|
[MANAGE](#)

## Reserved IP Blocks

TKGI reserves several CIDR blocks and IP addresses for internal use. When deploying TKGI, do not use a reserved IP address or CIDR block.



**Note: Do not use reserved IP addresses or CIDR blocks when configuring TKGI.**

| Reserv<br>ed<br><br>Addres<br>s/Rang<br>e | Compo<br>nent                    | Customi<br>zable   | Description                                                                                                                                                                                                                                                                                                                                                                        |
|-------------------------------------------|----------------------------------|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 10.100<br>.200.0<br>/24                   | Nodes                            | No.<br>IP<br>Block | Each Kubernetes cluster uses the 10.100.200.0/24 subnet for Kubernetes services.<br>Do not use this IP range for the Nodes IP Block.                                                                                                                                                                                                                                               |
| 172.17<br>.0.0/1<br>6                     | Worker<br>node<br>VM             | No.                | containerd is installed on each Tanzu Kubernetes Grid Integrated Edition worker node and is assigned the 172.17.0.0/16 network interface.<br>Do not use this CIDR range for any TKGI component, including Ops Manager, BOSH Director, the TKGI API VM, the TKGI DB VM, and the Harbor Registry VM. Note: This range is also reserved for the Management Console VM, but is unused. |
| 172.17<br>.0.0/1<br>6                     | Manage<br>ment<br>Consol<br>e VM | No.                | The TKGI Management Console VM also reserves an unused docker0 interface on 172.17.0.0/16. This cannot be customized.                                                                                                                                                                                                                                                              |

|               |                       |                                    |                                                                                                                                                                                                                              |
|---------------|-----------------------|------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 172.18.0.0/16 | Management Console VM | Yes. See OVA configuration.        | The Tanzu Kubernetes Grid Integrated Edition Management Console runs the Docker daemon and reserves 172.18.0.0/16 for the subnet. Do not use this CIDR range unless you customize them during OVA configuration.             |
| 172.18.0.1    | Management Console VM | Yes. See OVA configuration.        | The Tanzu Kubernetes Grid Integrated Edition Management Console runs the Docker daemon and reserves 172.18.0.1 for the gateway. Do not use this CIDR range or IP address unless you customize them during OVA configuration. |
| 172.20.0.0/16 | Harbor Registry VM    | Yes. See Harbor tile > Networking. | The Harbor Registry requires IP blocks in the range 172.20.0.0/16. Do not use this CIDR range, unless you change it in the <a href="#">Harbor tile configuration</a> .                                                       |

## Gather Other Required IP Addresses

To install Tanzu Kubernetes Grid Integrated Edition on vSphere with VMware NSX, you will need to know the following:

- Subnet name where you will install Tanzu Kubernetes Grid Integrated Edition
- VLAN ID for the subnet
- CIDR for the subnet
- Netmask for the subnet
- Gateway for the subnet
- DNS server for the subnet
- NTP server for the subnet
- IP address and CIDR you plan to use for the VMware NSX Tier-0 Router

## Considerations for Using the VMware NSX Policy API with TKGI

This topic provides considerations for using the NSX Policy API with Tanzu Kubernetes Grid Integrated Edition (TKGI) on vSphere.



**Warning:** The NSX Policy API feature is available at 50% of NSX Management Plane API scale. For detailed scale numbers, see [NSX 4.0.1 Configuration Limits](#).

## NSX Policy API Support

The NSX Policy API is the next-generation interface for integrating with the NSX networking and security framework.

In addition to supporting the NSX Management API, TKGI supports using the NSX Policy API to deploy Tanzu Kubernetes Grid Integrated Edition on vSphere.

If you are planning on using the NSX Policy API, keep in mind that only new deployments of TKGI are supported. You cannot configure an existing installation of TKGI to use the NSX Policy API.

In addition, while all TKGI functionality is supported in both NSX modes, Policy and Management, there are some differences to be aware of when configuring NSX objects for TKGI, and when configuring the BOSH and TKGI tiles. These differences are described in more detail below.

## NSX Versions

To use the NSX Policy API with your TKGI installation, you must use a supported NSX version. Refer to the [Release Notes](#).

## NSX Deployment Topologies

Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX supports several [deployment topologies](#).

Currently Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX Policy API supports all network topologies except the [VSS/VDS topology](#).

## NSX Installation

To use the NSX Policy API, there are no changes required to the installation of the main NSX components, including NSX Manager and Edge Nodes.

For installation instructions, see [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX Data Center](#).

## NSX Objects for Kubernetes Clusters

To use the NSX Policy API, you must configure the required NSX control plane objects using the NSX Policy API or UI. Specifically, you must configure the Tier-0 Router (called Gateway in the Policy terminology), the Nodes IP Block, the Pods IP Block, and the Floating IP Pool need to be created using the Policy API or UI.

For specific instructions on creating the required objects, see [Create the NSX Objects for Kubernetes Clusters Provisioned by TKGI](#).

## TKGI Configuration

When you configure the BOSH Director tile for Tanzu Kubernetes Grid Integrated Edition, you must enable the option vCenter Config > NSX Networking > **Use NSX Policy API**. See [Configure NSX Networking](#).

Also, when you configure the TKGI tile in Ops Manager, you must enabled Settings > Networking > NSX > **Policy API mode**. See [Configure TKGI Networking](#).

## Management Console

If you are using the TKGI Management Console, you need to select the Policy API in the TKGI configuration section.

## Network Profile

Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX supports the use of [Network Profile](#) for modifying specific NSX settings post-installation. A limited number of network profile use cases are not supported when using TKGI with the NSX Policy API.

The Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX Policy API does not support either the “Top Firewall” or the “Bottom Firewall” DFW Section Markers. For more information, see [DFW Section Markers](#).

The Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX Policy API does not support [NSGroups](#) if you create the group in a domain other than the default. With the Policy API, a group must be part of a domain. The `default` domain is supported, and if you create the group using the NSX Policy interface, the group is automatically put in the `default` domain. However, if you use the Policy REST API to create a group in a domain other than the default, it is not supported.

## Migrating the NSX Management Plane API to NSX Policy API - Overview

This topic provides an overview of migrating VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) from the NSXManagement Plane API to NSX Policy API.

To migrate a TKGI from NSX Management Plane API to NSX Policy API, see [Migrating the NSX Management Plane API to NSX Policy API](#).



**Warning:** The NSX Policy API feature is available at 50% of NSX Management Plane API scale. For detailed scale numbers, see [NSX 4.0.1 Configuration Limits](#).

## Overview

The NSX Management Plane API has been deprecated. VMware recommends that you instead use the NSX Policy API. For more information about the deprecation of the NSX Management Plane API, see [Deprecation announcement for NSX Manager APIs and NSX Advanced UIs in VMware NSX-T Data Center 3.2 Release Notes](#).

The NSX Management Plane API to NSX Policy API (MP2P) Migration feature switches TKGI environments using the NSX Management Plane API to the NSX Policy API.



**Note:** TKGI supports NSX Management Plane API to NSX Policy API Migration only. You cannot return TKGI to the NSX Management Plane API after starting TKGI MP2P Migration.

## NSX Management Plane API to NSX Policy API Migration Features

TKGI NSX Management Plane API to NSX Policy API Migration provides the following advantages over manual migration:

- Supports TKGI MP2P Migration in most environments.
- Reduces a complex manual workflow to a few procedures:
  - Automates most NSX Manager and TKGI configuration steps.
  - Migrates cluster NSX resources to NSX Policy API objects without disrupting NSX data path.
  - Migrates all cluster deployment NSX resources to Policy IDs, including Network Profile configurations and BOSH deployment properties.
- Reduces the impact on the workloads in your environment:
  - By promoting clusters to NSX Policy API one by one, the MP2P Migration procedure affects only the cluster being promoted. Other clusters continue to operate as usual.
- Supports migrating common topologies.

For more information, see:

- [TKGI MP2P Migration Configurations](#)
- [TKGI MP2P Migration Operational Limitations](#)

## TKGI MP2P Migration Configurations

Before initiating TKGI MP2P, consider the following supported and unsupported configurations:

- [Supported Configurations](#)
- [Unsupported Configurations](#)

## Supported Configurations

TKGI MP2P Migration supports migrating TKGI in only NSX environments dedicated to TKGI. Do not start TKGI MP2P Migration if TKGI shares your NSX environment with other products, for example, Tanzu Application Service or VMware Aria Automation.

TKGI MP2P Migration supports the following topologies:

- All deployment topologies defined in [NSX-T Deployment Topologies for TKGI](#) except vSAN stretched cluster topology.
- All Tier-1 Router topologies defined in [Shared and Dedicated Tier-1 Router Topologies](#).
- All Multi-Tier-0 topologies defined in [Isolating Tenants](#).

Contact VMware Support before initiating MP2P Migration if your TKGI environment uses a customized topology or is a multi-foundation deployment of TKGI.

## Unsupported Configurations

The following are not supported by TKGI MP2P Migration or TKGI using the NSX Policy API:

- Do not start TKGI MP2P Migration if TKGI shares your NSX environment with other products, for example, Tanzu Application Service or VMware Aria Automation.
- TKGI MP2P Migration does not support clusters configured with NSGroups, including [Bootstrap Security Group](#) and [BOSH VM Extensions](#) NSGroup configurations.
- The TKGI Management Console can not be used to manage TKGI after TKGI MP2P Migration.

## TKGI MP2P Migration Operational Limitations

Review the following and resolve concerns before initiating TKGI MP2P Migration:

- [Before MP2P Migration](#)
- [During MP2P Migration](#)
- [During Cluster Promotion to NSX Policy API](#)



**Note:** To reduce the possibility of unexpected issues, VMware recommends that you minimize the duration of your MP2P Migration mix-mode maintenance window by promoting all clusters as soon as possible.

## Before MP2P Migration

Consider the following before initiating TKGI MP2P Migration:

- Evaluate the complexity of your TKGI environment. PSO assistance might be required to migrate some complex environments.
- You cannot rotate cluster certificates during TKGI MP2P Migration. Verify your cluster cluster certificates, including the NSX-T certificate, are valid for the entire duration of your TKGI MP2P Migration before starting MP2P Migration.
- When planning your TKGI MP2P Migration, include redefining DFW rules in your cluster promotion procedure. For more information, see [Dealing with DFW Sections Created by NSX Admin](#) in the VMware NSX Container Plugin documentation.

## During MP2P Migration

During your MP2P Migration mixed-mode maintenance window:

- You cannot roll back your TKGI environment to the NSX Management Plane API.
- Migrate clusters serially. Do not attempt to migrate multiple clusters in parallel. NSX does not support multiple simultaneous MP2P Migration requests.

VMware recommends that you migrate your entire TKGI environment to NSX Policy API as quickly as possible to avoid the extra complexity of an NSX Management Plane/Policy API mixed-mode environment.



**Warning:** Limit upgrading NSX and TKGI to only resolving critical issues while your environment is in MP2P Migration mixed-mode.

## DFW Migration

Policy API supports new firewall sections. The new firewall sections have a higher priority than existing Management Plane-based firewall rules, including existing top firewall rules.

The DFW migration procedure described in [Migrating the NSX Management Plane API to NSX Policy API](#) documents the recommended two-step MP2P firewall migration procedure for typical DFW firewall configurations:

- Re-create the top firewall rules before promoting clusters.
- Re-create the bottom firewall rules immediately after promoting clusters.

Adhering to the recommended DFW migration sequence is critical to maintaining security and cluster workload network connectivity. If a TKGI cluster is:

- Promoted before existing Management Plane API top firewall rules have been recreated in the Policy API section: Cluster network policies will be enforced before the top firewall rules.
- Promoted after existing Management Plane API bottom firewall rules have been recreated in the Policy API section: The top firewall rules will be enforced before Kubernetes network policies and might override them.

An example result of failing to migrate DFW rules correctly is clusters with access to CIDRs that had been intended to be globally blocked.



**Warning:** If you do not configure your DFW Rules correctly, cluster workloads will lose network connectivity.

For information about migrating DFW rules for specific edge case configurations, see [Dealing with DFW Sections Created by NSX Admin](#) in the VMware NSX Container Plugin documentation.

## During Cluster Promotion to NSX Policy API

Promoting a cluster to NSX Policy API migrates the cluster's NSX resources and updates the cluster. Promote your clusters to the NSX Policy API serially only while the MP2P Migration mixed-mode maintenance window is active. If a cluster promotion fails, the only affected cluster is the one being promoted.

While promoting a cluster to NSX Policy API:

- Do not manage the cluster.
- Do not update or delete any of the cluster's existing workloads.
- Do not create new workloads on the cluster.

Workloads run as usual while promoting the cluster.

Clusters that are not actively being promoted:

- Can be managed as usual.
- Workloads on these clusters continue to run as usual and can be managed as usual.

The amount of time it takes to promote a cluster depends on the scale of resources NSX needs to migrate the cluster, and the time it takes to update the cluster.

## Migrating the NSX Management Plane API to NSX Policy API

This topic describes how to migrate VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) from NSX Management Plane API to NSX Policy API.

For an overview of NSX Management Plane API to NSX Policy API Migration, see [Migrating the NSX Management Plane API to NSX Policy API - Overview](#).



**Warning:** The NSX Policy API feature is available at 50% of NSX Management Plane API scale. For detailed scale numbers, see [NSX 4.0.1 Configuration Limits](#).

## Overview

The NSX Management Plane API to NSX Policy API (MP2P) Migration feature switches environments using the NSX Management Plane API TKGI to the NSX Policy API.

VMware recommends that your MP2P Migration adheres to the following Prepare -> Migrate -> Clean Up workflow:

| Procedure                           | Description                                                                                                                                                                                                                           | Outcomes                                                                                                                                                          |
|-------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Pr ep ar e</b>                   | <b>Enable Migration</b><br>Configure your environment for MP2P Migration in TKGI, NSX, and BOSH.                                                                                                                                      |                                                                                                                                                                   |
|                                     | <b>Re-Create Top DFW Firewall Rules</b><br>Re-create your high-priority user-defined DFW rules.                                                                                                                                       |                                                                                                                                                                   |
| <b>Acti vate Policy API in TKGI</b> | To create and configure NSX Policy API objects: <ul style="list-style-type: none"> <li>• Create and promote a single test cluster.</li> <li>• Configure the TKGI tile with the IDs for the created NSX Policy API objects.</li> </ul> | Migrates shared resources, including Tier-0, FIP block, Pod IP block, and Node IP block to NSX Policy API.<br><br>Newly created clusters will use NSX Policy API. |

|                         |                                                               |                                                                                                                                                                                     |                                                                                                                                                                                                                                                                      |
|-------------------------|---------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Mi<br>gr<br>at<br>e     | <b>Migrate a Cluster</b>                                      | Steps: <ul style="list-style-type: none"><li>Backup your cluster's critical workloads.</li><li>Promote the cluster.</li><li>Verify the migration of the cluster.</li></ul>          | Use <code>tkgi promote-cluster-to-policy</code> to promote the cluster.                                                                                                                                                                                              |
|                         | <b>Migrate All Production Clusters</b>                        | Repeat the <b>Migrate a Cluster</b> procedure above for each production cluster.                                                                                                    | Notes: <ul style="list-style-type: none"><li>Only promote clusters one at a time.</li><li>Always verify the migration of your cluster before migrating another cluster.</li></ul>                                                                                    |
|                         | <b>Re-Create Bottom DFW Firewall Rules</b>                    | Steps: <ul style="list-style-type: none"><li>Re-create your low-priority user-defined DFW rules.</li><li>Remove original Management Plane section user-defined DFW rules.</li></ul> |                                                                                                                                                                                                                                                                      |
| Cl<br>ea<br>n<br>U<br>p | <b>Remove NSX Management Plane API-related configurations</b> | Clean up: <ul style="list-style-type: none"><li>Clean up Network Profiles.</li></ul>                                                                                                |                                                                                                                                                                                                                                                                      |
|                         | <b>Promote Remaining NSX Objects</b>                          | Use the NSX Promoter to convert the NSX Objects that are configured outside of TKGI.                                                                                                | Notes: <ul style="list-style-type: none"><li>The NSX Promoter updates all objects in the environment to NSX Policy API.</li><li>Use to promote the Deployment Router, and IP blocks, etc.</li><li>Use with caution if TKGI shares NSX with other products.</li></ul> |
|                         | <b>Migrate BOSH to NSX Policy API mode</b>                    | Deactivate BOSH migration mode to support NSX Policy API clusters only.                                                                                                             |                                                                                                                                                                                                                                                                      |
|                         | <b>Verification</b>                                           | Verify the MP2P Migration.                                                                                                                                                          |                                                                                                                                                                                                                                                                      |

To complete an MP2P Migration in your TKGI environment:

- Before migrating your TKGI environment to NSX Policy API, review the warnings and considerations in [Migrating the NSX Management Plane API to NSX Policy API - Overview](#).
- Confirm your environment meets the [Prerequisites](#).
- [Prepare for MP2P Migration](#).
- [Migrate TKGI Clusters from the NSX Management Plane API to NSX Policy API](#).

## 5. Post-Migration Cleanup.



**Warning:** Limit upgrading NSX and TKGI to only resolving critical issues while your environment is in MP2P Migration mixed-mode.

## Prerequisites

Before migrating TKGI from NSX Management Plane API to NSX Policy API, verify your TKGI environment meets the following requirements:

- TKGI Version Requirements:
  - TKGI Tile and CLI: v1.16.0 or later.
  - TKGI clusters: v1.16.0 or later.
- Environment Version Requirements:
  - NSX: v4.0.1.1 or later.
  - NSX environment is a dedicated TKGI environment. For example, an environment without a TAS or other installations in production.
  - Ops Manager: v2.10.45 or later or v3.0.0 or later.
  - Ops Manager CLI: latest.
- Cluster Requirements:
  - TKGI MP2P Migration does not support clusters configured with:
    - NSGroups, including [Bootstrap Security Group](#) and [BOSH VM Extensions](#) NSGroup configurations.



**Note:** MP2P Migration supports NSGroups when migrating on NSX v4.1.0 and later.

- The `ncp.nsx_v3.k8s_np_use_ip_sets` Network Profile parameter set to `false`. Before migrating a cluster, you must restore the parameter to `true`, the default value for NSX Management Plane API. For more information, see [Migrate a TKGI Cluster to NSX Policy API](#) below.
- NSX Policy API Mode resources, such as the `default-balanced-client-ssl-profile`, `default-high-compatibility-client-ssl-profile` or `default-high-security-client-ssl-profile` default ssl profiles. MP2P Migration supports only clusters that reference only NSX Management Plane API resources.
- Other Requirements:
  - Administrator access to NSX, Ops Manager, BOSH, and TKGI.
  - For more information about TKGI MP2P Migration limitations, see [TKGI MP2P Migration Configurations](#) and [TKGI MP2P Migration Operational Limitations](#) in

*Migrating the NSX Management Plane API to NSX Policy API - Overview.*

## Prepare for MP2P Migration

To prepare your TKGI environment for MP2P Migration:

- [Enable Migration](#)
- [Migrate DFW Top Firewall Rules](#)
- [Activate NSX Policy API in TKGI](#)
- [Configure the Environment for NSX Policy API](#)

## Enable Migration

You must enable support for MP2P Migration in NSX before promoting clusters to NSX Policy API. Additionally, Ops Manager and BOSH must be configured to support a mixed environment of NSX Management Plane API and NSX Policy API clusters before promoting clusters.



**Note:** After activating NSX Policy API, existing NSX backups created while using NSX Management Plane API cannot be used to restore your environment or your clusters.

To prepare TKGI for MP2P Migration:

1. Disable the TKGI Upgrade All Clusters errand.

To prepare NSX for MP2P Migration:

1. To activate the NSX Migration Coordinator Service on all NSX managers:

1. SSH to the NSX Manager with administrative privileges:
  2. At the `nsxmanager>` prompt, run the following:

```
nsxmanager> start service migration-coordinator
```

2. If your NSX Manager cluster is configured with VIP, configure the Source IP Persistence Profile for LB or use a Source IP LB algorithm during TKGI foundation migration. All migration requests should be made on a single NSX Manager.

To prepare Ops Manager and BOSH for MP2P Migration:

- If you are using Ops Manager v3.0.0 or later:
  1. Open your Ops Manager BOSH Director for vSphere tile to the **Director Config** pane.

2. Activate **Use NSX-T Policy API Migration Mode**.



**Note:** Do not activate **Use NSX-T Policy API** at this time.

3. Click **Save**.
4. On the Ops Manager **Installation Dashboard**, select **Review Pending Changes**, review the changes, and select **Apply Changes**.
- If you are using Ops Manager v2.10.45 or later:
  1. Download [bosh\\_migration\\_mode.sh](#), the BOSH Migration Mode script, from the VMware Tanzu Kubernetes Grid Integrated Edition documentation GitHub repository.
  2. Select a virtual machine that is able to reach Ops Manager, and can run the [Ops Manager CLI](#), [BOSH CLI](#), and [yq CLI](#).
  3. Copy the the BOSH Migration Mode script to the virtual machine.
  4. Export the `BOSH_CLIENT`, `BOSH_CLIENT_SECRET`, `BOSH_ENVIRONMENT`, and `BOSH_CA_CERT` environment variables. For more information, see [Set the BOSH Environment Variables on the Ops Manager VM](#) in *Advanced Troubleshooting with the BOSH CLI*.

For example:

```
export BOSH_CLIENT=ops_manager BOSH_CLIENT_SECRET=FHoIRmt3qq1L
fbPncF4vAyxZWUSpqbZ- BOSH_ENVIRONMENT=88.0.0.3 bosh BOSH_CA_CERT=
RT=/var/tempest/workspaces/default/root_ca_certificate
```

5. To activate BOSH Migration Mode run the BOSH Migration Mode script:

```
./bosh_migration_mode enable OPSMAN-IP USERNAME PASSWORD
```

Where:

- `OPSMAN-IP` is the IP address for the Ops Manager.
- `USERNAME` is the account to use to run Ops Manager API commands.
- `PASSWORD` is the password for the account.

## Migrate DFW Top Firewall Rules

If your clusters are customized with DFW rules or use Network Profiles configured with DFW section markers, migrate the DFW rules:

1. Re-create your existing top section DFW rules from above your NSX Manager `top_firewall_section_marker` in the NSX Policy Environment section. Include the `top_firewall_section_marker` in the new copy.



**Warning:** If you do not configure your DFW Rules correctly, your workloads will lose network connectivity. Both the original copy of the

`top_firewall_section_marker` section and the re-created copy in the NSX Policy Environment section must be in your DFW rules after you complete this step.

For more information on configuring DFW rules, see [DFW Migration](#) in *Migrating the NSX Management Plane API to NSX Policy API - Overview*.

## Activate NSX Policy API in TKGI

To migrate NSX Management Plane API shared network resources to NSX Policy API objects:

1. Create a simple test cluster.
2. Migrate the cluster to the NSX Policy API:

```
pks promote-cluster-to-policy CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of the promoted cluster.

3. Verify successful cluster migration to the NSX Policy API:

```
pks cluster CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of the promoted cluster.

To collect the newly created Resource Policy IDs:

1. Retrieve the existing NSX Management Plane API IDs:
  1. Open the TKGI tile UI.
  2. Open the **Networking** tab.
  3. Retain the following NSX Management Plane API IDs:
    - **Pods IP Block ID**
    - **Nodes IP Block ID**
    - **T0 Router ID**
    - **Floating IP Pool ID**

For more information on **Networking** tab settings, see [Networking](#) in *Installing TKGI on vSphere with VMware NSX*.

2. Retrieve the new Resource Policy IDs:
  1. Log in to the NSX Web UI using an account with admin privileges.
  2. Go to the homepage.
  3. Enter one of the NSX Management Plane API IDs retrieved in the last step into the search bar and search.
  4. Retrieve the corresponding Resource Policy ID From the **ID** field. Note that the indicated **Creation Time** should match the time when you promoted your test cluster.

5. Retrieve and retain the Resource Policy ID for each NSX Management Plane API ID you collected in the last step.

## Configure the Environment for NSX Policy API

To reconfigure TKGI with Resource Policy IDs:

1. Open the TKGI tile UI.
2. Open the **Networking** tab.
3. Activate **Policy API mode**.
4. Replace the Management Plane API IDs with the retained Resource Policy IDs:
  - ◊ **Pods IP Block ID**
  - ◊ **Nodes IP Block ID**
  - ◊ **T0 Router ID**
  - ◊ **Floating IP Pool ID**

For more information on **Networking** tab settings, see [Networking in \*Installing TKGI on vSphere with VMware NSX\*](#).

5. Select **Apply Changes**.



**Note:** After configuring the TKGI tile, newly created clusters use the NSX Policy API.

## Migrate TKGI Clusters from the NSX Management Plane API to NSX Policy API

To migrate TKGI to NSX Policy API, serially promote all of your TKGI clusters individually:

1. [Migrate a TKGI Cluster to NSX Policy API](#)

After migrating all clusters:

1. [Migrate DFW Bottom Firewall Rules](#)



**Note:** Do not intentionally run TKGI in mixed mode for an extended period of time. Promote all TKGI clusters to NSX Policy API as quickly as possible.

## Migrate a TKGI Cluster to NSX Policy API

To migrate an individual cluster from the NSX Management Plane API to NSX Policy API:

1. If your cluster is custom configured with the `ncp.nsx_v3.k8s_np_use_ip_sets` Network

Profile parameter set to `false`, restore the default "`ncp.nsx_v3.k8s_np_use_ip_sets": true`" configuration by updating the cluster with a modified Network Profile.

- Promote the cluster using the TKGI CLI:

```
tkgi promote-cluster-to-policy CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of the promoted cluster.



**Note:** Do not attempt to promote an additional TKGI cluster to NSX Policy API before completing the promotion of the current clusters.

- Validate successful migration to NSX Policy API topology for the cluster before promoting subsequent clusters:

- Run the following:

```
pks cluster CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of the promoted cluster.

- Confirm successful cluster promotion by reviewing the migration log on the Kubernetes control plane VM.

For more information on cluster limitations while promoting a cluster, see [During Cluster Promotion to NSX Policy API](#) in *Migrating the NSX Management Plane API to NSX Policy API - Overview*.



**Note:** Do not attempt to promote an additional TKGI cluster to NSX Policy API before completing the promotion of the current clusters.

## Migrate DFW Bottom Firewall Rules

If your clusters are customized with DFW rules or use Network Profiles configured with DFW section markers, finish migrating the DFW rules:

- Re-create your existing bottom section DFW rules from below your NSX Manager `bottom_firewall_section_marker` in the NSX Policy Application section BELOW all migrated NCP rules. Include the `bottom_firewall_section_marker`.
- Remove all of the original NSX Management Plane customer-defined DFW rules. When done, there should be only one copy of the top firewall rules, the one in the NSX Policy Environment section, and only one copy of the bottom firewall rules, the one in the NSX Policy Application section.



**Warning:** If you do not configure your DFW Rules correctly, your workloads will lose network connectivity. You must remove all NSX Management Plane user-defined DFW rules before starting post-migration cleanup.

For more information on configuring DFW rules, see [DFW Migration](#) in *Migrating the NSX*

*Management Plane API to NSX Policy API - Overview.*

## Post-Migration Cleanup

After all TKGI clusters have been promoted to NSX Policy API, remaining NSX resources must be promoted and the TKGI configurations for NSX Management Plane API objects should be removed.

To clean up after promoting all clusters:

1. Remove NSX Management Plane API-related configurations:
  1. Remove the original NSX Management Plane API-configured Network Profiles.
2. Use the NSX Promoter to promote the remaining NSX resources:
  1. Log in to the NSX Web UI using an account with admin privileges.
  2. Go to the **NSX > System > General Settings**.
  3. Select **Start Objects Promotion**.

The following are reconfigured by the NSX Promoter:

- Deployment network and router.
- NSX resources in TKGI tile **Resource Config**.
- NSX resources in cluster `vm_extension` configurations.
- Custom infra-level resources out of TKGI scope. For example, NAT, IP allocations, subnet allocations, and the load balancers that you have created.

For information about the NSX Promoter, see [Promote Manager Objects to Policy Objects](#).

3. (Optional) To deactivate the NSX Migration Coordinator Service on all NSX managers:

1. SSH to the NSX Manager with administrative privileges.
2. At the `nsxmanager>` prompt, run the following:

```
nsxmanager> stop service migration-coordinator
```

4. If your NSX Manager cluster was configured with VIP before you started MP2P Migration, restore your VIP configuration.
5. To switch BOSH to Policy API mode:

- ◊ If you are using Ops Manager v3.0.0 or later:
  1. Open your Ops Manager BOSH Director for vSphere tile to the **Director Config** pane.
  2. Deactivate **Use NSX-T Policy API Migration Mode**.
  3. Activate **Use NSX-T Policy API**.
  4. Click **Save**.
  5. On the Ops Manager **Installation Dashboard**, select **Review Pending**

### Changes.

6. Ensure that **BOSH Director** is the only product selected.
  7. Select **Apply Changes**.
- ◊ If you are using Ops Manager v2.10.45 or later:
    1. Access the virtual machine where you ran the BOSH Migration Mode script when you prepared Ops Manager and BOSH for MP2P Migration.
    2. Export the `BOSH_CLIENT`, `BOSH_CLIENT_SECRET`, `BOSH_ENVIRONMENT`, and `BOSH_CA_CERT` environment variables. For more information, see [Set the BOSH Environment Variables on the Ops Manager VM in Advanced Troubleshooting with the BOSH CLI](#).

For example:

```
export BOSH_CLIENT=ops_manager BOSH_CLIENT_SECRET=FHoIRmt3qq1LfbPncF4vAyxZWUSpqbZ- BOSH_ENVIRONMENT=88.0.0.3 bosh
 BOSH_CA_CERT=/var/tempest/worksheets/default/root_ca_certificate
```

3. Run the following BOSH Migration Mode script command:

```
./bosh_migration_mode disable OPSMAN-IP USERNAME PASSWORD
```

Where:

- `OPSMAN-IP` is the IP address for the Ops Manager.
- `USERNAME` is the account to use to run Ops Manager API commands.
- `PASSWORD` is the password for the account.

## Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with VMware NSX

This topic lists the sections to follow when installing NSX-T Data Center on vSphere for use with VMware Tanzu Kubernetes Grid Integrated Edition.

### Install NSX-T on vSphere

To install NSX-T on vSphere for Tanzu Kubernetes Grid Integrated Edition, complete the following sections in the order listed:

- [NSX-T Installation Prerequisites](#)
- [Install and Configure the NSX-T Manager Hosts](#)
- [Generate and Register the NSX-T TLS Certificate and Private Key](#)
- [Create an IP Pool for VTEP](#)
- [Configure Transport Zones](#)

- Configure vSphere Networking for ESXi Hosts
- Deploy NSX-T Edge Nodes
- Deploy NSX-T Transport Nodes
- Create NSX-T Objects for Kubernetes Clusters Provisioned by TKGI
- Create NSX-T Objects for TKGI Management Plane Components
- Configure NSX-T Passwords

## Prerequisites for Installing and Configuring NSX-T Data Center v3 for TKGI

This topic provides instructions for installing and configuring NSX-T Data Center v3 VMware Tanzu Kubernetes Grid Integrated Edition on vSphere.

### Prerequisites for Installing NSX-T Data Center

To perform a new installation of NSX-T Data Center for Tanzu Kubernetes Grid Integrated Edition, complete the following steps in the order presented.

1. Read the [Release Notes](#) for the target TKGI version you are installing and verify NSX-T v3 support.
2. Read the topics in the [Preparing to Install Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T Data Center](#) section of the documentation.

## Installing and Configuring VMware NSX Managers

This topic provides instructions for installing and configuring NSX-T Managers on vSphere in a clustered arrangement for high-availability.

### Prerequisites

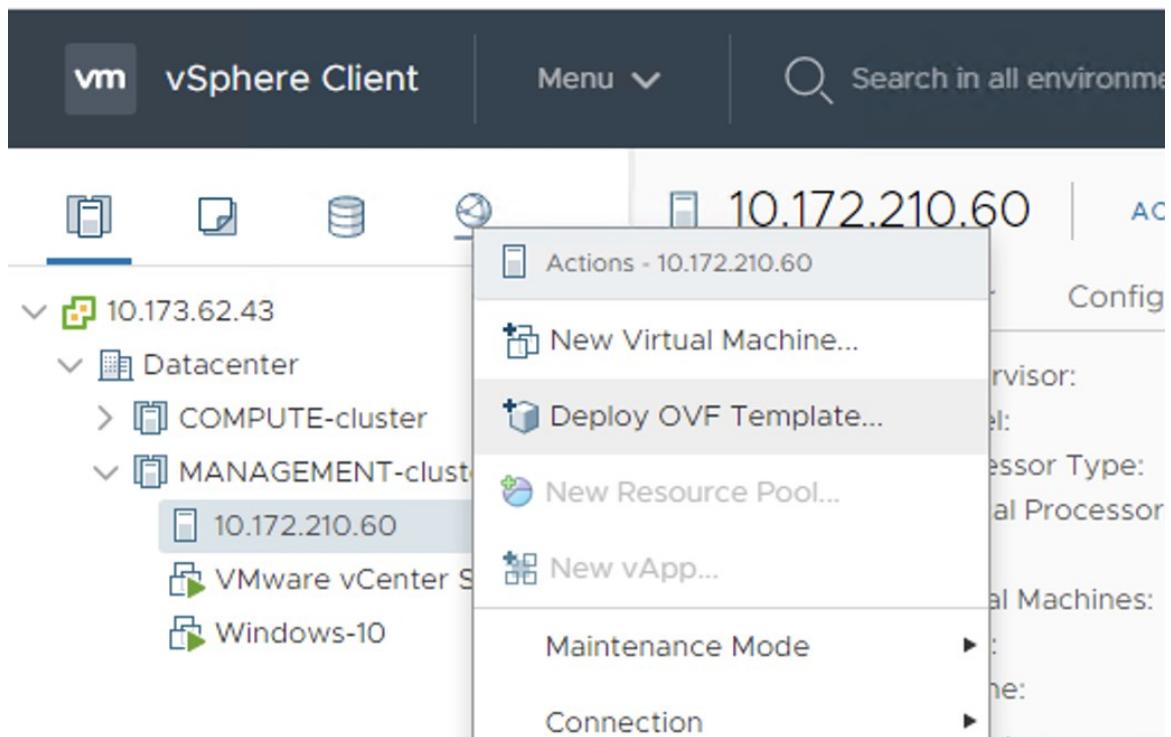
Before completing this section, make sure you have completed the following sections:

- [NSX-T Installation Prerequisites](#)

## Deploy NSX-T Manager 1

Deploy the NSX-T Manager OVA in vSphere. Download the OVA from the VMware software download site.

1. Using the vSphere Client, right-click the vCenter cluster and select **Deploy OVF Template**.



- At the **Select an OVF Template** screen, browse to and select the NSX Unified Appliance OVA file.

The screenshot shows the 'Deploy OVF Template' wizard, step 1: Select an OVF template. The steps are listed on the left:

- 1 Select an OVF template (selected)
- 2 Select a name and folder
- 3 Select a compute resource
- 4 Review details
- 5 Select storage
- 6 Ready to complete

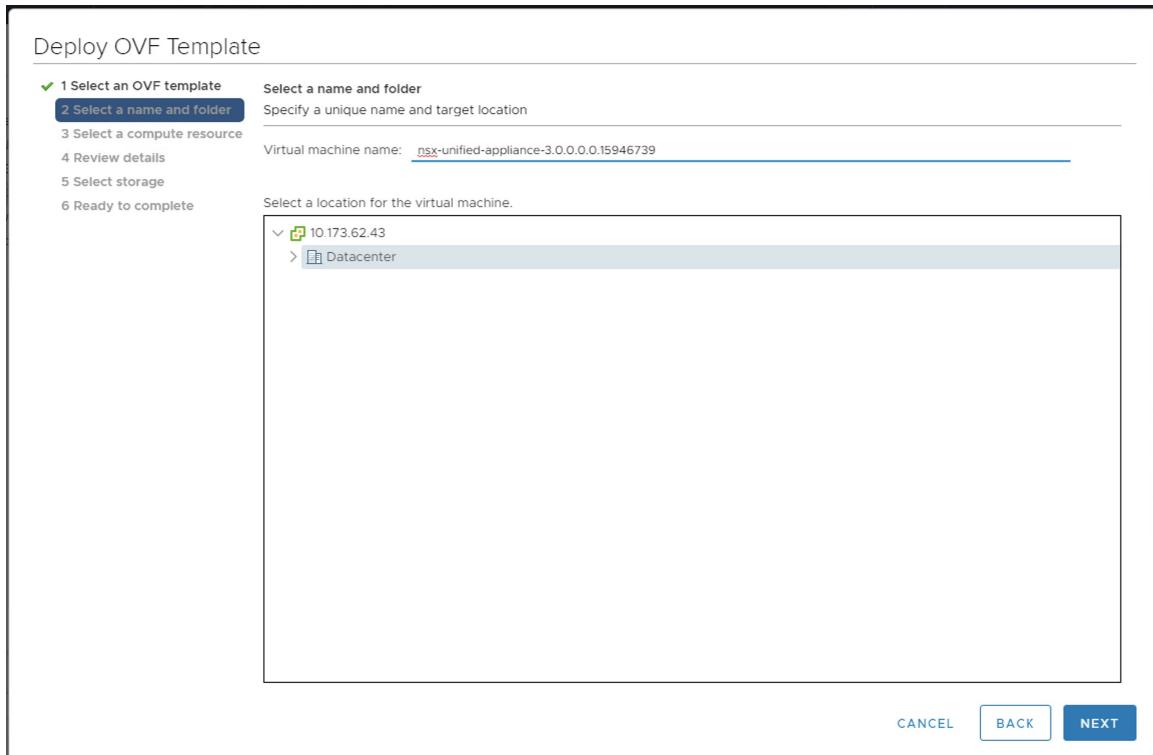
The main area has two sections:

- Select an OVF template**: A text input field with placeholder text 'Select an OVF template from remote URL or local file system'.
- Enter a URL to download and install the OVF package from the Internet, or browse to a location accessible from your computer, such as a local hard drive, a network share, or a CD/DVD drive.**: A text input field containing 'http://remote-server-address/filetodeploy.ovf'. Below it is a radio button group:
  - URL
  - Local file

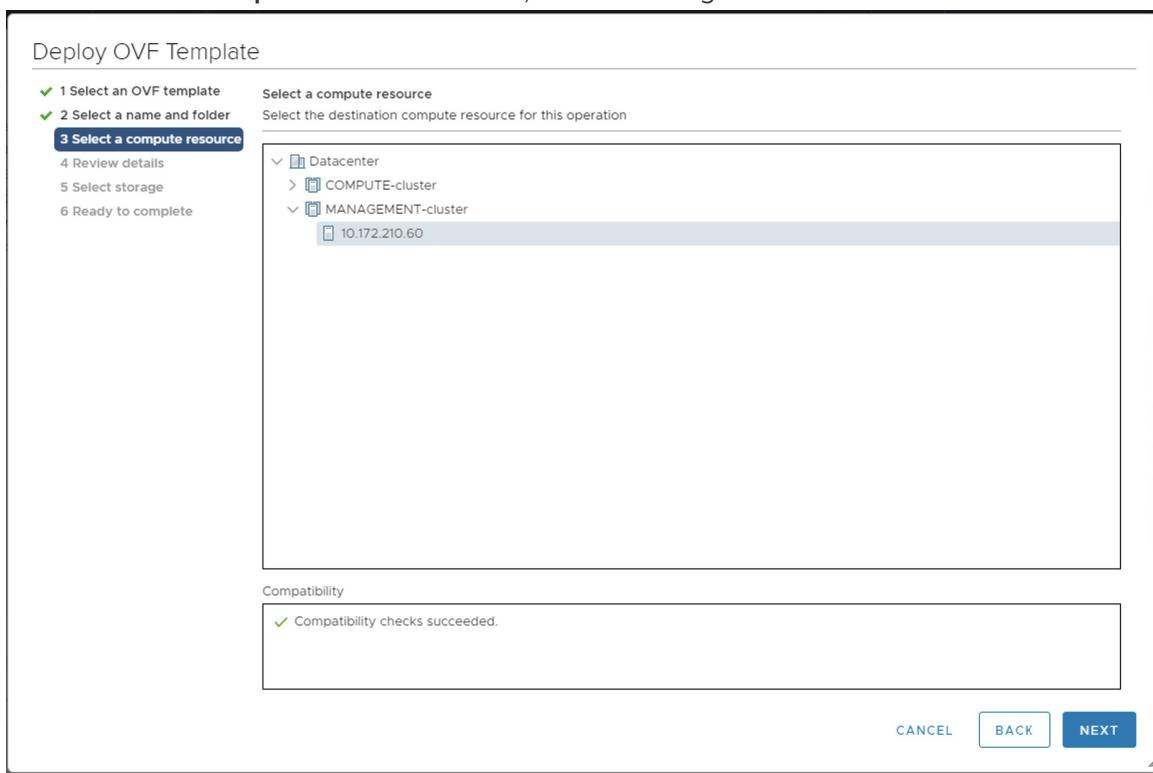
Below the input fields is a 'UPLOAD FILES' button followed by the file name 'nsx-unified-appliance-3.0.0.0.15946739.ova'.

At the bottom right are three buttons: CANCEL, BACK, and NEXT (highlighted).

- At the **Select a name and folder** screen, select the target Datacenter object.



- At the **Select a compute resource** screen, select the target vCenter cluster.



- Review the details.

Deploy OVF Template

**4 Review details**

- ✓ 1 Select an OVF template
- ✓ 2 Select a name and folder
- ✓ 3 Select a compute resource

The OVF package contains advanced configuration options, which might pose a security risk. Review the advanced configuration options below. Click next to accept the advanced configuration options.

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Publisher           | VMware, Inc. (Trusted certificate)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Product             | nsx-unified-appliance                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Version             | 3.0.0.0                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Vendor              | VMware, Inc                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Download size       | 11.0 GB                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Size on disk        | 5.6 GB (thin provisioned)<br>300.0 GB (thick provisioned)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Extra configuration | <pre>time.synchronize.tools.startup = false ethernet1.rxDataRingEnabled = 0 isolation.tools.vmxDnVersionGet.disable = true RemoteDisplay.maxConnections = 1 time.synchronize.restore = false time.synchronize.shrink = false isolation.tools.diskShrink.disable = true isolation.tools.memSchedFakeSampleStats.disable = true ethernet3.rxDataRingEnabled = 0 ethernet0.rxDataRingEnabled = 0 isolation.tools.guestDnDVersionSet.disable = true isolation.tools.unityActive.disable = true ethernet2.rxDataRingEnabled = 0 time.synchronize.continue = false time.synchronize.resume.disk = false isolation.tools.diskWiper.disable = true</pre> |

CANCEL BACK NEXT

6. At the **Configuration** screen, select at least **Medium** for the configuration size.

Deploy OVF Template

**5 Configuration**

- ✓ 1 Select an OVF template
- ✓ 2 Select a name and folder
- ✓ 3 Select a compute resource
- ✓ 4 Review details

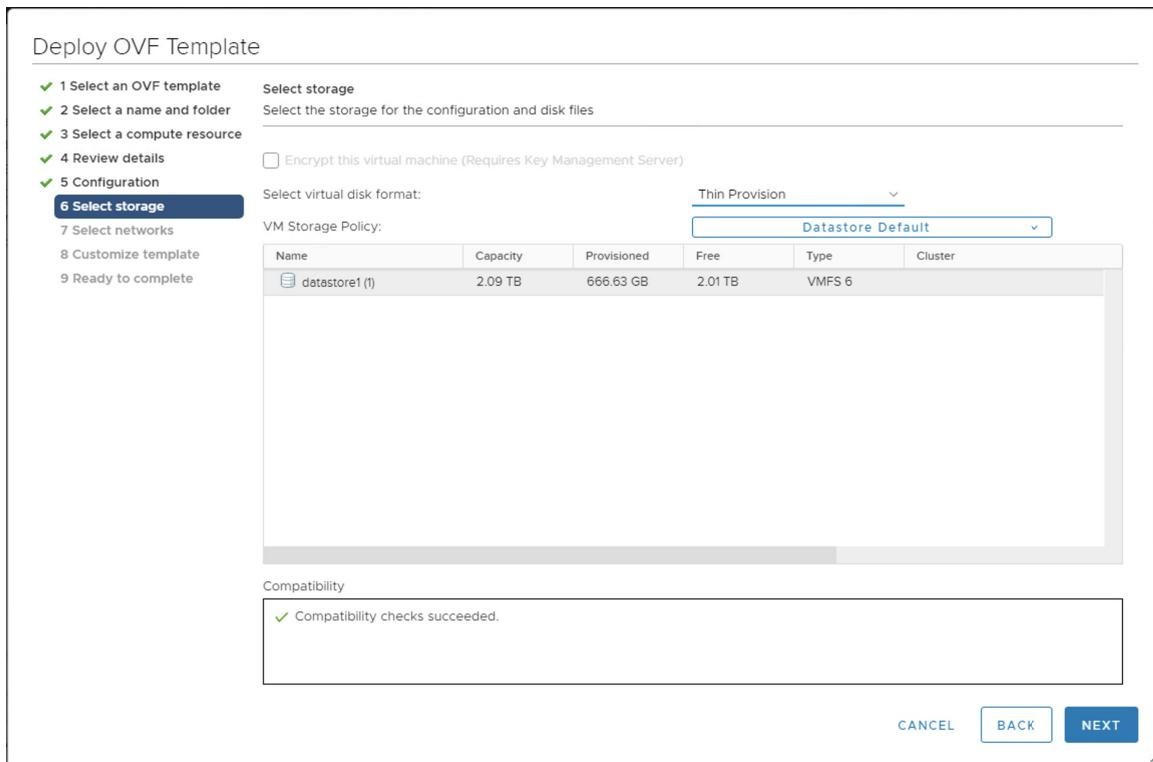
Configuration  
Select a deployment configuration

| Configuration                           | Description                                                                                                                                                                                                                                                                                                                         |
|-----------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <input type="radio"/> ExtraSmall        | IMPORTANT: This configuration is supported for Local Manager Production deployment ('NSX Manager' role). This is supported for Global Manager Production deployment (but not required). This configuration requires the following: * 6 vCPU * 24GB RAM * 300GB Storage * VM hardware version 10 or greater (vSphere 5.5 or greater) |
| <input type="radio"/> Small             |                                                                                                                                                                                                                                                                                                                                     |
| <input checked="" type="radio"/> Medium |                                                                                                                                                                                                                                                                                                                                     |
| <input type="radio"/> Large             |                                                                                                                                                                                                                                                                                                                                     |

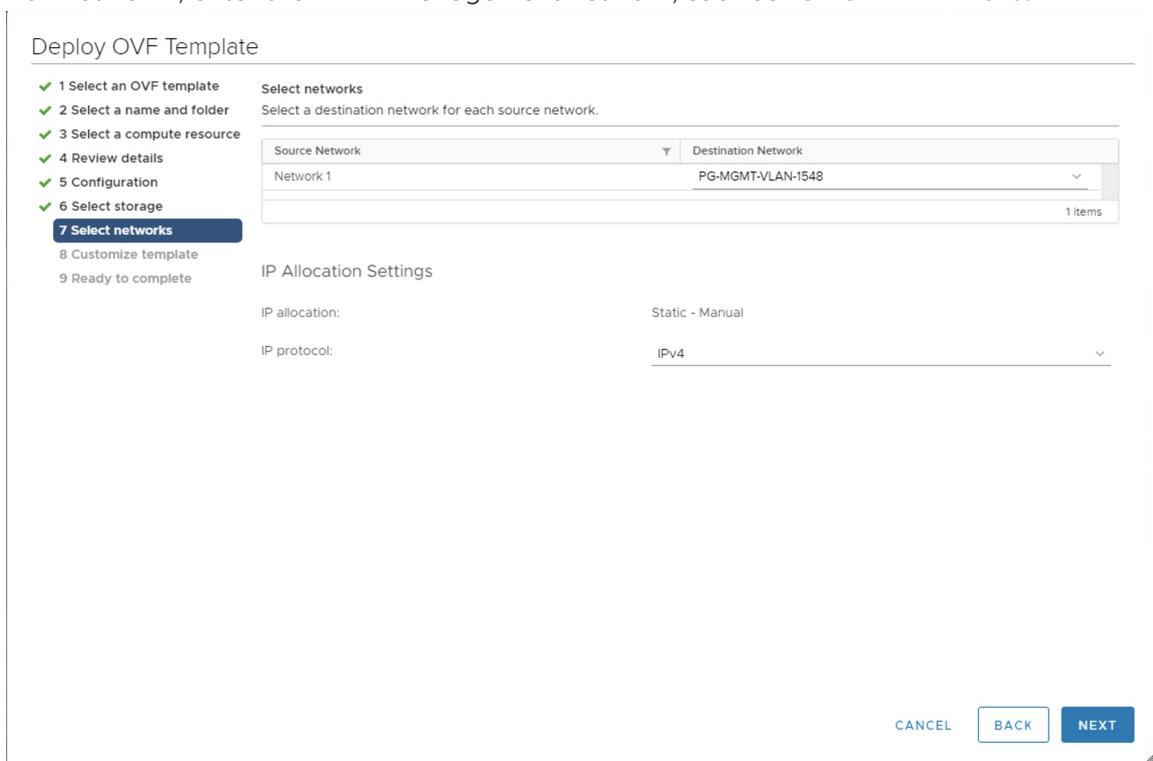
4 Items

CANCEL BACK NEXT

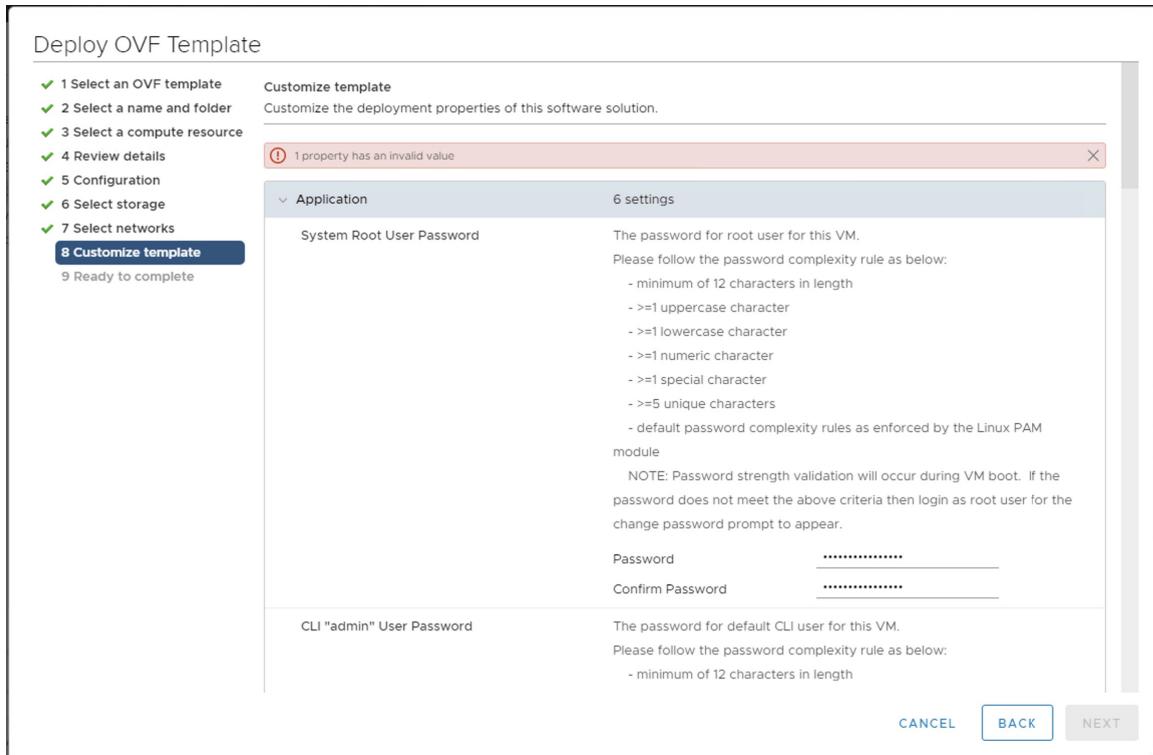
7. At the **Select storage** screen, choose **Thin Provision** and the desired datastore.



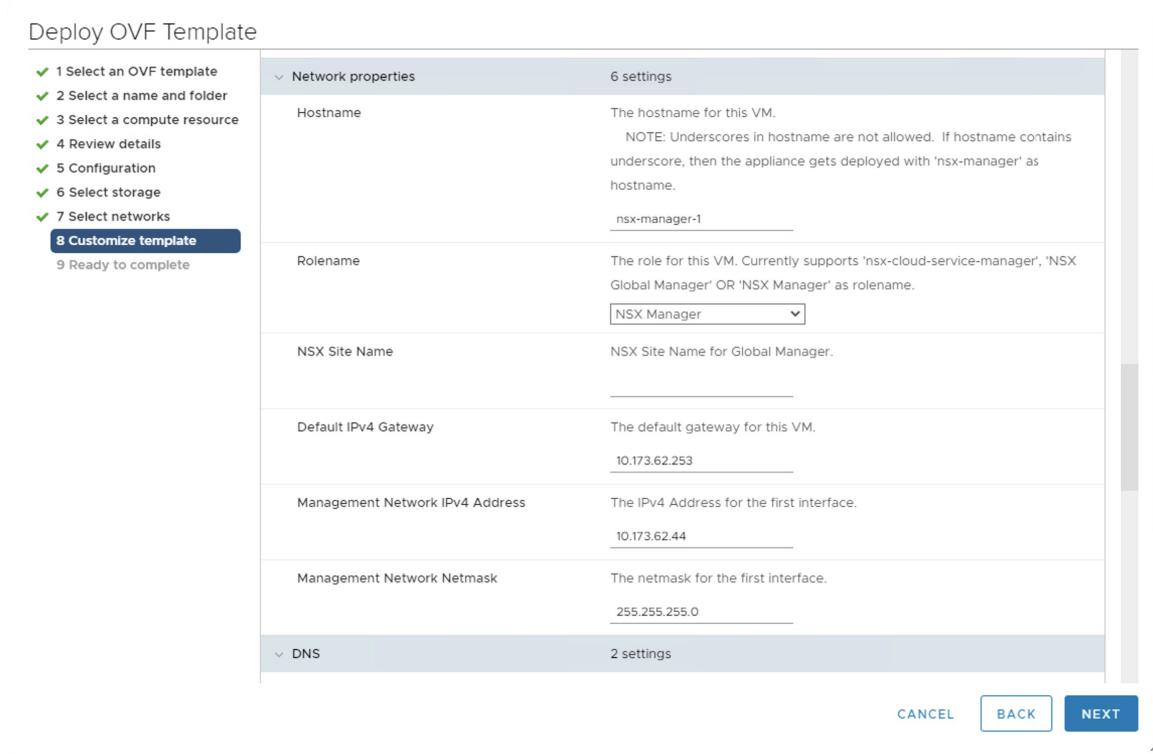
8. For **Network1**, enter the VLAN management network, such as [PG-MGMT-VLAN-1548](#).



9. Enter strong passwords for all user types.



10. Enter the hostname, such as `nsx-manager-1`.
11. Enter the rolename, such as `NSX Manager`.
12. Enter the Gateway IP address, such as `10.173.62.253`.
13. Enter a public IP address for the VM, such as `10.173.62.44`.
14. Enter the Netmask, such as `255.255.255.0`.



15. Enter the DNS server, such as `10.172.40.1`.

16. Enter the **NTP server**, such as **10.113.60.176**.
17. Enable the **Enable SSH** checkbox.
18. Enable the **Allow SSH root logins** checkbox.

**Deploy OVF Template**

|                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <ul style="list-style-type: none"> <li>✓ 1 Select an OVF template</li> <li>✓ 2 Select a name and folder</li> <li>✓ 3 Select a compute resource</li> <li>✓ 4 Review details</li> <li>✓ 5 Configuration</li> <li>✓ 6 Select storage</li> <li>✓ 7 Select networks</li> <li><b>8 Customize template</b></li> </ul> <p>9 Ready to complete</p> | <div style="background-color: #f0f0f0; padding: 5px;"> <p><b>DNS</b> 2 settings</p> <p>DNS Server list<br/>The space separated DNS server list for this VM (valid only if an IPv4 address is specified for the first interface).<br/>NOTE: At most three name servers can be configured (first 3 name servers passed in list will be used and all other will be ignored)<br/>10.172.40.1</p> <p>Domain Search List<br/>The space separated domain search list for this VM (valid only if an IPv4 address is specified for the first interface).</p> </div> <div style="background-color: #f0f0f0; padding: 5px;"> <p><b>Services Configuration</b> 3 settings</p> <p>NTP Server List<br/>The NTP server list(space separated) for this VM.<br/>10.113.60.176</p> <p>Enable SSH<br/>Enabling SSH service is not recommended for security reasons.<br/><input checked="" type="checkbox"/></p> <p>Allow root SSH logins<br/>Allowing root SSH logins is not recommended for security reasons.<br/><input checked="" type="checkbox"/></p> </div> <div style="background-color: #f0f0f0; padding: 5px;"> <p><b>Internal Properties - Do not set these parameters.</b> 5 settings</p> <p>Manager IP<br/>For internal use only. Do not set this parameter</p> </div> |
| <span style="border: 1px solid #ccc; padding: 2px 10px;">CANCEL</span> <span style="border: 1px solid #0070C0; background-color: #0070C0; color: white; padding: 2px 10px;">NEXT</span>                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

19. Click **Finish**, and NSX-T Manager 1 starts deploying.

**Deploy OVF Template**

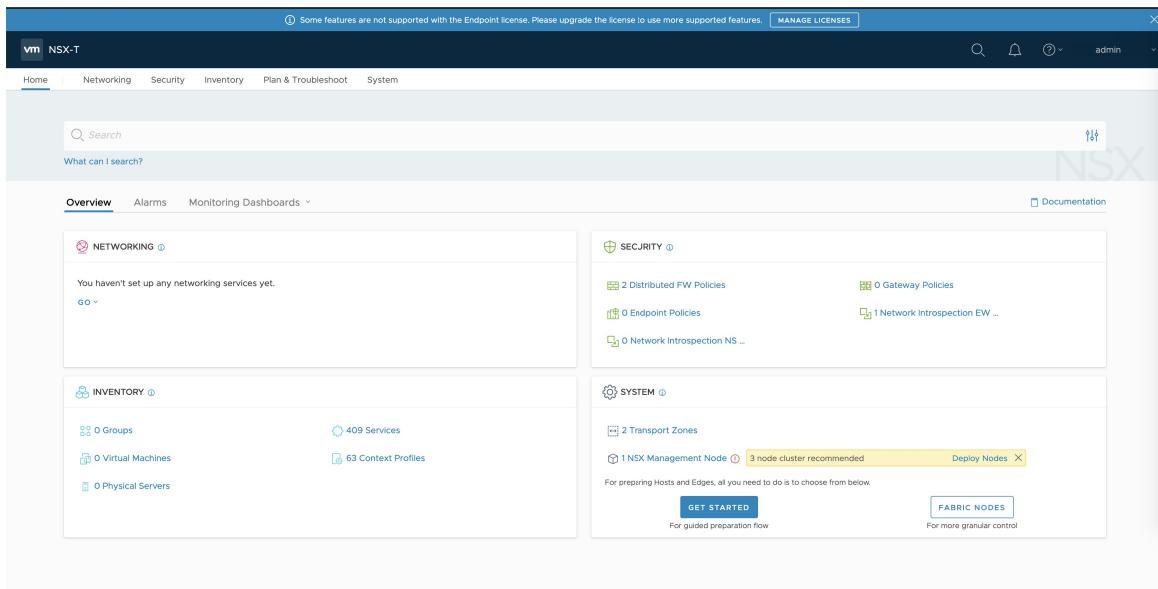
|                                                                                                                                                                                                                                                                                                                                                                                                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |      |                                          |               |                                          |               |         |              |        |        |            |          |               |                 |   |           |                                                   |                 |   |           |                   |                        |  |             |      |               |                 |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|------------------------------------------|---------------|------------------------------------------|---------------|---------|--------------|--------|--------|------------|----------|---------------|-----------------|---|-----------|---------------------------------------------------|-----------------|---|-----------|-------------------|------------------------|--|-------------|------|---------------|-----------------|
| <ul style="list-style-type: none"> <li>✓ 1 Select an OVF template</li> <li>✓ 2 Select a name and folder</li> <li>✓ 3 Select a compute resource</li> <li>✓ 4 Review details</li> <li>✓ 5 Configuration</li> <li>✓ 6 Select storage</li> <li>✓ 7 Select networks</li> <li>✓ 8 Customize template</li> <li><b>9 Ready to complete</b></li> </ul> <p>Ready to complete<br/>Click Finish to start creation.</p> | <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>Name</td> <td>nsx-unified-appliance-3.0.0.0.0.15946739</td> </tr> <tr> <td>Template name</td> <td>nsx-unified-appliance-3.0.0.0.0.15946739</td> </tr> <tr> <td>Download size</td> <td>11.0 GB</td> </tr> <tr> <td>Size on disk</td> <td>5.6 GB</td> </tr> <tr> <td>Folder</td> <td>Datacenter</td> </tr> <tr> <td>Resource</td> <td>10.172.210.60</td> </tr> <tr> <td>Storage mapping</td> <td>1</td> </tr> <tr> <td>All disks</td> <td>Datastore: datastore1 (1); Format: Thin provision</td> </tr> <tr> <td>Network mapping</td> <td>1</td> </tr> <tr> <td>Network 1</td> <td>PG-MGMT-VLAN-1548</td> </tr> <tr> <td>IP allocation settings</td> <td></td> </tr> <tr> <td>IP protocol</td> <td>IPV4</td> </tr> <tr> <td>IP allocation</td> <td>Static - Manual</td> </tr> </table> | Name | nsx-unified-appliance-3.0.0.0.0.15946739 | Template name | nsx-unified-appliance-3.0.0.0.0.15946739 | Download size | 11.0 GB | Size on disk | 5.6 GB | Folder | Datacenter | Resource | 10.172.210.60 | Storage mapping | 1 | All disks | Datastore: datastore1 (1); Format: Thin provision | Network mapping | 1 | Network 1 | PG-MGMT-VLAN-1548 | IP allocation settings |  | IP protocol | IPV4 | IP allocation | Static - Manual |
| Name                                                                                                                                                                                                                                                                                                                                                                                                       | nsx-unified-appliance-3.0.0.0.0.15946739                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |      |                                          |               |                                          |               |         |              |        |        |            |          |               |                 |   |           |                                                   |                 |   |           |                   |                        |  |             |      |               |                 |
| Template name                                                                                                                                                                                                                                                                                                                                                                                              | nsx-unified-appliance-3.0.0.0.0.15946739                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |      |                                          |               |                                          |               |         |              |        |        |            |          |               |                 |   |           |                                                   |                 |   |           |                   |                        |  |             |      |               |                 |
| Download size                                                                                                                                                                                                                                                                                                                                                                                              | 11.0 GB                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |      |                                          |               |                                          |               |         |              |        |        |            |          |               |                 |   |           |                                                   |                 |   |           |                   |                        |  |             |      |               |                 |
| Size on disk                                                                                                                                                                                                                                                                                                                                                                                               | 5.6 GB                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |      |                                          |               |                                          |               |         |              |        |        |            |          |               |                 |   |           |                                                   |                 |   |           |                   |                        |  |             |      |               |                 |
| Folder                                                                                                                                                                                                                                                                                                                                                                                                     | Datacenter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |      |                                          |               |                                          |               |         |              |        |        |            |          |               |                 |   |           |                                                   |                 |   |           |                   |                        |  |             |      |               |                 |
| Resource                                                                                                                                                                                                                                                                                                                                                                                                   | 10.172.210.60                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |      |                                          |               |                                          |               |         |              |        |        |            |          |               |                 |   |           |                                                   |                 |   |           |                   |                        |  |             |      |               |                 |
| Storage mapping                                                                                                                                                                                                                                                                                                                                                                                            | 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |      |                                          |               |                                          |               |         |              |        |        |            |          |               |                 |   |           |                                                   |                 |   |           |                   |                        |  |             |      |               |                 |
| All disks                                                                                                                                                                                                                                                                                                                                                                                                  | Datastore: datastore1 (1); Format: Thin provision                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |      |                                          |               |                                          |               |         |              |        |        |            |          |               |                 |   |           |                                                   |                 |   |           |                   |                        |  |             |      |               |                 |
| Network mapping                                                                                                                                                                                                                                                                                                                                                                                            | 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |      |                                          |               |                                          |               |         |              |        |        |            |          |               |                 |   |           |                                                   |                 |   |           |                   |                        |  |             |      |               |                 |
| Network 1                                                                                                                                                                                                                                                                                                                                                                                                  | PG-MGMT-VLAN-1548                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |      |                                          |               |                                          |               |         |              |        |        |            |          |               |                 |   |           |                                                   |                 |   |           |                   |                        |  |             |      |               |                 |
| IP allocation settings                                                                                                                                                                                                                                                                                                                                                                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |      |                                          |               |                                          |               |         |              |        |        |            |          |               |                 |   |           |                                                   |                 |   |           |                   |                        |  |             |      |               |                 |
| IP protocol                                                                                                                                                                                                                                                                                                                                                                                                | IPV4                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |      |                                          |               |                                          |               |         |              |        |        |            |          |               |                 |   |           |                                                   |                 |   |           |                   |                        |  |             |      |               |                 |
| IP allocation                                                                                                                                                                                                                                                                                                                                                                                              | Static - Manual                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |      |                                          |               |                                          |               |         |              |        |        |            |          |               |                 |   |           |                                                   |                 |   |           |                   |                        |  |             |      |               |                 |
| <span style="border: 1px solid #ccc; padding: 2px 10px;">CANCEL</span> <span style="border: 1px solid #0070C0; background-color: #0070C0; color: white; padding: 2px 10px;">BACK</span> <span style="border: 1px solid #0070C0; background-color: #0070C0; color: white; padding: 2px 10px;">FINISH</span>                                                                                                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |      |                                          |               |                                          |               |         |              |        |        |            |          |               |                 |   |           |                                                   |                 |   |           |                   |                        |  |             |      |               |                 |

20. Monitor the deployment using the **Recent Tasks** pane.

21. When the deployment completes, select the VM and power it on.
22. Access the NSX-T Manager 1 web console by navigating to the URL, such as:  
<https://10.173.62.44/>.

VMware® NSX-T™

23. Log in and verify the installation. Note the system message that a “3 node cluster” is recommended.



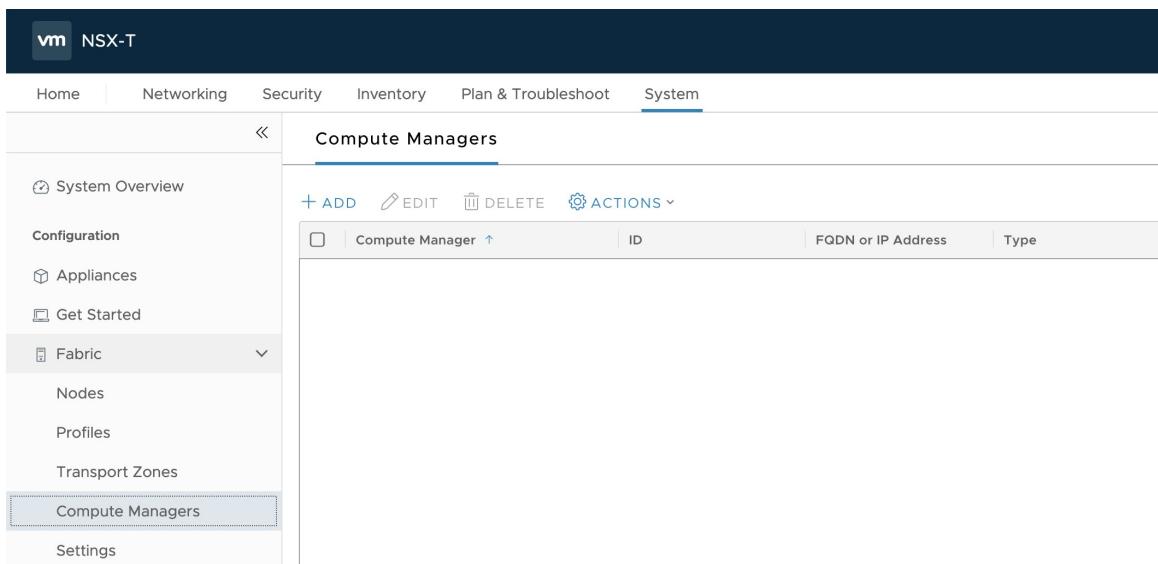
## Add vCenter as the Compute Manager

A compute manager is required for NSX-T environments with multiple NSX-T Manager nodes. A compute manager is an application that manages resources such as hosts and VMs. For TKGI we use the vCenter Server as the compute manager.

Complete the following steps to add vCenter as the Compute Manager. For additional guidance, refer to the [NSX-T documentation](#).

1. In the NSX Management console, navigate to **System > Appliances**.

2. Select **Compute Managers**.



3. Click **Add**.
4. Enter a **Name**, such as **vCenter**.
5. Enter an **IP address**, such as **10.173.62.43**.
6. Enter the vCenter username, such as **administrator@vsphere.local**.
7. Set the **Enable Trust** toggle to **Yes**.

## New Compute Manager

[?](#) [X](#)

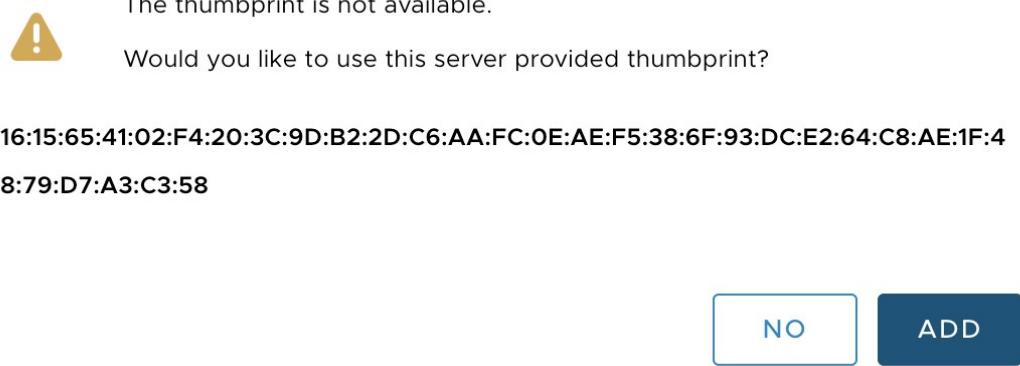
|                                                 |                                                                                      |
|-------------------------------------------------|--------------------------------------------------------------------------------------|
| Name *                                          | vCenter-demo                                                                         |
| Description                                     | <input type="text"/>                                                                 |
| Type *                                          | vCenter                                                                              |
| FQDN or IP Address *                            | 10.173.62.43                                                                         |
| HTTPS Port of Reverse Proxy * <a href="#">i</a> | 443                                                                                  |
| Username *                                      | administrator@vsphere.local                                                          |
| Password *                                      | *****                                                                                |
| SHA-256 Thumbprint                              | <input type="text"/>                                                                 |
| Enable Trust <a href="#">i</a>                  | <input checked="" type="checkbox"/> Yes<br>Supported for vCenter Server 7.0 or later |

[CANCEL](#) [ADD](#)

8. Click **Add**.
9. Click **Add** again at the thumbprint warning.

## Warning: Thumbprint is Missing

X



- Verify that the Compute Manager is added and registered.

|                          | ID                | FQDN or IP Address | Type         | Registration Status | Version    | Connection Status | Last Inventory Update | Alarms                     |
|--------------------------|-------------------|--------------------|--------------|---------------------|------------|-------------------|-----------------------|----------------------------|
| <input type="checkbox"/> | Compute Manager ↑ |                    |              |                     |            |                   |                       |                            |
| <input type="checkbox"/> | vCenter-demo      | Od04...248c        | 10.173.62.43 | vCenter             | Registered | 7.0.0             | Up                    | May 26, 2020 3:22:31 ... 0 |

## Deploy NSX-T Manager 2

Use the NSX-T Management Console to deploy an additional NSX-T Manager node as part of the NSX-T Management layer. For more information, refer to the [NSX-T documentation](#).

- In the NSX Management Console, navigate to **System > Appliances**.
- Select **Add NSX Appliance**.

NSX Appliances

- Cluster **STABLE**
- Cluster ID: [View ID](#)
- Virtual IP: [SET VIRTUAL IP](#)

**10.173.62.44**  
Available • IP: 10.173.62.44  
Version: 3.0.0.0.0.15946739

System Load: 1.5      Memory: 12 GB  
1 min ago      23 GB allocated  
6 vCPU allocated

[VIEW DETAILS](#)    [ACTIONS](#)

**ADD NSX APPLIANCE**

3. Enter a hostname, such as `nsx-manager-2`.
4. Enter the Management IP/netmask, such as `10.173.62.45/24`.
5. Enter the Gateway, such as `10.173.62.253`.
6. For the Node size, choose `medium`.

The screenshot shows the 'Add Appliance' wizard in progress. The left sidebar lists steps: 1. Appliance Information (selected), 2. Configuration, and 3. Access & Credentials. The main area is titled 'Appliance Information' and contains fields for Hostname, Management IP/Netmask, Management Gateway, DNS Servers, and NTP Servers. The 'Node Size' section shows three options: Small (4 vCPU, 16 GB RAM, 300 GB storage), Medium (6 vCPU, 24 GB RAM, 300 GB storage, selected), and Large (12 vCPU, 48 GB RAM, 300 GB storage). A 'LEARN MORE ABOUT APPLIANCE SELECTION' link is visible above the node size options. At the bottom right are 'CANCEL' and 'NEXT' buttons.

| Node Size | Description                                    |
|-----------|------------------------------------------------|
| Small     | 4 vCPU<br>16 GB RAM<br>300 GB storage          |
| Medium    | <b>6 vCPU<br/>24 GB RAM<br/>300 GB storage</b> |
| Large     | 12 vCPU<br>48 GB RAM<br>300 GB storage         |

7. For the Compute Manager, select `vCenter`.
8. For the Compute Cluster, enter `MANAGEMENT-cluster`.
9. For the Datastore, select the datastore, such as `datastore2`.
10. For the Virtual Disk Format, select `thin provision`.
11. For the Network, select the VLAN management network, such as `PG-MGMT-VLAN-1548`.

Add Appliance

Configuration

Compute Manager \* vCenter-demo

Compute Cluster \* MANAGEMENT-cluster (domain-c10)

Resource Pool Select resource pool

Host Select host

Datastore \* datastore2 (datastore-22)

Virtual Disk Format Thin Provision

Network \* PG-MGMT-VLAN-1548

CANCEL BACK NEXT

12. Select **Enable SSH**.
13. Select **Enable root access**.
14. Enter a strong password.

Add Appliance

Access & Credentials

Enable SSH  Yes

Enable Root Access  Yes

System Root Credentials

System Username root

Root Password VMware1!VMware1!

Confirm Root Password VMware1!VMware1!

Admin CLI Credentials

CLI Username admin

CLI password  Same as root password

Audit CLI Credentials

Audit CLI Username audit

Audit CLI password  Same as root password

INSTALL APPLIANCE

15. Click **Install Appliance**.
16. Verify that the NSX-T Manager 2 appliance is added.

**NSX-T**

The screenshot shows the NSX-T Appliances page. On the left, a sidebar menu includes Home, Networking, Security, Inventory, Plan & Troubleshoot, and System. Under Configuration, the Appliances option is selected. The main panel displays the NSX Appliances section. A cluster named "10.173.62.44" is listed as "Available" with IP 10.173.62.44 and Version 3.0.0.0.15946739. Another node, "nsx-manager-2", is shown as "1% Installing". There is also a placeholder for adding an NSX Intelligence Appliance.

**NSX-T**

The screenshot shows the NSX-T Appliances page. The sidebar menu is identical to the first one. In the main panel, the NSX Appliances section shows a cluster status of "DEGRADED". Two nodes are listed: "10.173.62.44" (Available, IP 10.173.62.44, Version 3.0.0.0.15946739) and "10.173.62.45" (Available, IP 10.173.62.45, Version 3.0.0.0.15946739). Both nodes show system load metrics and memory usage.

The screenshot shows the NSX-T Management Console interface. The top navigation bar includes Home, Networking, Security, Inventory, Plan & Troubleshoot, and System. The System tab is selected. On the left, a collapsed sidebar lists System Overview, Configuration (with Appliances selected), Get Started, Fabric (Nodes, Profiles, Transport Zones, Compute Managers, Settings), Service Deployments (Identity Firewall AD), and Lifecycle Management. The main content area is titled 'Appliances' and displays 'NSX Appliances'. It shows two entries: '10.173.62.44' and '10.173.62.45', both marked as 'Available' with their IP addresses and versions. Each entry includes a 'Cluster ID' (10.173.62.44 and 10.173.62.45 respectively), a 'View ID' button, and a 'Virtua' button. Below each entry is a chart showing 'System Load' (1 min ago) and 'Memory' usage (23 GB allocated). At the bottom of each card are 'VIEW DETAILS' and 'ACTIONS' buttons.

## Deploy NSX-T Manager 3

Use the NSX-T Management Console to deploy a third NSX-T Manager node as part of the NSX-T Management layer. For more information, refer to the [NSX-T documentation](#).

1. In the NSX Management Console, navigate to **System > Appliances**.
2. Select **Add NSX Appliance**.
3. Enter a hostname, such as `nsx-manager-3`.
4. Enter the Management IP/netmask, such as `10.173.62.46/24`.
5. Enter the Gateway, such as `10.173.62.253`.
6. For the Node size, choose `medium`.
7. For the Compute Manager, select `vCenter`.
8. For the Compute Cluster, enter `MANAGEMENT-cluster`.
9. For the Datastore, select the datastore, such as `datastore2`.
10. For the Virtual Disk Format, select `thin provision`.
11. For the Network, select the VLAN management network, such as `PG-MGMT-VLAN-1548`.
12. Select **Enable SSH**.
13. Select **Enable root access**.
14. Enter a strong password.
15. Click **Install Appliance**.
16. Verify that the NSX-T Manager 3 appliance is added.

The screenshot shows the NSX-T Management Console interface. The left sidebar contains navigation links for Home, Networking, Security, Inventory, Plan & Troubleshoot, and System. Under the System category, 'Appliances' is selected. The main content area is titled 'Appliances' and shows three NSX Appliances listed under 'NSX Appliances'. Each appliance entry includes its Cluster ID, status (e.g., Available), IP address, version, system load, memory usage, and vCPU allocated. Below the appliances, there is a separate section for the 'NSX Intelligence Appliance' with a button to add it.

## Configure the NSX-T Management VIP

The NSX-T Management layer includes three NSX-T Manager nodes. To support a single access point, assign a virtual IP Address (VIP) to the NSX-T Management layer. Once the VIP is assigned, any UI and API requests to NSX-T are redirected to the virtual IP address of the cluster, which is owned by the leader node. The leader node then routes the request forward to the other components of the appliance.

Using a VIP makes the NSX Management Cluster highly-available. If you need to scale, an alternative to the VIP is to provision a load balancer for the NSX-T Management Cluster. Provisioning a load balancer requires that NSX-T be fully installed and configured. It is recommended that you configure the VIP now, then install a load balancer after NSX-T is installed and configured, and only if needed.

Complete the following instructions to create a VIP for the NSX Management Cluster. The IP address you use for the VIP must be part of the same subnet as the NSX-T Management nodes.

1. In the NSX Management Console, navigate to **System > Appliances**.
2. Click the **Set Virtual IP** button.

3. Enter a Virtual IP address, such as [10.173.62.47](#).

## Set Virtual IP

NSX-T Managers Cluster offer a built-in VIP for high-availability, but the usage of an external load balancer offers the following benefits: 1) Load spread across all NSX-T Managers; 2) NSX-T Managers can be in different subnets and 3) Faster failover.

**Virtual IP Address\*** [10.173.62.47](#)

[CANCEL](#)

[SAVE](#)

4. Verify that the VIP is added.

## Set Virtual IP

NSX-T Managers Cluster offer a built-in VIP for high-availability, but the usage of an external load balancer offers the following benefits: 1) Load spread across all NSX-T Managers; 2) NSX-T Managers can be in different subnets and 3) Faster failover.



Setting up the Virtual IP Address for the Management Cluster.

It might take few minutes to set a new IP address.

[REFRESH](#)

**NSX Appliances**

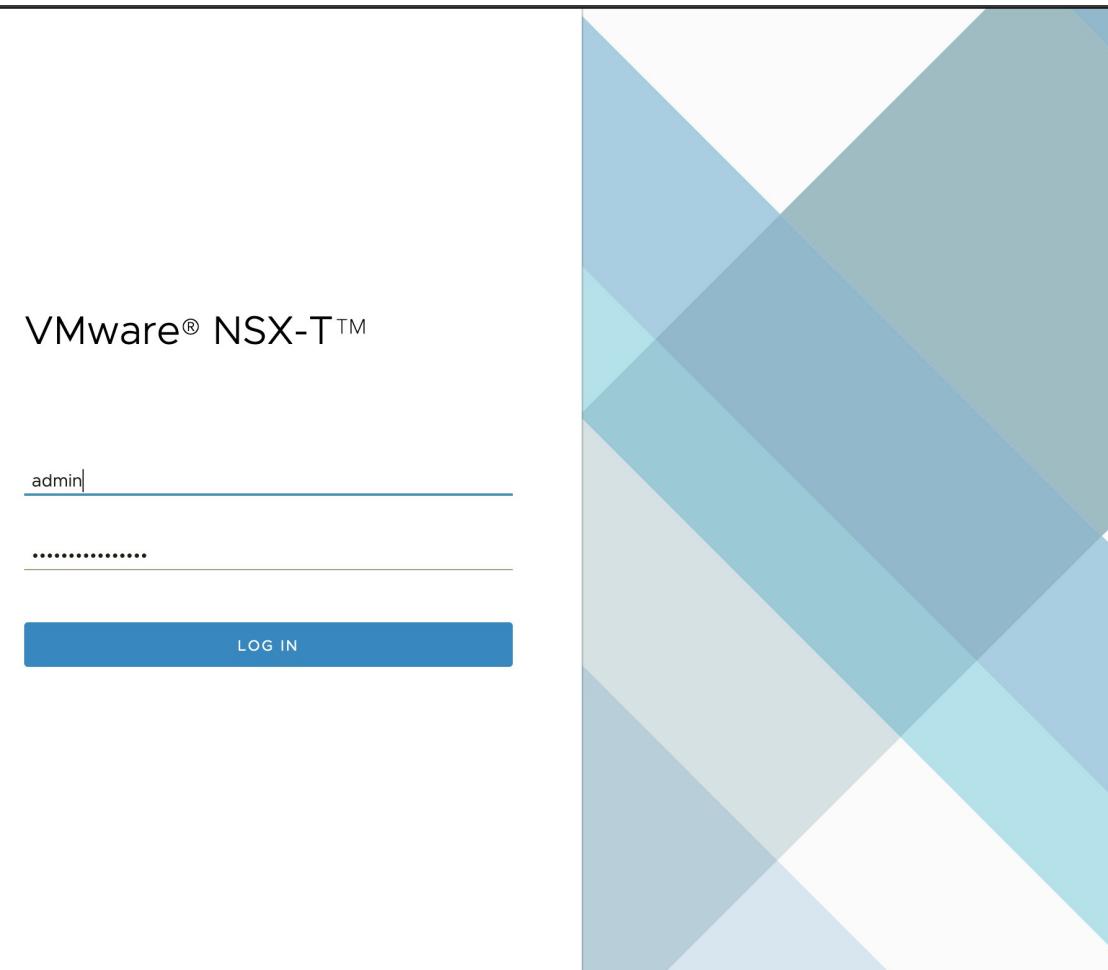
| Cluster                                                                   | Cluster ID | Virtual IP                                                                |
|---------------------------------------------------------------------------|------------|---------------------------------------------------------------------------|
| 10.173.62.44<br>Available • IP: 10.173.62.44<br>Version: 3.0.0.0.15946739 | View ID    | 10.173.62.47<br>Assigned to 10.173.62.46                                  |
| 10.173.62.45<br>Available • IP: 10.173.62.45<br>Version: 3.0.0.0.15946739 | View ID    | 10.173.62.46<br>Available • IP: 10.173.62.46<br>Version: 3.0.0.0.15946739 |
| 10.173.62.46<br>Available • IP: 10.173.62.46<br>Version: 3.0.0.0.15946739 | View ID    | 10.173.62.47<br>Assigned to 10.173.62.46                                  |

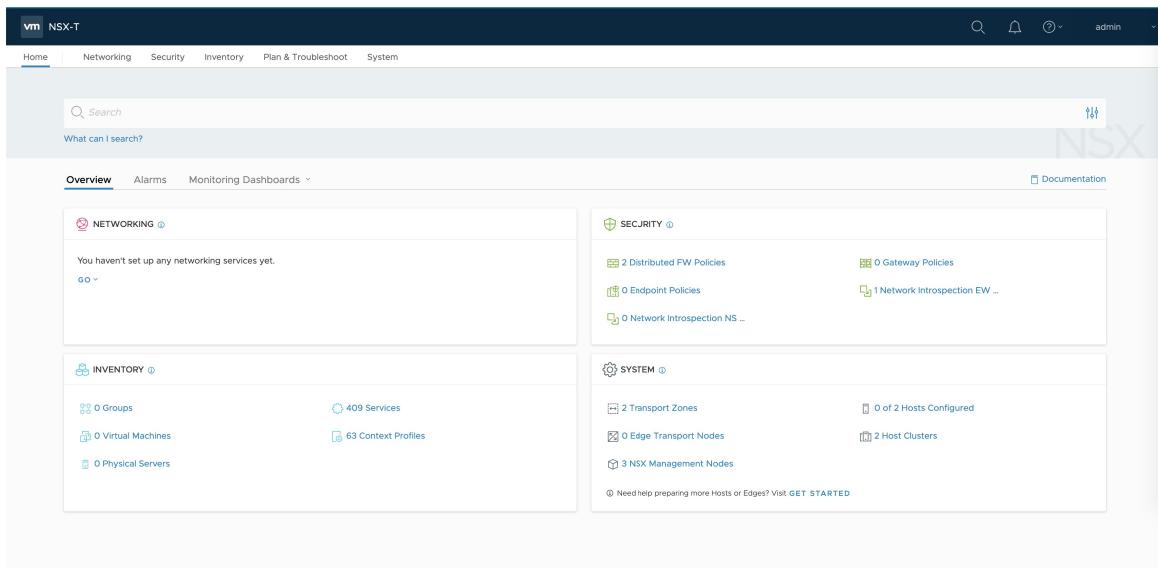
**NSX Intelligence Appliance**

ADD NSX INTELLIGENCE APPLIANCE

5. Access the NSX-T Management console using the VIP, such as

<https://10.173.62.47/login.jsp>.





## Add the NSX-T Manager License

If you do not add the proper NSX-T license, you will receive an error when you try to deploy an Edge Node VM.

1. In the NSX-T Manager console, navigate to **System > Licenses**.
2. Add the NSX Data Center Advanced (CPU) license.
3. Verify that the license is added.

## Enable the NSX-T Manager Interface (if necessary)

The NSX Management Console provides two user interfaces: **Policy** and **Manager**. TKGI supports both.

The **Policy** interface is the default. If you are using the **Manager** interface for configuring the networking and security objects, you need to enable the Manager interface.

1. In the NSX-T Manager console, navigate to **System > User Interface Settings**.

User Interface Settings

User Interface Mode Toggle [EDIT](#)

|                   |           |
|-------------------|-----------|
| Toggle Visibility | Not Set ⓘ |
| Default Mode      | Not Set ⓘ |

2. Click **Edit**.
3. For the **Toggle Visibility** field, select **Visible to all Users**.
4. For the **Default Mode** field, select **Manager**.
5. Click **Save**.

User Interface Settings

User Interface Mode Toggle [EDIT](#)

|                   |                        |
|-------------------|------------------------|
| Toggle Visibility | Visible to All Users ⓘ |
| Default Mode      | Manager ⓘ              |

**SAVE** **CANCEL**

6. Refresh the NSX-T Manager Console and navigate to an area of the console that is not listed under **System**.

- In the upper-right area of the console, verify that the **Manager** option is enabled.

## Generate and Register the NSX-T Management TLS Certificate and Private Key

This topic provides instructions for installing and configuring NSX-T Data Center v3 for use with VMware Tanzu Kubernetes Grid Integrated Edition on vSphere.

### Prerequisites

Before completing this section, make sure you have completed the following sections:

- [NSX-T Installation Prerequisites](#)
- [Install and Configure the NSX-T Manager Hosts](#)

## Generate and Register the NSX-T Management TLS Certificate and Private Key

An SSL certificate is automatically created for each NSX-T Manager. You can verify this by SSHing to one of the NSX Manager nodes and running the following command.

```
nsx-manager-1> get certificate cluster
```

You will see that Subject Alternative Name (SAN) listed in the certificate is the hostname of the appliance, for example `SAN=nsx-manager-1`. This means the cluster certificate is linked to a particular NSX Manager, in this case NSX-T Manager 1.

If you examine **System > Certificates**, you will see that the NSX-T manager VIP does not have a certificate. You must generate a new SSL certificate that uses the NSX-T Management VIP address so that the cluster certificate contains `SAN=VIP-ADDRESS`.

| ID          | Issued To              | Issued By              | Validity              | Type        |
|-------------|------------------------|------------------------|-----------------------|-------------|
| 2430...3b80 | VMware-NSX-AppProxyHub | VMware-NSX-AppProxyHub | 5/26/2020 - 5/24/2030 | Self Signed |
| 0e63...bde9 | VMware-NSX-AppProxyHub | VMware-NSX-AppProxyHub | 5/26/2020 - 5/24/2030 | Self Signed |
| 93e1...d850 | VMware-NSX-AppProxyHub | VMware-NSX-AppProxyHub | 5/26/2020 - 5/24/2030 | Self Signed |
| d895...3e49 | local-manager          | local-manager          | 5/26/2020 - 8/29/2022 | Self Signed |
| f9df...e0ef | nsx-manager-1          | nsx-manager-1          | 5/26/2020 - 8/29/2022 | Self Signed |
| 7d83...4bfd | nsx-manager-1          | nsx-manager-1          | 5/26/2020 - 8/29/2022 | Self Signed |
| 3afa...ca02 | nsx-manager-2          | nsx-manager-2          | 5/26/2020 - 8/29/2022 | Self Signed |
| 83e4...eb4f | nsx-manager-3          | nsx-manager-3          | 5/26/2020 - 8/29/2022 | Self Signed |

Complete the following steps to generate and register a SSL certificate and private key that uses the VIP address. The following steps assume that you are working on a Linux host where OpenSSL is installed.

## Generate the SSL Certificate and Private Key

1. Create a certificate signing request file named `nsx-cert.cnf` and populate it with the contents below.

```
[req]
default_bits = 2048
default_md = sha256
prompt = no
distinguished_name = req_distinguished_name
x509_extensions = v3_req
req_extensions = v3_req

[req_distinguished_name]
countryName = US
stateOrProvinceName = California
localityName = CA
organizationName = NSX
commonName = VIP-ADDRESS #CAN ONLY USE IF SAN IS ALSO USED

[v3_req]
basicConstraints = critical,CA:false
subjectKeyIdentifier = hash
authorityKeyIdentifier=keyid:always,issuer:always
subjectAltName = DNS:NSX-VIP-FQDN, IP:VIP-ADDRESS #MUST USE
```

Where:

- `NSX-VIP-FQDN` is your NSX VIP FQDN.
- `VIP-ADDRESS` is the VIP address for the NSX-T Management cluster.



**Note:** At a minimum you must use the SAN field for identifying the NSX Management VIP. You can also use the CN field, as long as the SAN field is populated. If you use only the CN field, the certificate will not be valid for TKGI.

2. Copy the `nsx-cert.cnf` file to a machine with `openssl` if yours does not have it.
3. Use [OpenSSL](#) to generate the SSL certificate and private key.

```
openssl req -newkey rsa -nodes -days 1100 -x509 -config nsx-cert.cnf -keyout nsx.key -out nsx.crt
```

4. Verify that you see the following:

```
Generating a 2048 bit RSA private key
.....+++
.....+++
writing new private key to 'nsx.key'
```

5. Verify certificate and key generation by running the `ls` command.

You should see three files: the initial signing request, and the certificate and private key

generated by running the signing request.

```
nsx-cert.cnf nsx.crt nsx.key
```

- Run the following command to verify the certificate and private key.

```
openssl x509 -in nsx.crt -text -noout
```

You should see that the Subject Alternative Name (SAN) and common name (CN) (if used) are both the VIP address. For example:

```
Subject: C=US, ST=California, L=CA, O=NSX, CN=myvip.mydomain.com
...
X509v3 extensions:
 X509v3 Subject Alternative Name:
 DNS:myvip.mydomain.com, IP Address:10.11.12.13
```

## Import the SSL Certificate and Private Key to the NSX-T Management Console

Import certificate and private key to NSX-T by completing the following steps. These steps require populating the NSX-T Management Console fields with the certificate and private key. You can copy/paste the contents, or if you save the `nsx.crt` and `nsx.key` files to your local machine, you can import them.

- In the NSX-T Management Console, navigate to the **System > Certificates** page.

| Certificate                                         | ID        | Issued To              | Issued By              | Validity              | Type        |
|-----------------------------------------------------|-----------|------------------------|------------------------|-----------------------|-------------|
| APH certificate for node 02d06348-dab6-4d73-8628... | 2430_3b80 | VMware-NSX-AppProxyHub | VMware-NSX-AppProxyHub | 5/26/2020 - 5/24/2030 | Self Signed |
| APH certificate for node 67272642-7090-fate-ec70... | 0e63_bde9 | VMware-NSX-AppProxyHub | VMware-NSX-AppProxyHub | 5/26/2020 - 5/24/2030 | Self Signed |
| APH certificate for node 6e270445-1b09-43bc-820a... | 93e1_d850 | VMware-NSX-AppProxyHub | VMware-NSX-AppProxyHub | 5/26/2020 - 5/24/2030 | Self Signed |
| LocalManager                                        | dc95_3e49 | local-manager          | local-manager          | 5/26/2020 - 8/29/2022 | Self Signed |
| mp-cluster certificate for node nsx-manager-1       | f9df_e0ef | nsx-manager-1          | nsx-manager-1          | 5/26/2020 - 8/29/2022 | Self Signed |
| tomcat certificate for node nsx-manager-1           | 7d83_4bfd | nsx-manager-1          | nsx-manager-1          | 5/26/2020 - 8/29/2022 | Self Signed |
| tomcat certificate for node nsx-manager-2           | 3afa_ca02 | nsx-manager-2          | nsx-manager-2          | 5/26/2020 - 8/29/2022 | Self Signed |
| tomcat certificate for node nsx-manager-3           | 83e4_eb4f | nsx-manager-3          | nsx-manager-3          | 5/26/2020 - 8/29/2022 | Self Signed |

- Click **Import > Import Certificate**. The **Import Certificate** screen is displayed.



**Note:** Be sure to select **Import Certificate** and not **Import CA Certificate**.

- Enter a **Name**. For example, `CERT-NSX-T-VIP`.
- Copy and paste the **Certificate Contents** from the `nsx.crt` file. Or, import the `nsx.crt` file clicking **Browse** and selecting it.
- Copy and paste the **Private Key** from the `nsx.key` file. Or, import the `nsx.key` file by clicking

**Browse** and selecting it.

6. For the **Service Certificate** option, make sure to select **No**.
7. Click **Import**.

## Import Certificate

(?) X

Name \*

CERT-NSX-T-VIP

Certificate Contents \*

R4IA880Y  
-----END CERTIFICATE-----**BROWSE...**

Private Key

/L3pZqtKG4EMWlyCZhgsfkI=  
-----END PRIVATE KEY-----**BROWSE...**

Passphrase

Description

Service Certificate

 No

Turn Service Certificate on to use the certificate with services such as Load Balancer and VPN.

Turn Service Certificate off to use the certificate with NSX Manager appliance nodes.

**CANCEL****IMPORT**

8. Verify that you see the certificate in the list of Certificates.

[Certificates](#) [CSRs](#) [CRLs](#)

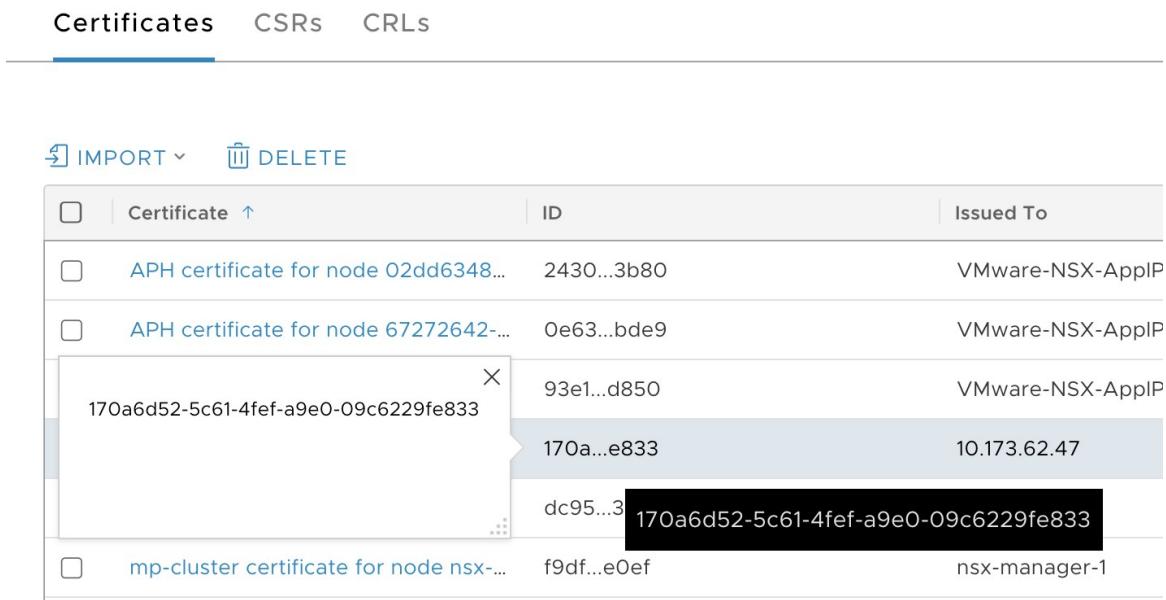
| <b>IMPORT</b> <span>DELETE</span>   |                                         | ID          | Issued To               | Issued By               | Validity                | Type        |
|-------------------------------------|-----------------------------------------|-------------|-------------------------|-------------------------|-------------------------|-------------|
| <input type="checkbox"/>            | Certificate                             | 2430...3b80 | VMware-NSX-AppiProxyHub | VMware-NSX-AppiProxyHub | ● 5/26/2020 - 5/24/2030 | Self Signed |
| <input type="checkbox"/>            | APH certificate for node 67272642...    | 0e63...bde9 | VMware-NSX-AppiProxyHub | VMware-NSX-AppiProxyHub | ● 5/26/2020 - 5/24/2030 | Self Signed |
| <input type="checkbox"/>            | APH certificate for node 6e270445...    | 93e1...d850 | VMware-NSX-AppiProxyHub | VMware-NSX-AppiProxyHub | ● 5/26/2020 - 5/24/2030 | Self Signed |
| <input checked="" type="checkbox"/> | CERT-NSX-T-VIP                          | 170a...e833 | 10.173.62.47            | 10.173.62.47            | ● 5/27/2020 - 2/21/2023 | Self Signed |
| <input type="checkbox"/>            | LocalManager                            | dc95...3e49 | local-manager           | local-manager           | ● 5/26/2020 - 8/29/2022 | Self Signed |
| <input type="checkbox"/>            | mp-cluster certificate for node nsx-... | f9df...e0ef | nsx-manager-1           | nsx-manager-1           | ● 5/26/2020 - 8/29/2022 | Self Signed |
| <input type="checkbox"/>            | tomcat certificate for node nsx-ma...   | 7d83...4bfd | nsx-manager-1           | nsx-manager-1           | ● 5/26/2020 - 8/29/2022 | Self Signed |
| <input type="checkbox"/>            | tomcat certificate for node nsx-ma...   | 3afa...ca02 | nsx-manager-2           | nsx-manager-2           | ● 5/26/2020 - 8/29/2022 | Self Signed |
| <input type="checkbox"/>            | tomcat certificate for node nsx-ma...   | 83e4...eb4f | nsx-manager-3           | nsx-manager-3           | ● 5/26/2020 - 8/29/2022 | Self Signed |

## Register the SSL Certificate and Private Key with the NSX-T API Server

To register the imported VIP certificate with the NSX-T Management Cluster Certificate API,

complete the following steps:

1. In the NSX-T Management Console, navigate to the **System > Certificates** page.
2. View the UUID of the certificate from the **NSX-T Management Console > Certificates** screen.



| <input type="checkbox"/> Certificate ↑                           | ID          | Issued To                            |
|------------------------------------------------------------------|-------------|--------------------------------------|
| <input type="checkbox"/> APH certificate for node 02dd6348...    | 2430...3b80 | VMware-NSX-AppIP                     |
| <input type="checkbox"/> APH certificate for node 67272642...    | 0e63...bde9 | VMware-NSX-AppIP                     |
| <input type="checkbox"/> 170a6d52-5c61-4fef-a9e0-09c6229fe833    | 93e1...d850 | VMware-NSX-AppIP                     |
|                                                                  | 170a...e833 | 10.173.62.47                         |
|                                                                  | dc95...3    | 170a6d52-5c61-4fef-a9e0-09c6229fe833 |
| <input type="checkbox"/> mp-cluster certificate for node nsx-... | f9df...e0ef | nsx-manager-1                        |

3. Copy the UUID to the clipboard. For example, `170a6d52-5c61-4fef-a9e0-09c6229fe833`.
4. Create the following environment variables. Replace the IP address with your VIP address and the UUID with the UUID of the imported certificate.

```
export NSX_MANAGER_IP_ADDRESS=10.173.62.47
export CERTIFICATE_ID=170a6d52-5c61-4fef-a9e0-09c6229fe833
```

5. Post the certificate to the NSX-T Manager API.

```
curl --insecure -u admin:'VMware1!VMware1!' -X POST "https://$NSX_MANAGER_IP_ADDRESS/api/v1/cluster/api-certificate?action=set_cluster_certificate&certificate_id=$CERTIFICATE_ID"
{
 "certificate_id": "170a6d52-5c61-4fef-a9e0-09c6229fe833"
}
```

6. (Optional) If you are running TKG in a test environment and you are not using a multi-node NSX Management cluster, then you must also post the certificate to the Nodes API.

```
curl --insecure -u admin:'VMware1!VMware1!' -X POST https://$NSX_MANAGER_IP_ADDRESS/api/v1/node/services/http?action=apply_certificate&certificate_id=$CERTIFICATE_ID
{
 "certificate_id": "170a6d52-5c61-4fef-a9e0-09c6229fe833"
}
```



**Note:** Using a single-node NSX Management cluster is an unsupported configuration.

- Verify by SSHing to one of the NSX-T Management nodes and running the following command.

The certificate that is returned should match the generated one.

```
nsx-manager-1> get certificate cluster
```

## Create an IP Pool for VTEP

This topic provides instructions for installing and configuring NSX-T Data Center v3.0 for use with VMware Tanzu Kubernetes Grid Integrated Edition on vSphere.

## Create an IP Pool for VTEP

Tunnel endpoints (TEPs) are the source and destination IP addresses used in the external IP header to identify the ESXi hosts that originate and end the NSX-T encapsulation of overlay frames. The TEP addresses do not need to be routable so you can use any random IP addressing scheme you want. For more information, refer to the [NSX-T Data Center documentation](#).

- In the NSX-T Management Console, select the **Manager** interface (upper right).
- Navigate to **Networking > IP Address Pool**.

| ID          | Subnets | Allocations |
|-------------|---------|-------------|
| 34f3...2b74 | 1       | 0 of 248    |
| 060a...4127 | 1       | 0 of 1022   |
| 5dee...3793 | 1       | 0 of 248    |

- Click **Add**.
- Enter a **Name**, such as **TEP-IP-POOL**.
- Enter an IP range, such as **192.23.213.1 – 192.23.213.10**.
- Enter a CIDR address, such as **192.23.213.0/24**.

Add New IP Pool

Name\* TEP-IP-POOL

Description

Subnets

+ ADD DELETE

| <input checked="" type="checkbox"/> IP Ranges*                   | Gateway | CIDR*           | DNS Servers | DNS Suffix |
|------------------------------------------------------------------|---------|-----------------|-------------|------------|
| <input checked="" type="checkbox"/> 192.23.213.1 - 192.23.213.10 |         | 192.23.213.0/24 |             |            |

CANCEL ADD

7. Click **Add**.
8. Verify that the pool is added.

IP Pools IP Blocks

+ ADD EDIT DELETE ACTIONS

| IP Pools                    | ID          | Subnets | Allocations |
|-----------------------------|-------------|---------|-------------|
| SL_Destination_IP_Pool      | 34f3...2b74 | 1       | 0 of 248    |
| SL_Service_Chain_ID_IP_Pool | 060a...4127 | 1       | 0 of 1022   |
| SL_Source_IP_Pool           | 5dee...3793 | 1       | 0 of 248    |
| TEP-IP-POOL                 | dfcf...b5d3 | 1       | 0 of 10     |

## Configuring NSX-T Data Center v3 Transport Zones and Edge Node Switches for Tanzu Kubernetes Grid Integrated Edition

This topic provides instructions for configuring NSX-T Data Center v3 Transport Zones and N-VDS switches on NSX Edge Nodes for use with VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) on vSphere.

## Prerequisites

Before completing this section, make sure you have completed the following sections:

- [NSX-T Installation Prerequisites](#)
- [Install and Configure the NSX-T Manager Hosts](#)
- [Generate and Register the NSX-T TLS Certificate and Private Key](#)
- [Create an IP Pool for VTEP](#)

## Overview of Transport Zones for NSX-T

TKGI requires two Transport Zones for TKGI: an Overlay Transport Zone for the ESXi Transport Nodes and a VLAN Transport Zone for Edge Nodes.

TKGI requires that the host switch name associated with the Transport Zones match exactly the **Edge Switch Name** value that you specify when you configure an NSX Edge Node for use with TKGI.

You can configure your Transport Zones in three ways. The three configuration options require different levels of customization to complete:

| Configuration                             | Transport Zone    | Host Switch Name  |
|-------------------------------------------|-------------------|-------------------|
| Option 1: Use the Default Transport Zones | No customization  | No customization  |
| Option 2: Create Custom Transport Zones   | Yes customization | No customization  |
| Option 3: Use the NSX API                 | Yes customization | Yes customization |



**Note:** In NSX-T 3.1 and later, the Transport Zone Host Switch Name has been deprecated and removed from the NSX-T configuration UI. For more information, see [TKGI NSX Edge Switch and Transport Zone Host Switch Name Requirements](#).

## Configure Your NSX Transport Zones for TKGI

TKGI requires the NSX **Edge Switch Name** and the Transport Zone host switch name to be identical. You can configure identical Edge Switch and Transport Zone host switch names using the following methods:

- Option 1: Use the Default Transport Zones with a Single N-VDS Switch (recommended)
- Option 2: Create Custom Transport Zones and Use the NSX API to Get the Host Switch Names
- Option 3: Use the NSX API to Create Custom Transport Zones and NSX Switches

### Option 1: Use the Default Transport Zones with a Single N-VDS Switch

By default NSX-T v3.x creates two transport zones for you: `nsx-overlay-transportzone` and `nsx-vlan-transportzone`. Both default Transport Zones use a single N-VDS host switch that is named `nsxHostSwitch`. The advantage of using the default Transport Zones is twofold. First, it simplifies the Edge Node configuration process. Second, you need only a single N-VDS for the Edge Nodes.

To use this option:

1. Do not create a Transport Zone.
2. Deploy the Edge Nodes and configure NSX.
3. Specify `nsxHostSwitch` as the **Edge Switch Name**.
4. Select both default transport zones, `nsx-overlay-transportzone` and `nsx-vlan-transportzone`.

For example:

| Field            | Description                                     |
|------------------|-------------------------------------------------|
| Edge Switch Name | <code>nsxHostSwitch</code> (must match exactly) |

|                       |                                                                                                                      |
|-----------------------|----------------------------------------------------------------------------------------------------------------------|
| <b>Transport Zone</b> | <code>nsx-overlay-transportzone</code> and <code>nsx-vlan-transportzone</code> (select both default transport zones) |
| <b>Uplink Profile</b> | <code>nsx-edge-single-nic-uplink-profile</code>                                                                      |
| <b>IP Assignment</b>  | Use IP Pool                                                                                                          |
| <b>IP Pool</b>        | TEP-IP-POOL                                                                                                          |
| <b>Uplinks</b>        | uplink-1 / EDGE-VTEP-PG                                                                                              |



**Note:** If you use the default Transport Zones, but do not use the exact name `nsxHostSwitch` when configuring NSX on the Edge Node, you will receive the `pks-nsx-t-osb-proxy` BOSH error when you try to deploy TKGI.

## Option 2: Create Custom Transport Zones and Use the NSX API to Get the Host Switch Names

If you want to create a custom Transport Zone, you can do so using the NSX user interface. In this case, because the host switch name is deprecated from the NSX user interface, you have to make an NSX API call to get the host switch name so that you can configure the Edge Nodes with the correct switch.

When you create a custom Transport Zone using the NSX web interface, NSX will generate the associated switch name for you.

To create custom Transport Zones using the Host Switch names:

1. Create a custom Overlay Transport Zone:
  1. In the NSX-T Management Console, navigate to **System > Fabric > Transport Zone**.
  2. Click **Add**.
  3. Enter a Name, such as `tz-overlay`.
  4. For the **Traffic Type**, select `Overlay`.
  5. Click **Add**.
  6. Verify that you see the newly created Transport Zone named `tz-overlay` in the list.
2. Create a custom VLAN Transport Zone:
  1. In the NSX-T Management Console, navigate to **System > Fabric > Transport Zone**.
  2. Click **Add**.
  3. Enter a name, such as `tz-vlan`.
  4. For the **Traffic Type**, select `VLAN`.
  5. Click **Add**.
  6. Verify that you see the newly created Transport Zone named `tz-vlan` in the list.
3. To retrieve the host switch name:
  1. Make a call to the NSX API:

```
curl -k -u USER:PASSWORD -X GET "https://${NSX_MANAGER}/api/v1/transport-zones"
```

2. Retrieve the host switch name from the `host_switch_name` property in the return.

For example, `nsxHostSwitch` is the host switch name in the following return:

```
$ curl -k -u user:password -X GET "https://10.20.30.40/api/v1/transport-zones"
{
 "results" : [{
 "transport_type" : "OVERLAY",
 "host_switch_name" : "nsxHostSwitch",
 "host_switch_id" : "5bfdbfc4-c2ab-4ca7-a021-bb1fc1b45ceb",
 "transport_zone_profile_ids" : [{
 "resource_type" : "BfdHealthMonitoringProfile",
 "profile_id" : "52035bb3-ab02-4a08-9884-18631312e50a"
 }],
 "host_switch_mode" : "STANDARD",
 "nested_nsx" : false,
 "is_default" : true,
 "resource_type" : "TransportZone",
 "id" : "1b3a2f36-bfd1-443e-a0f6-4de01abc963e",
 "display_name" : "nsx-overlay-transportzone",
 "_create_user" : "system",
 "_create_time" : 1594850884969,
 "_last_modified_user" : "system",
 "_last_modified_time" : 1594850884969,
 "_system_owned" : false,
 "_protection" : "NOT_PROTECTED",
 "_revision" : 0,
 "_schema" : "/v1/schema/TransportZone"
 }, {
 "transport_type" : "VLAN",
 "host_switch_name" : "nsxHostSwitch",
 "host_switch_id" : "5bfdbfc4-c2ab-4ca7-a021-bb1fc1b45ceb",
 "transport_zone_profile_ids" : [{
 "resource_type" : "BfdHealthMonitoringProfile",
 "profile_id" : "52035bb3-ab02-4a08-9884-18631312e50a"
 }],
 "host_switch_mode" : "STANDARD",
 "nested_nsx" : false,
 "is_default" : true,
 "resource_type" : "TransportZone",
 "id" : "a95c914d-748d-497c-94ab-10d4647daeba",
 "display_name" : "nsx-vlan-transportzone",
 "_create_user" : "system",
 "_create_time" : 1594850885002,
 "_last_modified_user" : "system",
 "_last_modified_time" : 1594850885002,
 "_system_owned" : false,
 "_protection" : "NOT_PROTECTED",
 "_revision" : 0,
 "_schema" : "/v1/schema/TransportZone"
 }],
 "result_count" : 2,
 "sort_by" : "display_name",
 "sort_ascending" : true
}
```

#### 4. Configure NSX for the Edge Nodes:

1. Deploy each Edge Node, configuring NSX with the custom Transport Zones and generated switch names as follows:

- **Switch 1 for Overlay TZ**

| Field                   | Description                                                                                                    |
|-------------------------|----------------------------------------------------------------------------------------------------------------|
| <b>Edge Switch Name</b> | Enter the exact switch name for the custom Overlay Transport Zone retrieved from the API call                  |
| <b>Transport Zone</b>   | <code>tz-overlay</code> , for example (use the exact name you specified for the custom Overlay Transport Zone) |
| <b>Uplink Profile</b>   | <code>nsx-edge-single-nic-uplink-profile</code>                                                                |
| <b>IP Assignment</b>    | Use IP Pool                                                                                                    |
| <b>IP Pool</b>          | TEP-IP-POOL                                                                                                    |
| <b>Uplinks</b>          | uplink-1 / EDGE-VTEP-PG                                                                                        |

2. Select **Add Switch** at the top of the dialog. Configure the VLAN Transport Zone switch as follows:

- **Switch 2 for VLAN TZ**

| Field                   | Description                                                                                              |
|-------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Edge Switch Name</b> | Enter the exact switch name for the custom VLAN Transport Zone retrieved from the API call               |
| <b>Transport Zone</b>   | <code>tz-vlan</code> , for example (use the exact name you specified for the custom VLAN Transport Zone) |
| <b>Uplink Profile</b>   | <code>nsx-edge-single-nic-uplink-profile</code>                                                          |
| <b>Uplinks</b>          | uplink-1 / EDGE-UPLINK-PG                                                                                |

### Option 3: Use the NSX API to Create Custom Transport Zones and NSX Switches

If you want to customize the NSX host switch name, you must do so using the NSX API. The required parameters are `host_switch_name` and `transport_type` (OVERLAY or VLAN). The optional parameters are `description` and `display_name`.

To create custom Transport Zones and NSX Switches:

1. Create a custom Overlay Transport Zone and associated custom NSX switch:

```
curl -k -u USER:PASSWORD -X POST -H 'Content-type: application/json' \
--data-binary '{ "display_name": "tz-overlay", "host_switch_name": "switch-overl" \
ay", "description": "Overlay Transport Zone", "transport_type": "OVERLAY" }' \
https://${NSX_MANAGER}/api/v1/transport-zones
```

Where:

- `USER` is the account name to use to authenticate.

- **PASSWORD** is the password to use to authenticate.
2. Retrieve the property values from the returned responses.

For example:

```
{
 "_revision": 0,
 "id": "c6626083-1a86-4370-85c0-791cf9f947e9",
 "display_name": "tz-overlay",
 "description": "Overlay Transport Zone",
 "resource_type": "TransportZone",
 "transport_type": "OVERLAY",
 "host_switch_name": "switch-overlay",
 "_last_modified_user": "admin",
 "_last_modified_time": 1414179082458,
 "_create_time": 1414179082458,
 "_create_user": "admin",
 "_schema": "/v1/schema/TransportZone"
}
```

3. Create a custom VLAN Transport Zone and associated custom NSX switch:

```
curl -k -u USER:PASSWORD -X POST -H 'Content-type: application/json' \
--data-binary '{ "display_name": "tz-vlan", "host_switch_name": "switch-vlan", "description": "VLAN Transport Zone", "transport_type": "VLAN" }' \
https://${NSX_MANAGER}/api/v1/transport-zones
```

Where:

- **USER** is the account name to use to authenticate.
  - **PASSWORD** is the password to use to authenticate.
4. Retrieve the property values from the returned responses.

For example:

```
{
 "_revision": 0,
 "id": "c6626083-1a86-4370-85c0-791cf9f947e9",
 "display_name": "tz-vlan",
 "description": "VLAN Transport Zone",
 "resource_type": "TransportZone",
 "transport_type": "VLAN",
 "host_switch_name": "switch-vlan",
 "_last_modified_user": "admin",
 "_last_modified_time": 1414179082458,
 "_create_time": 1414179082458,
 "_create_user": "admin",
 "_schema": "/v1/schema/TransportZone"
}
```

5. Use the retrieved values to configure NSX as described in the **Configure NSX for the Edge Node** step in [Option 2: Create Custom Transport Zones and Use the NSX API to Get the Host Switch Names](#) above.

# TKGI NSX Edge Switch and Transport Zone Host Switch Name Requirements

In NSX-T 3.1 and later, the Transport Zone Host Switch Name has been deprecated and removed from the NSX-T configuration UI.

For TKGI, the NSX **Edge Switch Name** and the Transport Zone host switch name must be identical. When configuring NSX-T, configure the **Edge Switch Name** to be the same as the Transport Zone host switch name.



**Note:** The NSX 3.x Edge Node configuration displays the following message beside the **Edge Switch Name** field: “*The switch name value need not be identical to host switch name associated with the Transport Zone.*” This message does not apply to TKGI.

If there is a mismatch between the host switch name associated with the Transport Zone and the **Edge Switch Name**, TKGI installation fails with the following error:

```
Failed to get NSX provisioning properties: No transport zone with overlay type found in transport node as switch name is not same across the TZ and ESXI TN
```

## Configure vSphere Networking for ESXi Hosts

This topic provides instructions for configuring vSphere Networking for ESXi Hosts.

### Prerequisites

Before completing this section, make sure you have completed the following sections:

- [NSX-T Installation Prerequisites](#)
- [Install and Configure the NSX-T Manager Hosts](#)
- [Generate and Register the NSX-T TLS Certificate and Private Key](#)
- [Create an IP Pool for VTEP](#)
- [Configure Transport Zones](#)

## Configure vSphere Networking for ESXi Hosts

In this section of the [NSX-T installation for TKGI](#), you configure the vSphere networking and port groups for ESXi hosts (the vSwitch). If you have created separate vSphere clusters for Management and Compute, perform this operation on each ESXi host in the Management cluster. If you have not created separate vSphere clusters, perform this operation on each ESXi host in the cluster.

The following instructions describe how to configure a vSphere Virtual Standard vSwitch (VSS). For production environments, it is recommended that you configure a Virtual Distributed vSwitch (VDS). You configure the VDS from the vCenter **Networking** tab and then add the ESXi hosts to the VDS. The configuration settings for the VDS are similar to the VSS configuration described below. For instructions on configuring the VDS, see [Create a vSphere Distributed Switch](#) in the vSphere 7

documentation.

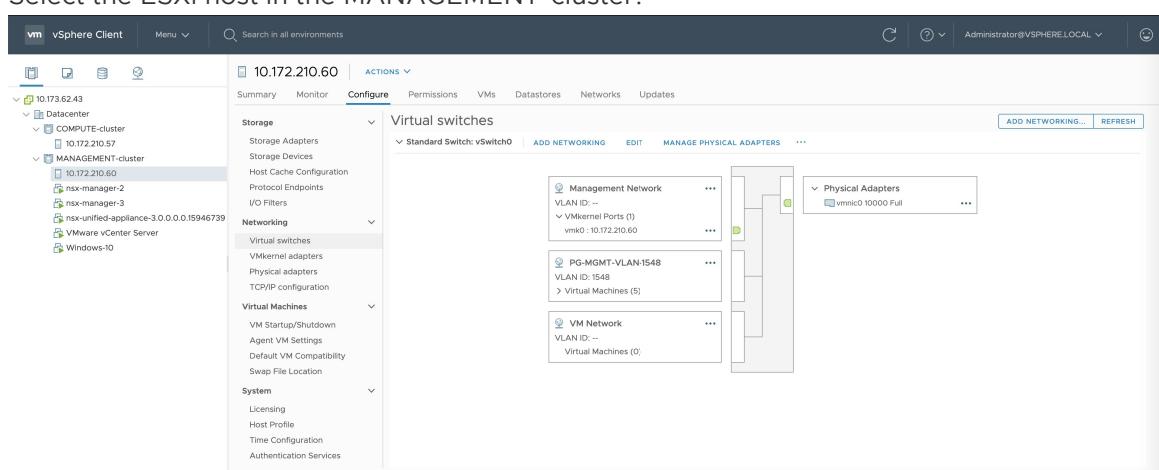
Refer to the [Release Notes](#) for details on TKGI support for vSphere 7 VDS for NSX-T transport node traffic.

## Create vSwitch Port-Groups for Edge Nodes

Create vSwitch Port-Groups for the Edge Nodes on the ESXi hosts in the MANAGEMENT-cluster.

For each ESXi host in the MANAGEMENT-cluster, create the following vSwitch Port Groups:

- EDGE-VTEP-PG: VLAN 3127
- EDGE-UPLINK-PG: VLAN trunk (All (4095))
- Log in to the vCenter Server.
- Select the ESXi host in the MANAGEMENT-cluster.



- Select **Configure > Virtual switches**.
- Select **Add Networking** (upper right).
- Select the option **Virtual Machine Port Group for a Standard Switch** and click **Next**.

## 10.172.210.60 - Add Networking

**1 Select connection type**

**2 Select target device**

**3 Connection settings**

**4 Ready to complete**

**Select connection type**

Select a connection type to create.

**VMkernel Network Adapter**

The VMkernel TCP/IP stack handles traffic for ESXi services such as vSphere vMotion, iSCSI, NFS, FCoE, Fault Tolerance, vSAN and host management.

**Virtual Machine Port Group for a Standard Switch**

A port group handles the virtual machine traffic on standard switch.

**Physical Network Adapter**

A physical network adapter handles the network traffic to other hosts on the network.

CANCEL

BACK

NEXT

- Select the existing standard switch named `vSwitch0` and click **Next**.

## 10.172.210.60 - Add Networking

**1 Select connection type**

**2 Select target device**

**3 Connection settings**

**4 Ready to complete**

**Select target device**

Select a target device for the new connection.

**Select an existing standard switch**

vSwitch0

BROWSE ...

**New standard switch**

MTU (Bytes)

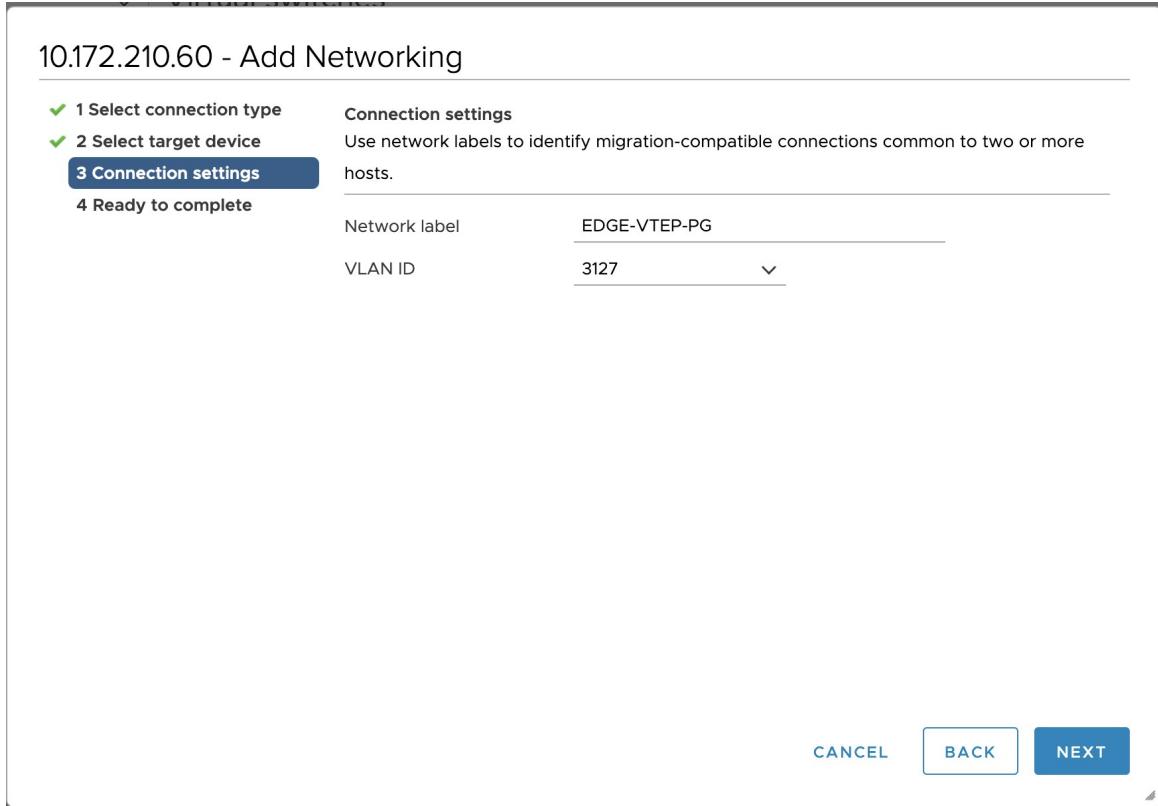
1500

CANCEL

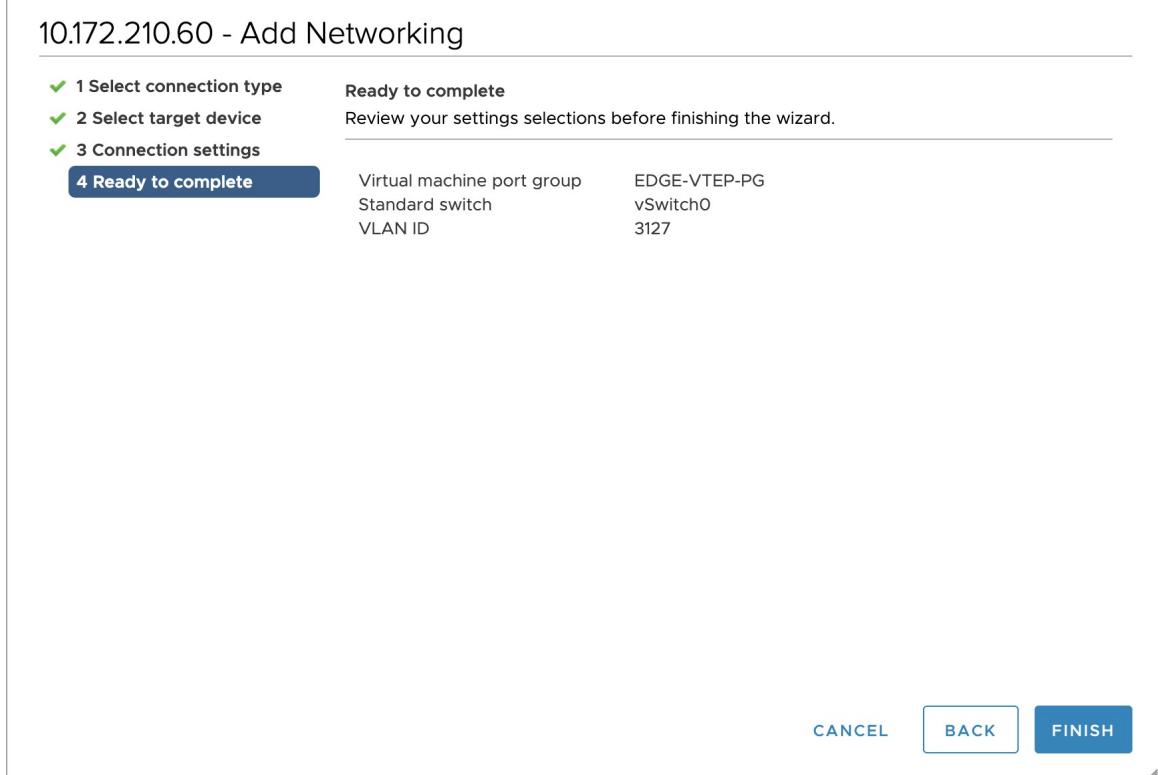
BACK

NEXT

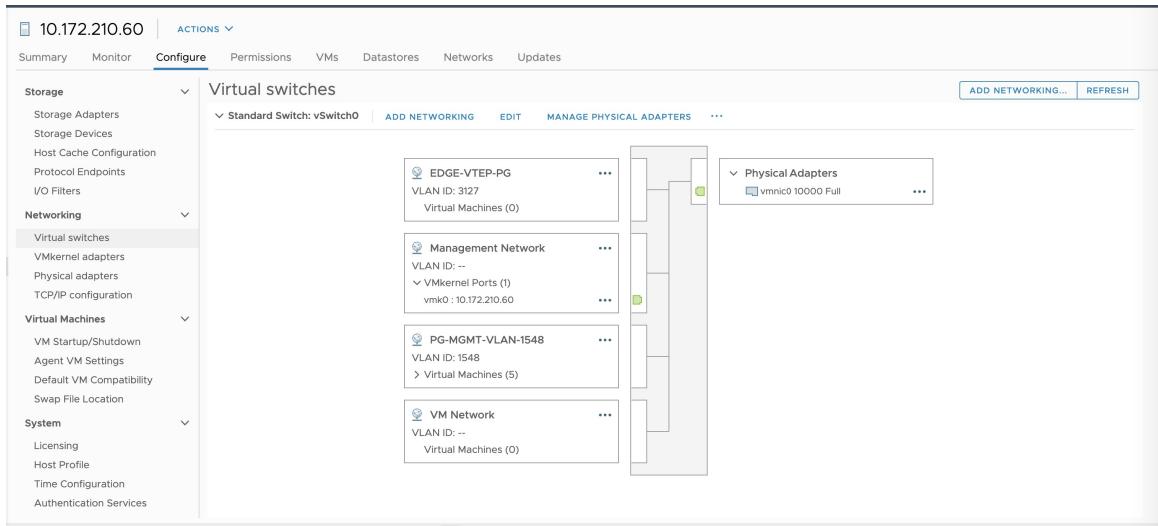
- Enter a **Network Label**, such as `EDGE-VTEP-PG`.
- Enter a **VLAN ID**, such as `3127`.



- Click **Finish**.



- Verify that you see the newly created port group.



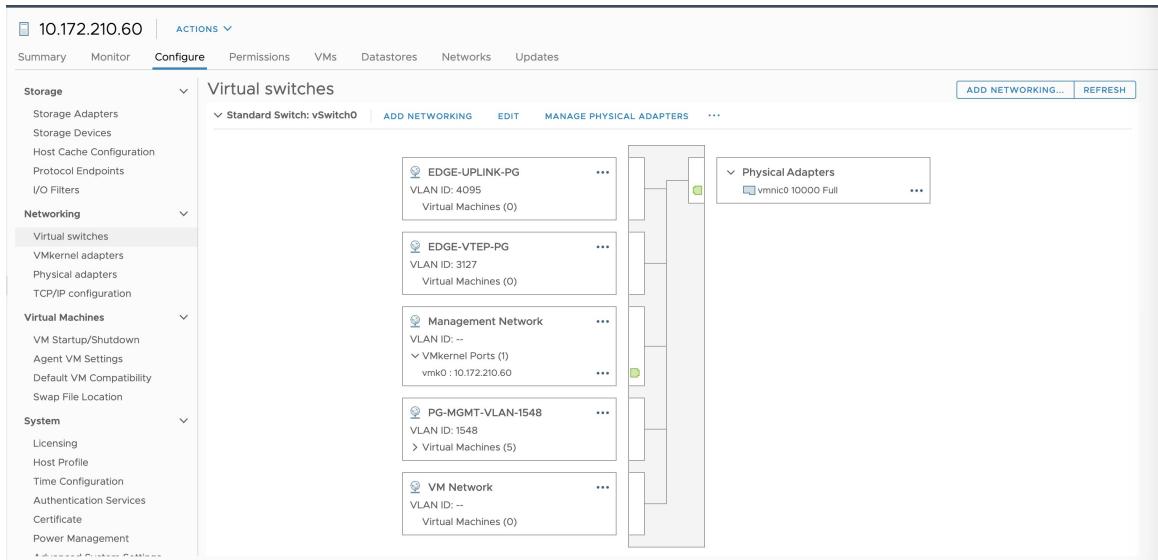
- Select **Add Networking** (upper right).
- Select the option **Virtual Machine Port Group for a Standard Switch** and click **Next**.
- Select the existing standard switch named **vSwitch0** and click **Next**.
- Enter a **Network Label**, such as **EDGE-UPLINK-PG**.
- For the **VLAN ID**, select **All (4095)** from the drop-down.

10.172.210.60 - Add Networking

|                                                                                                                                                                                     |                                                                                                                            |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------|
| <span style="color: green;">✓</span> 1 Select connection type<br><span style="color: green;">✓</span> 2 Select target device<br><b>3 Connection settings</b><br>4 Ready to complete | <b>Connection settings</b><br>Use network labels to identify migration-compatible connections common to two or more hosts. |
|                                                                                                                                                                                     | Network label: <input type="text" value="EDGE-VTEP-PG"/><br>VLAN ID: <input type="text" value="3127"/>                     |

CANCEL BACK NEXT

- Click **Finish**.
- Verify that you see the newly created port group.



## Set vSwitch0 with MTU at 9000

For each ESXi host in the MANAGEMENT-cluster, or each ESXi host in the vCenter cluster if you have not created separate Management and Compute clusters, you must enable the virtual switch with jumbo MTU, that is, set vSwitch0 with MTU=9000. If you do not do this, network overlay traffic will jam. The TEP interface for the NSX-T Edge Nodes must be connected to a port group that supports > 1600 bytes. The default is 1500.

1. Select the Virtual Switch on each ESXi host in the MANAGEMENT-cluster, or each host in the vCenter cluster.
2. Click Edit.
3. For the MTU (bytes) setting, enter **9000**.

**vSwitch0 - Edit Settings**

| Properties           |                 |                                                                             |
|----------------------|-----------------|-----------------------------------------------------------------------------|
| Security             | Number of ports | Elastic                                                                     |
| Traffic shaping      | MTU (Bytes)     | <input type="text" value="9000"/> <span style="color: #0070C0;">9000</span> |
| Teaming and failover |                 |                                                                             |

CANCEL OK

4. Click OK to complete the operation.

## Next Step

[Deploy NSX-T Edge Nodes](#).

## Install and Configure the NSX-T Edge Nodes

This topic provides instructions for installing and configuring NSX-T Data Center v3.0 for use with VMware Tanzu Kubernetes Grid Integrated Edition on vSphere.

## Prerequisites

Before completing this section, make sure you have completed the following sections:

- [NSX-T Installation Prerequisites](#)
- [Install and Configure the NSX-T Manager Hosts](#)
- [Generate and Register the NSX-T TLS Certificate and Private Key](#)
- [Create an IP Pool for VTEP](#)
- [Configure Transport Zones](#)
- [Configure vSphere Networking for ESXi Hosts](#)

## Deploy NSX-T Edge Nodes

In this section you deploy two NSX-T Edge Nodes.

NSX Edge Nodes provide the bridge between the virtual network environment implemented using NSX-T and the physical network. Edge Nodes for Tanzu Kubernetes Grid Integrated Edition run load balancers for TKGI API traffic, Kubernetes load balancer services, and ingress controllers. See [Load Balancers in Tanzu Kubernetes Grid Integrated Edition](#) for more information.

In NSX-T, a load balancer is deployed on the Edge Nodes as a virtual server. The following virtual servers are required for Tanzu Kubernetes Grid Integrated Edition:

- 1 TCP Layer 4 virtual server for each Kubernetes service of type:[LoadBalancer](#)
- 2 Layer 7 global virtual servers for Kubernetes pod ingress resources (HTTP and HTTPS)
- 1 global virtual server for the TKGI API

The number of virtual servers that can be run depends on the size of the load balancer which depends on the size of the Edge Node.

The default size of the load balancer deployed by NSX-T for a Kubernetes cluster is [small](#). Tanzu Kubernetes Grid Integrated Edition supports only the [medium](#), [large](#) and larger VM Edge Node form factors and the bare metal Edge Node. Customize the size of the load balancer using [Network Profiles](#).

For this installation, we use the Large VM form factor for the Edge Node. See [VMware Configuration Maximums](#) for more information.

## Install and Configure Edge Node 1

Deploy the Edge Node 1 VM using the NSX-T Manager interface.

- From your browser, log in with admin privileges to NSX Manager at <https://NSX-MANAGER-IP-ADDRESS>.
- In NSX Manager, go to **System > Fabric > Nodes > Edge Transport Nodes**.

The screenshot shows the NSX-T Manager interface with the following details:

- Header:** NSX-T
- Navigation:** Home, Networking, Security, Inventory, Plan & Troubleshoot, System (selected).
- Left Sidebar:** System Overview, Configuration, Appliances, Get Started, Fabric (selected), Nodes, Profiles.
- Central Area:** Host Transport Nodes, Edge Transport Nodes (selected), Edge Clusters, ESXi Bridge Clusters, NCP Clusters.
- Action Bar:** + ADD EDGE VM, EDIT, DELETE, ACTIONS.
- Table Headers:** Edge, ID, Deployment Type, Management IP, Host, Configuration Stat, NSX Version, N-VDS, Tunnels.

- Click **Add Edge VM**.
- Configure the Edge VM as follows:

- Name:** edge-node-1
- Host name/FQDN:** edge-node-1.lab.com
- Form Factor:** Large

The dialog is titled "Add Edge VM" and contains the following fields:

| Name and Description                                                                                                                                                                                                                                                                                                   |                     |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|
| Name*                                                                                                                                                                                                                                                                                                                  | edge-node-1         |
| Host name/FQDN*                                                                                                                                                                                                                                                                                                        | edge-node-1.lab.com |
| Description                                                                                                                                                                                                                                                                                                            | (Empty)             |
| Form Factor*                                                                                                                                                                                                                                                                                                           |                     |
| <input type="radio"/> Small <input type="radio"/> Medium <input checked="" type="radio"/> Large <input type="radio"/> Extra Large<br>2 vCPU      4 vCPU      8 vCPU      16 vCPU<br>4 GB RAM      8 GB RAM      32 GB RAM      64 GB RAM<br>200 GB Storage      200 GB Storage      200 GB Storage      200 GB Storage |                     |
| <small>&gt; Advanced Resource Reservations</small>                                                                                                                                                                                                                                                                     |                     |
| <small>CANCEL</small> <b>NEXT</b>                                                                                                                                                                                                                                                                                      |                     |

- Configure **Credentials** as follows:
  - CLI User Name:** admin
  - CLI Password:** Enter a strong password for the `admin` user that complies with the NSX-T requirements.
  - Enable SSH Login:** Yes
  - System Root Password:** Enter a strong password for the `root` user that complies with

the NSX-T requirements.

- ◊ **Enable Root SSH Login:** Yes
- ◊ **Audit Credentials:** Enter an `audit` user name and password.

| Add Edge VM                                                                                                       |  | Credentials                                                                                                                                                                                                                                                    |
|-------------------------------------------------------------------------------------------------------------------|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 Name and Description                                                                                            |  | CLI credentials will be set on the NSX Edge VM. These credentials can be used to login to the read only command line interface of the appliance.                                                                                                               |
| 2 Credentials                                                                                                     |  | <b>CLI Credentials</b><br>CLI User Name* <input type="text" value="admin"/><br>CLI Password* <input type="password" value="*****"/><br>CLI Confirm Password* <input type="password" value="*****"/><br>Allow SSH Login <input checked="" type="checkbox"/> Yes |
| 3 Configure Deployment                                                                                            |  | <b>Root Credentials</b><br>System Root Password* <input type="password" value="*****"/><br>System Root Confirm Password* <input type="password" value="*****"/><br>Allow Root SSH Login <input checked="" type="checkbox"/> Yes                                |
| 4 Configure Node Settings                                                                                         |  | > Audit Credentials                                                                                                                                                                                                                                            |
| 5 Configure NSX                                                                                                   |  |                                                                                                                                                                                                                                                                |
| <input type="button" value="CANCEL"/> <input type="button" value="PREVIOUS"/> <input type="button" value="NEXT"/> |  |                                                                                                                                                                                                                                                                |

6. Configure the deployment as follows:

- ◊ **Compute Manager:** vCenter
- ◊ **Cluster:** MANAGEMENT-Cluster
- ◊ **Datastore:** Select the datastore

| Add Edge VM                                                                                                       |  | Configure Deployment                                       |
|-------------------------------------------------------------------------------------------------------------------|--|------------------------------------------------------------|
| 1 Name and Description                                                                                            |  | Compute Manager* <input type="text" value="vCenter-demo"/> |
| 2 Credentials                                                                                                     |  | Cluster* <input type="text" value="MANAGEMENT-cluster"/>   |
| 3 Configure Deployment                                                                                            |  | Resource Pool <input type="text"/>                         |
| 4 Configure Node Settings                                                                                         |  | Host <input type="text"/>                                  |
| 5 Configure NSX                                                                                                   |  | Datastore* <input type="text" value="datastore2"/>         |
| Did not find expected? Try refresh to fetch latest datastores from System.                                        |  |                                                            |
| <input type="button" value="CANCEL"/> <input type="button" value="PREVIOUS"/> <input type="button" value="NEXT"/> |  |                                                            |

7. Configure the node settings as follows:

- ◊ **IP Assignment:** Static
- ◊ **Management IP:** 10.173.62.49/24, for example
- ◊ **Default Gateway:** 10.173.62.253, for example

❖ **Management Interface:** PG-MGMT-VLAN-1548, for example

Add Edge VM

Configure Node Settings

IP Assignment\*  Static  DHCP  
Management IP\*   
Default Gateway

Management Interface\* 

### Configure the N-VDS Switch or Switches for Edge Node 1

The next step is to configure the N-VDS switch and Transport Zones for NSX Edge Node 1. How you do this differs depending on if you are using the default Transport Zones, which requires a single N-VDS switch, or custom Transport Zones, which require multiple N-VDS switches. Refer to [Configuring NSX-T Data Center v3.1 Transport Zones and Edge Node Switches for TKG1](#).

### Complete the Edge Node 1 Installation

1. Click **Finish** to complete the configuration. The installation begins.
2. In vCenter, use the **Recent Tasks** panel at the bottom of the page to verify that you see the Edge Node 1 VM being deployed.
3. Once the process completes, you should see the Edge Node 1 deployed successfully in NSX-T Manager.

| Host Transport Nodes          |                 | Edge Transport Nodes |               | Edge Clusters          |                                              | ESXi Bridge Clusters      |       | NCP Clusters |                  |
|-------------------------------|-----------------|----------------------|---------------|------------------------|----------------------------------------------|---------------------------|-------|--------------|------------------|
|                               |                 |                      |               |                        |                                              |                           |       |              |                  |
| <a href="#">+ ADD EDGE VM</a> |                 | <a href="#">EDIT</a> |               | <a href="#">DELETE</a> |                                              | <a href="#">ACTIONS</a> ▾ |       |              |                  |
| Edge                          | ID              | Deployment Type      | Management IP | Host                   | Configuration State                          | NSX Version               | N-VDS | Tunnels      | TEP IP Addresses |
| Edge Cluster                  | Logical Routers | Node Status          | Alarms        |                        |                                              |                           |       |              |                  |
| <a href="#">edge-node-1</a>   | 9099..ba72      | Virtual Machi...     | 10.173.62.49  |                        | <span style="color: green;">● Success</span> | 3.0.0.0.0.159...          | 2     | Not Availab  | 192.23.213.1     |
|                               |                 |                      |               |                        |                                              |                           |       |              |                  |

4. Click the N-VDS link and verify that you see the switch or switches.

The screenshot shows the NSX Manager interface under the 'System' tab. In the 'Edge Transport Nodes' section, a modal window titled 'Switches: edge-node-1' is open. It lists two entries:

| Edge Switch Name | Type  | Uplink Profile     | Transport Zones |
|------------------|-------|--------------------|-----------------|
| switch-overlay   | N-VDS | nsx-edge-single... | tz-overlay      |
| switch-vlan      | N-VDS | nsx-edge-single... | tz-vlan         |

- In vCenter verify that the Edge Node is created.

The screenshot shows the vSphere Client interface. The inventory tree on the left shows a hierarchy of datacenters, clusters, and hosts. A specific host named 'edge-node-1' is selected. The main pane displays the 'Summary' tab for this host, showing the following details:

- Guest OS:** Ubuntu Linux (64-bit)
- Compatibility:** ESXi 6.5 and later (VM version 13)
- VMware Tools:** Not running, version:10304 (Guest Managed)
- DNS Name:**
- IP Addresses:**
- Host:** 10.172.210.60

The 'VM Hardware' section lists the following components:

|                   |                                                                                                             |
|-------------------|-------------------------------------------------------------------------------------------------------------|
| CPU               | 8 CPU(s)                                                                                                    |
| Memory            | 32 GB, 24 GB memory active                                                                                  |
| Hard disk 1       | 200 GB                                                                                                      |
| Network adapter 1 | PG-MGMT-VLAN-1548 (connected)                                                                               |
| Network adapter 2 | EDGE-VTEP-PG (connected)                                                                                    |
| Network adapter 3 | EDGE-UPLINK-PG (connected)                                                                                  |
| Network adapter 4 | none (disconnected)                                                                                         |
| Video card        | 4 MB                                                                                                        |
| VMCI device       | Device on the virtual machine PCI bus that provides support for the virtual machine communication interface |

## Install and Configure Edge Node 2

Deploy the Edge Node 2 VM using the NSX-T Manager interface.

- In NSX Manager, go to **System > Fabric > Nodes > Edge Transport Nodes**.
- Click **Add Edge VM**.
- Configure the Edge VM as follows:
  - Name:** `edge-node-2`
  - Host name/FQDN:** `edge-node-2.lab.com`
  - Form Factor:** Large
- Configure **Credentials** as follows:
  - CLI User Name:** `admin`

- **CLI Password:** Enter a strong password for the `admin` user that complies with the NSX-T requirements.
- **Enable SSH Login:** Yes
- **System Root Password:** Enter a strong password for the `root` user that complies with the NSX-T requirements.
- **Enable Root SSH Login:** Yes
- **Audit Credentials:** Enter an `audit` user name and password.

5. Configure the deployment as follows:

- **Compute Manager:** vCenter
- **Cluster:** MANAGEMENT-Cluster
- **Datastore:** Select the datastore

6. Configure the node settings as follows:

- **IP Assignment:** Static
- **Management IP:** 10.173.62.58/24, for example
- **Default Gateway:** 10.173.62.253, , for example
- **Management Interface:** PG-MGMT-VLAN-1548, for example

## Configure the N-VDS Switch or Switches for Edge Node 2

The next step is to configure the N-VDS switch and transport zones for NSX Edge Node 2. How you do this differs depending on if you are using the default Transport Zones, which requires a single N-VDS switch, or custom Transport Zones, which require multiple N-VDS switches. Refer to [Configuring NSX-T Data Center v3.1 Transport Zones and Edge Node Switches for TKG1](#).

## Complete the Installation of Edge Node 2

1. Click **Finish** to complete the configuration. The installation begins.
2. In vCenter, use the **Recent Tasks** panel at the bottom of the page to verify that you see the Edge Node 1 VM being deployed.
3. Once the process completes, you should see the Edge Node 2 deployed successfully in NSX-T Manager.
4. Click the N-VDS link and verify that you see the N-VDS switch or switches.
5. In vCenter verify that Edge Node 2 is created.
6. In NSX Manager, verify that you see both Edge Nodes.

| Host Transport Nodes |             | Edge Transport Nodes |               | Edge Clusters |                    | ESXi Bridge Clusters |       | NCP Clusters  |                  | View         |                 | All         |        |
|----------------------|-------------|----------------------|---------------|---------------|--------------------|----------------------|-------|---------------|------------------|--------------|-----------------|-------------|--------|
|                      |             |                      |               |               |                    |                      |       |               |                  |              |                 |             |        |
| + ADD EDGE VM        |             | EDIT                 |               | DELETE        |                    | ACTIONS              |       |               |                  |              |                 |             |        |
| Edge                 | ID          | Deployment Type      | Management IP | Host          | Configuration Stat | NSX Version          | N-VDS | Tunnels       | TEP IP Addresses | Edge Cluster | Logical Routers | Node Status | Alarms |
| edge-node-1          | 9099...ba72 | Virtual Machine      | 10.173.62.49  |               | Success            | 3.0.0.0.1594..       | 2     | Not Available | 192.23.213.1     |              | 0               | Up          | 0      |
| edge-node-2          | 84ea...687e | Virtual Machine      | 10.173.62.58  |               | Success            | 3.0.0.0.1594..       | 2     | Not Available | 192.23.213.2     |              | 0               | Up          | 0      |

## Next Steps

Deploy NSX-T Transport Nodes.

## Installing and Configuring NSX-T Transport Nodes

This topic provides instructions for installing and configuring NSX-T Data Center v3.0 for use with VMware Tanzu Kubernetes Grid Integrated Edition on vSphere.

### Prerequisites

Before completing this section, make sure you have completed the following sections:

- [NSX-T Installation Prerequisites](#)
- [Install and Configure the NSX-T Manager Hosts](#)
- [Generate and Register the NSX-T TLS Certificate and Private Key](#)
- [Create an IP Pool for VTEP](#)
- [Configure Transport Zones](#)
- [Configure vSphere Networking for ESXi Hosts](#)
- [Deploy NSX-T Edge Nodes](#)

### Prerequisites for Installing NSX-T Data Center v3.0 for Tanzu Kubernetes Grid Integrated Edition

To perform a new installation of NSX-T Data Center for Tanzu Kubernetes Grid Integrated Edition, complete the following steps in the order presented.

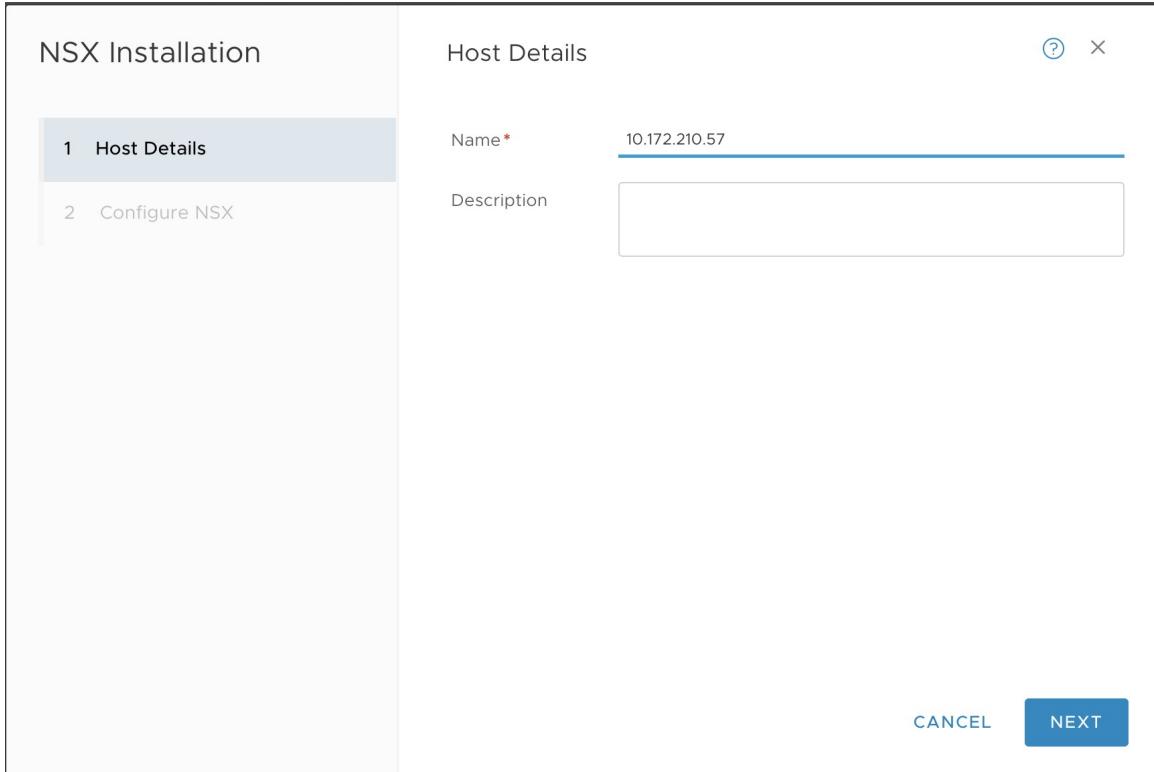
## Deploy ESXi Host Transport Nodes Using VDS

Deploy each ESXi host in the COMPUTE-cluster as an ESXi host transport node (TN) in NSX-T. If you have not created a separate COMPUTE-cluster for ESXi hosts, deploy each ESXi host in the vSphere cluster as a host transport node in NSX-T.

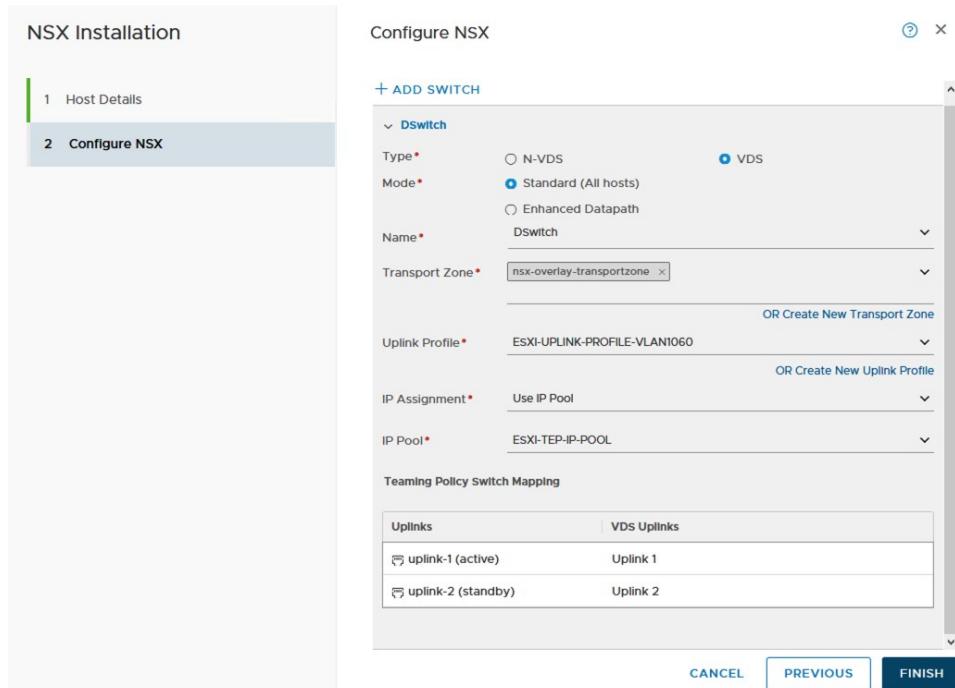
1. Go to **System > Fabric > Nodes > Host Transport Nodes**.

| Node                   | ID              | IP Addresses            | QoS Type   | NSX Configuration | NSX Version | Host Switches | Tunnels       | TEP IP Addresses | Node Status           | Alarms |
|------------------------|-----------------|-------------------------|------------|-------------------|-------------|---------------|---------------|------------------|-----------------------|--------|
| MANAGEMENT-cluster (1) | MoRef ID: do... |                         |            |                   |             |               |               |                  | 1 Host Not Configured |        |
| COMPUTE-cluster (1)    | MoRef ID: do... |                         |            |                   |             |               |               |                  | 1 Host Not Configured |        |
| 10.172.210.57          | 0d04...t15      | 10.172.210.57, fd01:... | ESXi 7.0.0 | Not Configured    |             | 0             | Not Available |                  | Not Available         | 0      |

2. Expand the Compute Manager and select the ESXi host in the COMPUTE-cluster, or each ESXi host in the vSphere cluster.



3. Click **Configure NSX**.
4. In the **Host Details** tab, enter a name, such as `10.172.210.57`.
5. In the **Configure NSX** tab, configure the transport node as follows:
  - ◊ **Type:** `vDS` (do not select the N-VDS option)
  - ◊ **Name:** `switch-overlay` (you must use the same switch name that was configured for `tz-overlay` transport zone)
  - ◊ **Transport Zone:** `tz-overlay`
  - ◊ **NIOC Profile:** `nsx-default-nioc-hostswitch-profile`
  - ◊ **Uplink Profile:** `nsx-esxi-uplink-hostswitch-profile`
  - ◊ **LLDP Profile:** `LLDP [Send Packet Disabled]`
  - ◊ **IP Assignment:** `Use IP Pool`
  - ◊ **IP Pool:** `TEP-IP-POOL`
  - ◊ **Teaming Policy Switch Mapping**
    - **Uplinks:** `uplink-1`
    - **Physical NICs:** `vmnic1`



6. Click **Finish**.
7. Verify that the host TN is configured.

| Host Transport Nodes                         |                       | Edge Transport Nodes     | Edge Clusters | ESXi Bridge Clusters | NCP Clusters      |             |                              |              |                     |                                      |        |
|----------------------------------------------|-----------------------|--------------------------|---------------|----------------------|-------------------|-------------|------------------------------|--------------|---------------------|--------------------------------------|--------|
| Managed by                                   | vCenter-demo          | ID                       | IP Addresses  | OS Type              | NSX Configuration | NSX Version | Host Switches                | Tunnels      | TEP IP Addresses    | Node Status                          | Alarms |
| <input type="checkbox"/> Node                | MANAGEMENT-cluster .. | MoRef ID: do...          |               |                      |                   |             |                              |              |                     | <span>● 1 Host Not Configured</span> |        |
| <input type="checkbox"/> COMPUTE-cluster (1) | MoRef ID: do...       |                          |               |                      |                   |             |                              |              |                     | <span>● 1 Host Up ⓘ</span>           |        |
| <input type="checkbox"/> 10.172.210.57       | 9cc5...7af2           | 10.172.210.57, fd01:1... | ESXi 7.0.0    | <span>Success</span> | 3.0.0.0.159...    | 1           | <span>Not Available ⓘ</span> | 192.23.213.3 | <span>● Up ⓘ</span> | 0                                    | —      |

## Verify TEP to TEP Connectivity

To avoid any overlay communication in the future due to MTU issue, test TEP to TEP connectivity and verify that it is working.

1. SSH to edge-node-1 and get the local TEP IP address, such as `192.23.213.1`. Use the command `get vteps` to get the IP.
2. SSH to edge-node-2 and get the local TEP IP address, ushc as `192.23.213.2`. Use the command `get vteps` to get the IP.
3. SSH to the ESXi host and get the TEP IP address, such as `192.23.213.3`. Use the command `esxcfg-vmknic -l` to get the IP. The interface will be `vmk10` and the NetStack will be `vxlan`.
4. From each ESXi transport node, test the connections to each NSX-T Edge Node, for example:

```
vmkping ++netstack=vxlan 192.23.213.1 -d -s 1572 -I vmk10: OK
vmkping ++netstack=vxlan 192.23.213.2 -d -s 1572 -I vmk10: OK
```

1. Test the connection from NSX-T edge node 1 and edge node 2 to ESXi TN:

```
> vrf 0
> ping 192.23.213.1 size 1572 dfbit enable: OK
```

2. Test the connection from NSX-T edge node 1 to NSX-T edge node 2:

```
> vrf 0
> ping 192.23.213.2 size 1572 dfbit enable: OK
```

## Create NSX-T Edge Cluster

1. Go to System > Fabric > Nodes > Edge Clusters.

2. Click Add.

- ◊ Enter a name, such as `edge-cluster-1`.
- ◊ Add members, including `edge-node-1` and `edge-node-2`.

**Add Edge Cluster**

Name\*

Description

Edge Cluster Profile  ✖️ ▾

**Transport Nodes**

Member Type  ▼

|                                                                                                                                                                                                          |                                                                                                                                                                                                          |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <input type="checkbox"/> Available (0)                                                                                                                                                                   | <input type="checkbox"/> Selected (2)                                                                                                                                                                    |
| <div style="text-align: center; margin-top: 10px;">No records found</div>                                                                                                                                | <div style="text-align: center; margin-top: 10px;">edge-node-2<br/>edge-node-1</div>                                                                                                                     |
| <span style="border: 1px solid #ccc; padding: 2px;">◀ BACK</span> <span style="border: 1px solid #ccc; padding: 2px;">NEXT ▶</span> <span style="border: 1px solid #ccc; padding: 2px;">No record</span> | <span style="border: 1px solid #ccc; padding: 2px;">◀ BACK</span> <span style="border: 1px solid #ccc; padding: 2px;">NEXT ▶</span> <span style="border: 1px solid #ccc; padding: 2px;">No record</span> |

CANCEL
ADD

3. Click **Add**.

4. Verify.

| Edge Clusters            |                |             |             |                                            |                      |
|--------------------------|----------------|-------------|-------------|--------------------------------------------|----------------------|
|                          |                | ID          | Member Type | Cluster Profile                            | Edge Transport Nodes |
| <input type="checkbox"/> | Edge Cluster   | 2518...Se15 | Edge Node   | nsx-default-edge-high-availability-profile | 2                    |
| <input type="checkbox"/> | edge-cluster-1 |             |             |                                            |                      |

## Create Uplink Logical Switch

Create an uplink Logical Switch to be used for the Tier-0 Router.

1. At upper-right, select the **Manager** tab.
2. Go to **Networking > Logical Switches**.

3. Click **Add**.
4. Configure the new logical switch as follows:

- ◊ Name: **LS-T0-uplink**
- ◊ Transport Zone: **tz-vlan**
- ◊ VLAN: **1548**

**Add New Logical Switch** (?) X

---

**General**    [Switching Profiles](#)

---

|                                                                                                                                                                                                                               |                                                                                                                  |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------|
| Name *                                                                                                                                                                                                                        | LS-T0-uplink                                                                                                     |
| Description                                                                                                                                                                                                                   | <input type="text"/>                                                                                             |
| Transport Zone *                                                                                                                                                                                                              | tz-vlan                                                                                                          |
| Uplink Teaming Policy Name *                                                                                                                                                                                                  | [Use Default]                                                                                                    |
| Admin Status                                                                                                                                                                                                                  | <input checked="" type="checkbox"/> Up                                                                           |
| Replication Mode                                                                                                                                                                                                              | <input checked="" type="radio"/> Hierarchical Two-Tier replication<br><input type="radio"/> Head End replication |
| VLAN *                                                                                                                                                                                                                        | <input type="text" value="1548"/> <span style="border: 1px solid #ccc; padding: 2px 5px;">X</span>               |
| VLAN Id or VLAN Trunk Spec is allowed.                                                                                                                                                                                        |                                                                                                                  |
| <span style="border: 1px solid #ccc; padding: 5px 10px; margin-right: 10px;">CANCEL</span> <span style="background-color: #0070C0; color: white; border: 1px solid #0070C0; padding: 5px 10px; font-weight: bold;">ADD</span> |                                                                                                                  |

5. Click **Add**.
6. Verify.

| Logical Switch | ID          | Admin Status | Logical Ports | Traffic Type | Config State | Transport Zone |
|----------------|-------------|--------------|---------------|--------------|--------------|----------------|
| LS-T0-uplink   | f70b...3a68 | Up           | 0             | VLAN : 1548  | Pending      | tz-vlan        |

## Create Tier-0 Router

- Select **Networking** from the **Manager** tab.

| Limit                                   | Current Inventory | Maximum Capacity | Minimum Capacity Threshold | Maximum Capacity Threshold |      |
|-----------------------------------------|-------------------|------------------|----------------------------|----------------------------|------|
| Logical Switches                        | 0                 | 0%               | 10,000                     | 70%                        | 100% |
| System-wide Logical Switch Ports        | 0                 | 0%               | 25,000                     | 70%                        | 100% |
| Tier-0 Logical Routers                  | 0                 | 0%               | 160                        | 70%                        | 100% |
| Tier-1 Logical Routers                  | 0                 | 0%               | 4,000                      | 70%                        | 100% |
| Prefix-lists                            | 0                 | 0%               | 500                        | 70%                        | 100% |
| System-wide NAT rules                   | 0                 | 0%               | 25,000                     | 70%                        | 100% |
| DHCP Server Instances                   | 0                 | 0%               | 10,000                     | 70%                        | 100% |
| System-wide DHCP Range / Pools          | 0                 | 0%               | 10,000                     | 70%                        | 100% |
| Tier-1 Logical Routers with NAT Enabled | 0                 | 0%               | 4,000                      | 70%                        | 100% |

- Select **Tier-0 Logical Router**.

| Logical Router | ID | High Availability Mode |
|----------------|----|------------------------|
|                |    |                        |

- Click **Add**.

- Configure the new Tier-0 Router as follows:

- **Name:** T0-router
- **Edge Cluster:** edge-cluster-1
- **HA mode:** Either Active-Active or Active-Standby
- **Failover mode:** Non-Preemptive

 **Note:** Configuring **Failover mode** is optional if **HA mode** is configured as **Active-Active**. For more information on NSX-T HA mode configuration, see [Add a Tier-0 Gateway](#) in the VMware NSX-T Data Center documentation.

New Tier-O Router

[?](#) [X](#)

Tier-O Router Advanced

|                        |                                                                                  |
|------------------------|----------------------------------------------------------------------------------|
| Name *                 | TO-router                                                                        |
| Description            |                                                                                  |
| Edge Cluster           | edge-cluster-1 <a href="#">X</a> <a href="#">▼</a>                               |
| High Availability Mode | <input type="radio"/> Active-Active <a href="#">OR Create a New Edge Cluster</a> |
| Failover Mode          | <input checked="" type="radio"/> Active-Standby                                  |
|                        | <input checked="" type="radio"/> Preemptive                                      |
|                        | <input checked="" type="radio"/> Non-Preemptive                                  |

[CANCEL](#) [ADD](#)

5. Click **Save** and verify.

| Tier-O Logical Routers   |                |             |                        |                |
|--------------------------|----------------|-------------|------------------------|----------------|
|                          |                | ID          | High Availability Mode | Transport Zone |
|                          |                | ID          | High Availability Mode | Transport Zone |
| <input type="checkbox"/> | Logical Router | 02b8...a3c6 | Active-Standby         | edge-cluster-1 |
| <input type="checkbox"/> | TO-router      |             |                        |                |

6. Select the TO router.

| Tier-O Logical Routers                                                                                                                                                                                                                                        |                  |                |                 |              |                 |              |     |  |  |  |  |  |  |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|----------------|-----------------|--------------|-----------------|--------------|-----|--|--|--|--|--|--|
| <span style="font-size: 1.5em;">+</span> <span style="font-size: 1.5em;">✎</span> <span style="font-size: 1.5em;">ⓧ</span> <span style="font-size: 1.5em;">ⓧ</span> <span style="font-size: 1.5em;">⚙️</span> <span style="font-size: 1.5em;">▼</span>        |                  |                |                 |              |                 |              |     |  |  |  |  |  |  |
| <input checked="" type="checkbox"/>                                                                                                                                                                                                                           | Logical Router ↑ |                |                 |              |                 |              |     |  |  |  |  |  |  |
| <input checked="" type="checkbox"/>                                                                                                                                                                                                                           | TO-router >      |                |                 |              |                 |              |     |  |  |  |  |  |  |
| <a href="#">TO-router</a>                                                                                                                                                                                                                                     |                  |                |                 |              |                 |              |     |  |  |  |  |  |  |
| <a href="#">Overview</a> <a href="#">Configuration</a> <a href="#">Routing</a> <a href="#">Services</a>                                                                                                                                                       |                  |                |                 |              |                 |              |     |  |  |  |  |  |  |
| <b>Logical Router Ports</b>                                                                                                                                                                                                                                   |                  |                |                 |              |                 |              |     |  |  |  |  |  |  |
| <span style="font-size: 1.5em;">+</span> <span style="font-size: 1.5em;">✎</span> <span style="font-size: 1.5em;">ⓧ</span> <span style="font-size: 1.5em;">ⓧ</span> <span style="font-size: 1.5em;">⚙️</span> <span style="font-size: 1.5em;">▼</span>        |                  |                |                 |              |                 |              |     |  |  |  |  |  |  |
| <table border="1"> <thead> <tr> <th>Logical Router</th> <th>ID</th> <th>Type</th> <th>IP Address/mask</th> <th>Connected To</th> <th>Tra</th> </tr> </thead> <tbody> <tr> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table> |                  | Logical Router | ID              | Type         | IP Address/mask | Connected To | Tra |  |  |  |  |  |  |
| Logical Router                                                                                                                                                                                                                                                | ID               | Type           | IP Address/mask | Connected To | Tra             |              |     |  |  |  |  |  |  |
|                                                                                                                                                                                                                                                               |                  |                |                 |              |                 |              |     |  |  |  |  |  |  |

7. Select **Configuration > Router Ports**.

8. Click **Add**.

9. Configure a new router port as follows:

- ◊ **Name:** TO-uplink-1
- ◊ **Type:** uplink
- ◊ **Transport Node:** edge-node-1
- ◊ **Logical Switch:** LS-TO-uplink
- ◊ **Logical Switch Port:** Attach to a new switch port
- ◊ **Subnet:** 10.173.62.50 / 24

New Router Port

| Name*                                                                                                                                                                                                                                                                                                                             | TO-uplink-1                                                                                                     |             |                |              |    |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|-------------|----------------|--------------|----|
| Description                                                                                                                                                                                                                                                                                                                       |                                                                                                                 |             |                |              |    |
| Type                                                                                                                                                                                                                                                                                                                              | Uplink                                                                                                          | MTU         |                |              |    |
| Transport Node*                                                                                                                                                                                                                                                                                                                   | edge-node-1                                                                                                     |             |                |              |    |
| URPF Mode                                                                                                                                                                                                                                                                                                                         | <input checked="" type="radio"/> Strict <input type="radio"/> None                                              |             |                |              |    |
| Logical Switch                                                                                                                                                                                                                                                                                                                    | LS-TO-uplink                                                                                                    |             |                |              |    |
| OR Create a New Switch                                                                                                                                                                                                                                                                                                            |                                                                                                                 |             |                |              |    |
| Logical Switch Port                                                                                                                                                                                                                                                                                                               | <input checked="" type="radio"/> Attach to new switch port <input type="radio"/> Attach to existing switch port |             |                |              |    |
| Switch Port Name                                                                                                                                                                                                                                                                                                                  |                                                                                                                 |             |                |              |    |
| <b>Subnets</b>                                                                                                                                                                                                                                                                                                                    |                                                                                                                 |             |                |              |    |
| <input type="button" value="+ ADD"/> <input type="button" value="DELETE"/> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th style="background-color: #f2f2f2;">IP Address*</th> <th style="background-color: #f2f2f2;">Prefix Length*</th> </tr> <tr> <td>10.173.62.50</td> <td>24</td> </tr> </table> |                                                                                                                 | IP Address* | Prefix Length* | 10.173.62.50 | 24 |
| IP Address*                                                                                                                                                                                                                                                                                                                       | Prefix Length*                                                                                                  |             |                |              |    |
| 10.173.62.50                                                                                                                                                                                                                                                                                                                      | 24                                                                                                              |             |                |              |    |
| <input type="button" value="CANCEL"/> <input type="button" value="ADD"/>                                                                                                                                                                                                                                                          |                                                                                                                 |             |                |              |    |

10. Click **Add** and verify.

| Logical Router ID | Type   | IP Address/mask | Connected To | Transport Node | Relay Service | Statistics |
|-------------------|--------|-----------------|--------------|----------------|---------------|------------|
| TO-uplink-1       | Uplink | 10.173.62.50/24 | LS-TO-uplink | edge-node-1    |               |            |

11. Select the TO router.
12. Select **Configuration > Router Ports**.
13. Add a second uplink by creating a second router port for edge-node-2:

- ◊ **Name:** TO-uplink-1
- ◊ **Type:** uplink
- ◊ **Transport Node:** edge-node-2

- ◊ **Logical Switch:** LS-T0-uplink
- ◊ **Logical Switch Port:** Attach to a new switch port
- ◊ **Subnet:** 10.173.62.51 / 24

14. Once completed, verify that you have two connected router ports.

| Logical Router ID | Type   | IP Address/mask | Connected To                                | Transport Node | Relay Service | Statistics           |
|-------------------|--------|-----------------|---------------------------------------------|----------------|---------------|----------------------|
| TO-uplink-1       | Uplink | 10.173.62.50/24 | ↳ LS-T0-uplink<br>( feb31e09-811e-482c... ) | edge-node-1    |               | <a href="#">View</a> |
| TO-uplink-2       | Uplink | 10.173.62.51/24 | ↳ LS-T0-uplink<br>( 4e99912a-e75c-47c... )  | edge-node-2    |               | <a href="#">View</a> |

## Next Steps

Create NSX-T Objects for Kubernetes Clusters Provisioned by TKGI.

## Create the VMware NSX Objects for Kubernetes Clusters Provisioned by TKGI

This topic provides instructions for creating the NSX-T objects for the Tanzu Kubernetes Grid Integrated Edition control plane where Kubernetes clusters run.

## Prerequisites

Before completing this section, make sure you have completed the following sections:

- [NSX-T Installation Prerequisites](#)
- [Install and Configure the NSX-T Manager Hosts](#)
- [Generate and Register the NSX-T TLS Certificate and Private Key](#)
- [Create an IP Pool for VTEP](#)
- [Configure Transport Zones](#)
- [Configure vSphere Networking for ESXi Hosts](#)
- [Deploy NSX-T Edge Nodes](#)
- [Deploy NSX-T Transport Nodes](#)

## Required NSX-T Objects for the Tanzu Kubernetes Grid Integrated Edition Control Plane

To install TKGI on vSphere with NSX-T, you need to create the following NSX-T objects:

- Tier-0 Gateway (also known as a Router)
- Pods IP Block
- Nodes IP Block

- Floating IP Pool

For more information, see [Network Planning for TKGI](#).

When you configure the TKGI tile, you specify the object IDs:

UGFsbyBBbHRvMB4XDTlxMDUyNDA5MTgzN1oXDTlyMDUyNDA5MTgzN1owczEkMCIG

Plan 13

Kubernetes Cloud Provider

Networking

UAA

Host Monitoring

In-Cluster Monitoring

Tanzu Mission Control

CEIP and Telemetry

Storage

Errands

Resource Config

Disable SSL certificate verification

NAT mode

Policy API mode

Pods IP Block ID \*

cluster-0-pod-network-block-0

Nodes IP Block ID \*

nodes-network-block-0

T0 Router ID \*

t0-shared

Floating IP Pool ID \*

cluster-0-pool-0

Nodes DNS \*

192.168.115.1

vSphere Cluster Names \*

cluster-0

Kubernetes Service Network CIDR Range \*

10.150.0.0/16

HTTP/HTTPS Proxy (for vSphere and AWS only)\*

The following instructions describe how to create these objects. You will need to create these objects using the appropriate interface. If you are using the Management API, create them using the Manager tab. If you are using the Policy API, create these objects using the Policy tab.

## Create NSX-T Objects for Kubernetes Clusters Using the Management Interface

This section provides instructions for creating the required NSX-T objects for Kubernetes clusters using the Management interface.

### Create Tier-0 Router Using the Management Interface

1. Log in to the NSX-T Manager and select the **Networking** tab.
2. Verify that the **Manager** interface is selected. If not, select it.

| Limit                                   | Current Inventory | Maximum Capacity | Minimum Capacity Threshold | Maximum Capacity Threshold |      |
|-----------------------------------------|-------------------|------------------|----------------------------|----------------------------|------|
| Logical Switches                        | 0                 | 0%               | 10,000                     | 70%                        | 100% |
| System-wide Logical Switch Ports        | 0                 | 0%               | 25,000                     | 70%                        | 100% |
| Tier-0 Logical Routers                  | 0                 | 0%               | 160                        | 70%                        | 100% |
| Tier-1 Logical Routers                  | 0                 | 0%               | 4,000                      | 70%                        | 100% |
| prefix-lists                            | 0                 | 0%               | 500                        | 70%                        | 100% |
| System-wide NAT rules                   | 0                 | 0%               | 25,000                     | 70%                        | 100% |
| DHCP Server Instances                   | 0                 | 0%               | 10,000                     | 70%                        | 100% |
| System-wide DHCP Ranges / Pools         | 0                 | 0%               | 10,000                     | 70%                        | 100% |
| Tier-1 Logical Routers with NAT Enabled | 0                 | 0%               | 4,000                      | 70%                        | 100% |

Maximum capacity displayed above is for large appliance only.

### 3. Select Tier-0 Logical Routers.

#### 4. Click Add.

#### 5. Configure a new Tier-0 Router as follows:

- **Name:** T0-router
- **Edge Cluster:** edge-cluster-1
- **HA mode:** Either Active-Active OR Active-Standby
- **Failover mode:** Non-Preemptive



**Note:** Configuring **Failover mode** is optional if **HA mode** is configured as **Active-Active**. For more information on NSX-T HA mode configuration, see [Add a Tier-0 Gateway](#) in the VMware NSX-T Data Center documentation.

## New Tier-O Router

[?](#) [X](#)

Tier-O Router [Advanced](#)

|                                            |                                                                                                    |
|--------------------------------------------|----------------------------------------------------------------------------------------------------|
| Name *                                     | TO-router                                                                                          |
| Description                                |                                                                                                    |
| Edge Cluster                               | edge-cluster-1 <a href="#">x</a> <a href="#">▼</a>                                                 |
| High Availability Mode                     | <input type="radio"/> Active-Active <a href="#">OR Create a New Edge Cluster</a>                   |
| Failover Mode                              | <input checked="" type="radio"/> Active-Standby<br><input checked="" type="radio"/> Non-Preemptive |
| <a href="#">CANCEL</a> <a href="#">ADD</a> |                                                                                                    |

6. Click **Save** and verify.

| Tier-O Logical Routers                                                                                                |             |                        |                        |                |
|-----------------------------------------------------------------------------------------------------------------------|-------------|------------------------|------------------------|----------------|
| <a href="#">+ ADD</a> <a href="#">EDIT</a> <a href="#">DELETE</a> <a href="#">ACTIONS</a> <input type="text"/> Search |             | ID                     | High Availability Mode | Transport Zone |
| Logical Router                                                                                                        | ID          | High Availability Mode | Transport Zone         | Edge Cluster   |
| TO-router                                                                                                             | 02b8...a3c6 | Active-Standby         |                        | edge-cluster-1 |

7. Select the TO-router you created.

| Tier-O Logical Routers                                                                                                                                                                                                                                        |                           |                |                 |              |                 |              |     |  |  |  |  |  |  |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------|----------------|-----------------|--------------|-----------------|--------------|-----|--|--|--|--|--|--|
| <a href="#">+ ADD</a> <a href="#">EDIT</a> <a href="#">DELETE</a> <a href="#">ACTIONS</a>                                                                                                                                                                     |                           |                |                 |              |                 |              |     |  |  |  |  |  |  |
| <input checked="" type="checkbox"/> Logical Router                                                                                                                                                                                                            | <a href="#">TO-router</a> |                |                 |              |                 |              |     |  |  |  |  |  |  |
| <input checked="" type="checkbox"/> TO-router                                                                                                                                                                                                                 |                           |                |                 |              |                 |              |     |  |  |  |  |  |  |
| <a href="#">Overview</a> <a href="#">Configuration</a> <a href="#">Routing</a> <a href="#">Services</a>                                                                                                                                                       |                           |                |                 |              |                 |              |     |  |  |  |  |  |  |
| <b>Logical Router Ports</b>                                                                                                                                                                                                                                   |                           |                |                 |              |                 |              |     |  |  |  |  |  |  |
| <a href="#">+ ADD</a> <a href="#">EDIT</a> <a href="#">DELETE</a> <a href="#">ACTIONS</a>                                                                                                                                                                     |                           |                |                 |              |                 |              |     |  |  |  |  |  |  |
| <table border="1"> <thead> <tr> <th>Logical Router</th> <th>ID</th> <th>Type</th> <th>IP Address/mask</th> <th>Connected To</th> <th>Tra</th> </tr> </thead> <tbody> <tr> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table> |                           | Logical Router | ID              | Type         | IP Address/mask | Connected To | Tra |  |  |  |  |  |  |
| Logical Router                                                                                                                                                                                                                                                | ID                        | Type           | IP Address/mask | Connected To | Tra             |              |     |  |  |  |  |  |  |
|                                                                                                                                                                                                                                                               |                           |                |                 |              |                 |              |     |  |  |  |  |  |  |

8. Select **Configuration > Router Ports**.

9. Click **Add**.

10. Configure a new router port as follows:

- ◊ **Name:** TO-uplink-1
- ◊ **Type:** uplink
- ◊ **Transport Node:** edge-node-1
- ◊ **Logical Switch:** LS-TO-uplink
- ◊ **Logical Switch Port:** Attach to a new switch port
- ◊ **Subnet:** 10.173.62.50 / 24

New Router Port

| Name*                                                                                                                                                                                                                                                                                                                             | TO-uplink-1                                                                                                     |             |                |              |    |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|-------------|----------------|--------------|----|
| Description                                                                                                                                                                                                                                                                                                                       |                                                                                                                 |             |                |              |    |
| Type                                                                                                                                                                                                                                                                                                                              | Uplink                                                                                                          | MTU         |                |              |    |
| Transport Node*                                                                                                                                                                                                                                                                                                                   | edge-node-1                                                                                                     |             |                |              |    |
| URPF Mode                                                                                                                                                                                                                                                                                                                         | <input checked="" type="radio"/> Strict <input type="radio"/> None                                              |             |                |              |    |
| Logical Switch                                                                                                                                                                                                                                                                                                                    | LS-TO-uplink                                                                                                    |             |                |              |    |
| OR Create a New Switch                                                                                                                                                                                                                                                                                                            |                                                                                                                 |             |                |              |    |
| Logical Switch Port                                                                                                                                                                                                                                                                                                               | <input checked="" type="radio"/> Attach to new switch port <input type="radio"/> Attach to existing switch port |             |                |              |    |
| Switch Port Name                                                                                                                                                                                                                                                                                                                  |                                                                                                                 |             |                |              |    |
| <b>Subnets</b>                                                                                                                                                                                                                                                                                                                    |                                                                                                                 |             |                |              |    |
| <input type="button" value="+ ADD"/> <input type="button" value="DELETE"/> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th style="background-color: #f2f2f2;">IP Address*</th> <th style="background-color: #f2f2f2;">Prefix Length*</th> </tr> <tr> <td>10.173.62.50</td> <td>24</td> </tr> </table> |                                                                                                                 | IP Address* | Prefix Length* | 10.173.62.50 | 24 |
| IP Address*                                                                                                                                                                                                                                                                                                                       | Prefix Length*                                                                                                  |             |                |              |    |
| 10.173.62.50                                                                                                                                                                                                                                                                                                                      | 24                                                                                                              |             |                |              |    |
| <input type="button" value="CANCEL"/> <input type="button" value="ADD"/>                                                                                                                                                                                                                                                          |                                                                                                                 |             |                |              |    |

11. Click **Add** and verify.

12. Select the TO-router you created.
13. Select **Configuration > Router Ports**.
14. Add a second uplink by creating a second router port for edge-node-2:

- ◊ **Name:** TO-uplink-1
- ◊ **Type:** uplink
- ◊ **Transport Node:** edge-node-2

- ◊ **Logical Switch:** LS-T0-uplink
- ◊ **Logical Switch Port:** Attach to a new switch port
- ◊ **Subnet:** 10.173.62.51 / 24

15. Once completed, verify that the T0-router you created has two connected router ports.

| Logical Router ID | Type   | IP Address/mask | Connected To                                | Transport Node | Relay Service | Statistics           |
|-------------------|--------|-----------------|---------------------------------------------|----------------|---------------|----------------------|
| TO-uplink-1       | Uplink | 10.173.62.50/24 | ↳ LS-T0-uplink<br>( feb3ie89-811e-482c... ) | edge-node-1    |               | <a href="#">View</a> |
| TO-uplink-2       | Uplink | 10.173.62.51/24 | ↳ LS-T0-uplink<br>( 4e99912a-e75c-47c... )  | edge-node-2    |               | <a href="#">View</a> |

## Configure and Test the Tier-0 Router

Create an HA VIP for the T0 router, and a default route for the T0 router. Then test the T0 router.

1. Select the T0-router you created.
2. Select Configuration > HA VIP.
3. Click Add.

| VIP Address | Uplink Ports | Status |
|-------------|--------------|--------|
|             |              |        |

4. Configure the HA VIP as follows:

- ◊ **VIP address:** 10.173.62.52/24, for example.
- ◊ **Uplink ports:** T0-uplink-1 and T0-uplink-2.

Add HA VIP Configuration

VIP Address: 10.173.62.52/24

Status: Enabled

Uplink Ports:

| Available (2) | Selected (2) |
|---------------|--------------|
| TO-uplink-1   | TO-uplink-1  |
| TO-uplink-2   | TO-uplink-2  |

**CANCEL** **ADD**

5. Click Add and verify.

The screenshot shows the 'Tier-0 Logical Routers' section. A logical router named 'TO-router' is selected. In the 'HA VIP Configuration' tab, there is one entry for 'VIP Address' (10.173.62.52/24) with 'Uplink Ports' set to 'T0-uplink-1,T0-uplink-2' and 'Status' set to 'Enabled'.

#### 6. Select Routing > Static Routes.

The screenshot shows the 'Routing' tab selected under 'Static Routes'. There is a table with columns: Network (IP/mask), ID, Next Hop, Admin Distance, and Logical Router Port. No routes are currently listed.

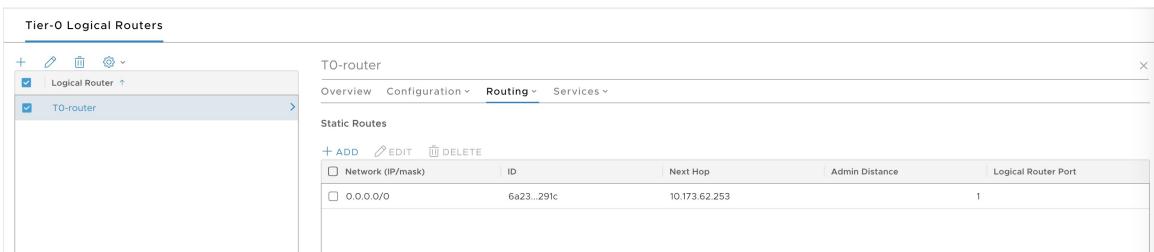
#### 7. Click Add.

- ◊ **Network:** 0.0.0.0/0
- ◊ **Next Hop:** 10.173.62.253

The dialog box has the title 'Add Static Route'. It contains fields for 'Network (IP/mask)\*' (0.0.0.0/0), 'Description' (empty), and 'Next Hops' (a table). The 'Next Hops' table has columns: Next Hop, Admin Distance, and Logical Router Port. One entry is present: 10.173.62.253 with Admin Distance 1. At the bottom, a note says 'Select NULL as Next Hop to configure Null Routes'. There are 'CANCEL' and 'ADD' buttons at the bottom right.

| Next Hop      | Admin Distance | Logical Router Port |
|---------------|----------------|---------------------|
| 10.173.62.253 | 1              |                     |

#### 8. Click Add and verify.



- Verify the Tier 0 router by making sure the TO uplinks and HA VIP are reachable from your laptop.

For example:

```
> ping 10.173.62.50
PING 10.173.62.50 (10.173.62.50): 56 data bytes
Request timeout for icmp_seq 0
64 bytes from 10.173.62.50: icmp_seq=1 ttl=58 time=71.741 ms
64 bytes from 10.173.62.50: icmp_seq=0 ttl=58 time=1074.679 ms

> ping 10.173.62.51
PING 10.173.62.51 (10.173.62.51): 56 data bytes
Request timeout for icmp_seq 0
64 bytes from 10.173.62.51: icmp_seq=0 ttl=58 time=1156.627 ms
64 bytes from 10.173.62.51: icmp_seq=1 ttl=58 time=151.413 ms

> ping 10.173.62.52
PING 10.173.62.52 (10.173.62.52): 56 data bytes
64 bytes from 10.173.62.52: icmp_seq=0 ttl=58 time=6.864 ms
64 bytes from 10.173.62.52: icmp_seq=1 ttl=58 time=7.776 ms
```

## Create the Nodes IP Block for Kubernetes Clusters Using the Management Interface

TKGI requires a dedicated IP Block for Kubernetes nodes. When you configure the TKGI tile, you will need to provide the ID for this IP Block. The recommended size for this IP Block is /16. For more information, see [Nodes IP Block](#).

In the example that follows, we create the following Nodes IP block: TKGI-NODES-IP-BLOCK: 172.23.0.0/16.

To create the required Nodes IP Block using the Management interface, complete the following steps:

- Log in to the NSX-T Manager and select the **Networking** tab.
- Verify that the **Manager** interface is selected. If not, select it.
- Select **Network Services > IP Address Pools > IP Block**.

| ID | CIDR |
|----|------|
|    |      |

4. Click **Add**.
5. Configure the Nodes IP Block as follows:
  - ◊ **Name:** TKGI-NODES-IP-BLOCK
  - ◊ **CIDR:** 172.23.0.0/16
6. Click **Add** and verify.

| ID          | CIDR          |
|-------------|---------------|
| 5b5c...2615 | 172.23.0.0/16 |
| 77c8..0c23  | 172.16.0.0/16 |

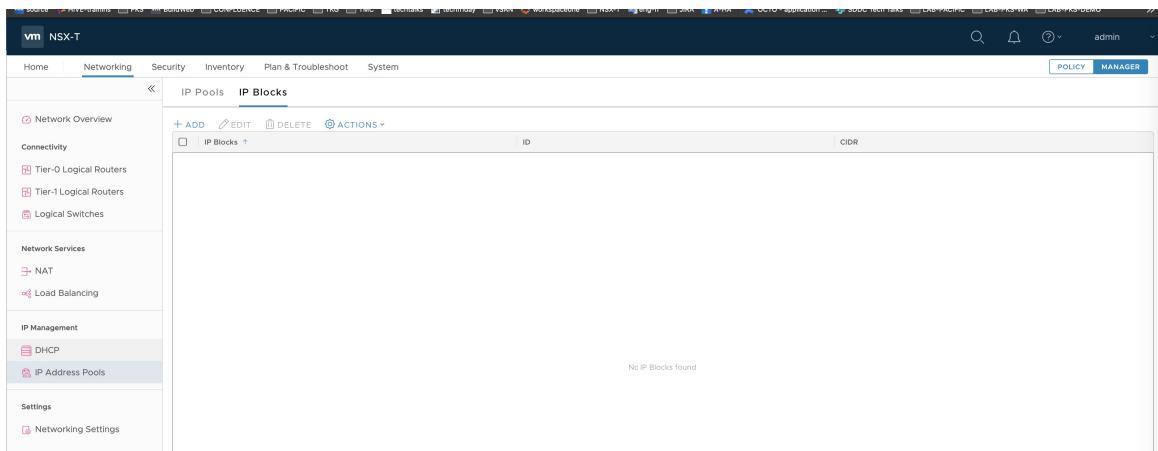
## Create the Pods IP Block for Kubernetes Clusters Using the Management Interface

TKGI requires a dedicated IP Block for Kubernetes pods. When you configure the TKGI tile, you will need to provide the ID for this IP Block. The recommended size for this IP Block is /16. For more information, see [Pods IP Block](#).

In the example that follows, we create the following Pods IP block: TKGI-PODS-IP-BLOCK: 172.16.0.0/16.

To create the required Pods IP Block using the Management interface, complete the following steps:

1. Log in to the NSX-T Manager and select the **Networking** tab.
2. Verify that the **Manager** interface is selected. If not, select it.
3. Select **Network Services > IP Address Pools > IP Block**.



4. Click **Add**.
5. Configure the Pods IP Block as follows:
  - ◊ **Name:** TKGI-PODS-IP-BLOCK
  - ◊ **CIDR:** 172.16.0.0/16
6. Click **Add** and verify.

### New IP Block

? X

|             |                  |
|-------------|------------------|
| Name *      | PKS-POD-IP-BLOCK |
| Description |                  |
| CIDR *      | 172.16.0.0/16    |

CANCEL
ADD

## Create the Floating IP Pool for Kubernetes Clusters Using the Management Interface

TKGI requires a floating IP pool for Kubernetes services such as load balancer instances. When you configure the TKGI tile, you will need to provide the ID for this IP Pool. For more information, see [Plan Network CIDRs](#).

To create the required Floating IP Pool using the Management interface, complete the following steps:

1. Log in to the NSX-T Manager and select the **Networking** tab.
2. Verify that the **Manager** interface is selected. If not, select it.

3. Select Network Services > IP Address Pools > IP Block.

4. Click Add.

5. Configure the IP pool as follows:

- ◊ **Name:** TKGI-FLOATING-IP-POOL
- ◊ **IP ranges:** 10.173.62.111 - 10.173.62.150
- ◊ **CIDR:** 10.173.62.0/24

| IP Range*                     | Gateway | CIDR*          | DNS Servers | DNS Suffix |
|-------------------------------|---------|----------------|-------------|------------|
| 10.173.62.111 - 10.173.62.150 |         | 10.173.62.0/24 |             |            |

6. Click Add and verify.

| ID          | Subnets | Allocations |
|-------------|---------|-------------|
| 8840...6a27 | 1       | 0 of 40     |
| 34f3...2b74 | 1       | 0 of 248    |
| 060a...4127 | 1       | 0 of 1022   |
| 5dee...3793 | 1       | 0 of 248    |
| dfcf...b5d3 | 1       | 3 of 10     |

## Create NSX-T Objects for Kubernetes Clusters Using the Policy Interface

This section provides instructions for creating the required NSX-T objects for Kubernetes clusters

using the Policy interface.

## Create a Tier-0 Gateway Using the Policy Interface

1. Log in to the NSX-T Manager and select the **Networking** tab.
2. Verify that the **Policy** interface is selected. If not, select it.
3. Select **Tier-0 Gateways** from the navigation on the left.
4. Click **Add Gateway**.
5. Select **Tier-0**.
6. Create a new Tier-0 Gateway as follows:
  - **Name:** Enter `t0-shared`, for example
  - **HA Mode:** Select `Active Standby`
  - **Fail Over:** Select `Non Preemptive`
  - Click **Save**
  - At the prompt select **Yes** to continue creating the gateway
  - **Edge Cluster:** Select the Edge Cluster you created previously, such as `edge-cluster-0`
  - Click **Close Editing** to complete the creation of the Tier-0 Gateway

The screenshot shows the 'Add Gateway' dialog for creating a Tier-0 Gateway. The form includes fields for 'Tier-0 Gateway Name' (t0-shared-testing), 'HA Mode' (Active Standby), 'Fail Over' (Non Preemptive), 'Edge Cluster' (Select Edge Cluster), and 'Description'. Below the form, a success message states: 'Tier-0 Gateway test-0 is successfully created. Want to continue configuring this Tier-0 Gateway?' with 'YES' and 'NO' buttons.

| ADD GATEWAY ▾                                                   |                     | Filter by Name, Path and more |                        |                                                  |          |
|-----------------------------------------------------------------|---------------------|-------------------------------|------------------------|--------------------------------------------------|----------|
|                                                                 | Tier-0 Gateway Name | HA Mode                       | Linked Tier-1 Gateways | Linked Segments                                  | Status ⓘ |
|                                                                 | t0-shared-testing * | Active Standby *              |                        |                                                  |          |
|                                                                 | Fail Over           | Non Preemptive                |                        |                                                  |          |
|                                                                 | Edge Cluster        | Select Edge Cluster           |                        |                                                  |          |
| > Additional Settings<br>> Route Distinguisher for VRF Gateways |                     |                               |                        |                                                  |          |
| Description                                                     | Description         |                               | Tags                   | Tag Scope +<br>Max 30 allowed. Click (+) to add. |          |

**Tier-0 Gateway test-0 is successfully created.**  
Want to continue configuring this Tier-0 Gateway?

**YES** | **NO**

The screenshot shows the 'ADD GATEWAY' form. Key fields include:

- Tier-0 Gateway Name:** t0-shared-testing
- HA Mode:** Active Standby
- Edge Cluster:** edge-cluster-0
- HA VIP Configuration:** Set
- Description:** Description
- Tags:** Tag (with a note: Max 30 allowed. Click (+) to add.)

## Configure the Tier-0 Gateway Using the Policy Interface

Now that the Tier-0 Gateway is created, you need to configure it for TKGI. This requires configuring two interfaces, an HA VIP, and a static route.

1. **Edit** the **t0-shared** gateway you created.
2. Select **Interfaces > Set**.
3. Select **Add Interface** and configure the first interface as follows:
  - **Name:** Uplink1EdgeFirst (for example)
  - **Type:** External
  - **Edge Node:** tn-cluster-0-edge-0
  - **MTU:** 1500
  - **Connected To (Segment):** internet-vlan-0
  - **Subnet:** 192.168.115.10/24 (for example)
  - Click **Save** and verify.

| Name             | Type     | IP Address / Mask | Connected To(Segment) | Status  |
|------------------|----------|-------------------|-----------------------|---------|
| Uplink1EdgeFirst | External | 192.168.115.10/24 | internet-vlan-0       | Success |

Below the table, interface settings are displayed:

- Edge Node: tn-cluster-0-edge-0
- MTU: 1500
- URPF Mode: Strict
- Description: Not Set
- Tags: 1

4. Select **Add Interface** and configure the second interface as follows:
  - **Name:** Uplink2EdgeFirst (for example)
  - **Type:** External
  - **Edge Node:** tn-cluster-0-edge-1
  - **MTU:** 1500
  - **Connected To (Segment):** internet-vlan-0

- ◊ **Subnet:** 192.168.115.11/24 (for example)

- ◊ Click **Save** and verify.

|             |                     |                  |                             |
|-------------|---------------------|------------------|-----------------------------|
| Edge Node   | tn-cluster-0-edge-1 | ND Profile       | default <a href="#">(1)</a> |
| MTU         | 1500                | Proxy ARP Filter | Not Set                     |
| URPF Mode   | Strict              |                  |                             |
| Description | Not Set             | Tags             | 1                           |
| > OSPF      |                     |                  |                             |
| > MULTICAST |                     |                  |                             |

[VIEW STATISTICS](#)

5. When you are done adding the interfaces, click **Apply**.

6. Configure the HA VIP as follows:

- ◊ For the **HA VIP Configuration** field, click **Set**.
- ◊ Click **Add HA VIP Configuration**.
- ◊ For the **IP Address / Mask** field, enter a valid IP address and subnet mask
- ◊ For the **Interface** field, select the 2 interfaces you created
- ◊ Click **Apply**

Set HA VIP Configuration X

Tier-0 Gateway    t0-shared #HA VIP Configuration [\(1\)](#)

| ADD HA VIP CONFIGURATION |         | Search                             |
|--------------------------|---------|------------------------------------|
| IP Address / Mask        | Enabled | Interface                          |
| 192.168.115.2/24         | Enabled | Uplink1EdgeFirst, Uplink2EdgeFirst |

7. Configure a static route as follows:

- ◊ Select **Routing > Static Routes**.
- ◊ Click **Set**.
- ◊ Click **Add Static Route**.
- ◊ **Name:** default
- ◊ **Network:** 0.0.0.0/0
- ◊ For **Next Hops**, click **Set**.
- ◊ Click **Set Next Hops**.
- ◊ **Next Hop:** 192.168.115.1 (for example)
- ◊ **Admin Distance:** 1
- ◊ **Scope:** None
- ◊ Click **Save**.
- ◊ Click **Close**.

## Next Hops

Tier-0 Gateway    t0-shared    |    Static Route    default #Next Hops 1

Search

| IP Address    | Admin Distance | Scope |
|---------------|----------------|-------|
| 192.168.115.1 | 1              | None  |

## Set Static Routes

Tier-0 Gateway    t0-shared #Static Routes 1

**ADD STATIC ROUTE**

Search

| Name    | Network   | Next Hops | Status                                     |
|---------|-----------|-----------|--------------------------------------------|
| default | 0.0.0.0/0 | 1         | <span style="color: green;">Success</span> |

- Click **Close Editing** and verify the configuration of the t0-shared gateway.

| Tier-0 Gateway Name                                                                                                    | HA Mode               | Linked Tier-1 Gateways | Linked Segments          | Status                                     | Alarms                                |
|------------------------------------------------------------------------------------------------------------------------|-----------------------|------------------------|--------------------------|--------------------------------------------|---------------------------------------|
| t0-shared                                                                                                              | Active Standby        | 2                      | 0                        | <span style="color: green;">Success</span> | 0                                     |
| Fall Over                                                                                                              | Non Preemptive        | DHCP                   | No Dynamic IP Allocation |                                            | <a href="#">VIEW DNS</a>              |
| Edge Cluster                                                                                                           | edge-cluster-0        |                        |                          |                                            | <a href="#">VIEW NAT</a>              |
| HA VIP Configuration                                                                                                   | 1                     |                        |                          |                                            | <a href="#">VIEW GATEWAY FIREWALL</a> |
| <a href="#">Additional Settings</a> <a href="#">Route Distinguisher for VRF Gateways</a> <a href="#">EVPN Settings</a> |                       |                        |                          |                                            |                                       |
| Description                                                                                                            | Tier-0 provisioned by | Tags                   | 1                        |                                            |                                       |
| <b>INTERFACES</b>                                                                                                      |                       |                        |                          |                                            |                                       |
| External and Service Interfaces 2                                                                                      |                       |                        |                          |                                            |                                       |

## Test the Tier-0 Gateway

Now that the **t0-shared** gateway is configured, it is time to test it.

- Verify the TO-gateway by making sure the TO uplinks and HA VIP are reachable from your laptop.

For example:

```
> ping 10.173.62.50
PING 10.173.62.50 (10.173.62.50): 56 data bytes
Request timeout for icmp_seq 0
64 bytes from 10.173.62.50: icmp_seq=1 ttl=58 time=71.741 ms
64 bytes from 10.173.62.50: icmp_seq=0 ttl=58 time=1074.679 ms

> ping 10.173.62.51
PING 10.173.62.51 (10.173.62.51): 56 data bytes
Request timeout for icmp_seq 0
64 bytes from 10.173.62.51: icmp_seq=0 ttl=58 time=1156.627 ms
64 bytes from 10.173.62.51: icmp_seq=1 ttl=58 time=151.413 ms

> ping 10.173.62.52
PING 10.173.62.52 (10.173.62.52): 56 data bytes
64 bytes from 10.173.62.52: icmp_seq=0 ttl=58 time=6.864 ms
```

```
64 bytes from 10.173.62.52: icmp_seq=1 ttl=58 time=7.776 ms
```

## Create the Nodes IP Block for Kubernetes Clusters Using the Policy Interface

TKGI requires a dedicated IP Block for Kubernetes nodes. When you configure the TKGI tile, you will need to provide the ID for this IP Block. The recommended size for this IP Block is /16. For more information, see [Nodes IP Block](#).

In the example that follows, we create the following Nodes IP block: TKGI-NODES-IP-BLOCK: 172.23.0.0/16.

To create the required Nodes IP Block using the Management interface, complete the following steps:

1. Log in to the NSX-T Manager and select the **Networking** tab.
2. Verify that the **Policy** interface is selected. If not, select it.
3. Select **IP Management > IP Address Pools > IP Address Blocks**.
4. Click **Add IP Address Block**.
5. Configure the Nodes IP Block as follows:
  - **Name:** TKGI-NODES-IP-BLOCK
  - **CIDR:** 172.23.0.0/16
6. Click **Add** and verify.

## Create the Pods IP Block for Kubernetes Clusters Using the Policy Interface

TKGI requires a dedicated IP Block for Kubernetes pods. When you configure the TKGI tile, you will need to provide the ID for this IP Block. The recommended size for this IP Block is /16. For more information, see [Pods IP Block](#).

For example:

- TKGI-PODS-IP-BLOCK: 172.16.0.0/16

To create the required Pods IP Block using the Management interface, complete the following steps:

1. Log in to the NSX-T Manager and select the **Networking** tab.
2. Verify that the **Policy** interface is selected. If not, select it.
3. Select **IP Management > IP Address Pools > IP Address Blocks**.
4. Click **Add IP Address Block**.
5. Configure the Pods IP Block as follows:
  - **Name:** TKGI-PODS-IP-BLOCK
  - **CIDR:** 172.16.0.0/16
6. Click **Add** and verify.

## Create the Floating IP Pool for Kubernetes Clusters Using the

## Management Interface

TKGI requires a floating IP pool for Kubernetes services such as load balancer instances. When you configure the TKGI tile, you will need to provide the ID for this IP Pool. For more information, see [Plan Network CIDRs](#).

To create the required Floating IP Pool using the Management interface, complete the following steps:

1. Log in to the NSX-T Manager and select the **Networking** tab.
2. Verify that the **Policy** interface is selected. If not, select it.
3. Select **IP Management > IP Address Pools > IP Address Pools**.
4. Click **Add IP Address Pool**.
5. Configure the IP pool as follows:
  - ◊ **Name:** TKGI-FLOATING-IP-POOL
  - ◊ **IP ranges:** 10.173.62.111 - 10.173.62.150
  - ◊ **CIDR:** 10.173.62.0/24
6. Click **Add** and verify.

## Next Steps

[Create NSX-T Objects for TKGI Management Plane Components](#)

## Create VMware NSX Objects for the Management Plane

This topic provides instructions for creating the NSX-T objects for the TKGI Management Plane.

## Prerequisites

Before completing this section, make sure you have completed the following sections:

- [NSX-T Installation Prerequisites](#)
- [Install and Configure the NSX-T Manager Hosts](#)
- [Generate and Register the NSX-T TLS Certificate and Private Key](#)
- [Create an IP Pool for VTEP](#)
- [Configure Transport Zones](#)
- [Configure vSphere Networking for ESXi Hosts](#)
- [Deploy NSX-T Edge Nodes](#)
- [Deploy NSX-T Transport Nodes](#)
- [Create NSX-T Objects for Kubernetes Clusters Provisioned by TKGI](#)

## Create Management Plane

Networking for the TKGI Management Plane consists of a **Tier-1 Router and Switch** with **NAT Rules** for the Management Plane VMs.

## Create Tier-1 Router and Switch

Create Tier-1 Logical Switch and Router for TKGI Management Plane VMs. Complete the configuration by enabling Route Advertisement on the T1 router.

1. In the NSX Management console, navigate to **Networking > Logical Switches**.
2. Click **Add**.
3. Create the LS for TKGI Management plane VMs:
  - **Name:** LS-PKS-MGMT
  - **Transport Zone:** tz-overlay

**Add New Logical Switch**

[?](#) [X](#)

[General](#) [Switching Profiles](#)

---

|                                                           |                                                                                                                  |
|-----------------------------------------------------------|------------------------------------------------------------------------------------------------------------------|
| Name *                                                    | LS-PKS-MGMT                                                                                                      |
| Description                                               | <input type="text"/>                                                                                             |
| Transport Zone *                                          | tz-overlay                                                                                                       |
| Uplink Teaming Policy Name *                              | [Use Default]                                                                                                    |
| Admin Status                                              | <input checked="" type="checkbox"/> Up                                                                           |
| Replication Mode                                          | <input checked="" type="radio"/> Hierarchical Two-Tier replication<br><input type="radio"/> Head End replication |
| VLAN                                                      | <input type="text"/>                                                                                             |
| Only VLAN Trunk Spec is allowed (eg: 1, 5, 10-12, 31-35). |                                                                                                                  |
| <a href="#">CANCEL</a> <a href="#">ADD</a>                |                                                                                                                  |

4. Click **Add** and verify creation of the T1 logical switch.

| Switches                 |                |            |              |               |                 |              |
|--------------------------|----------------|------------|--------------|---------------|-----------------|--------------|
| Actions                  |                | ID         | Admin Status | Logical Ports | Traffic Type    | Config State |
| <input type="checkbox"/> | Logical Switch | fa6a..afbb | Up           | 0             | Overlay : 69636 | Pending      |
| <input type="checkbox"/> | LS-PKS-MGMT    | f70b..3a68 | Up           | 4             | VLAN : 1548     | Success      |
| <input type="checkbox"/> | LS-T0-uplink   |            |              |               |                 |              |

5. Go to Networking > Tier-1 Logical Router.

| Logical Router | ID | Connected Tier-0 Router |
|----------------|----|-------------------------|
| LS-PKS-MGMT    |    |                         |

6. Click Add.

7. Configure the Tier-1 logical router as follows:

- ◊ **Name:** T1-PKS-MGMT
- ◊ **To router:** T0-router
- ◊ **Edge Cluster:** edge-cluster-1
- ◊ **Edge Cluster Members:** edge-node-1 and edge-node-2

## New Tier-1 Router

[?](#) [X](#)

[Tier-1 Router](#) [Advanced](#)

---

|                                        |                                                                                                                                 |
|----------------------------------------|---------------------------------------------------------------------------------------------------------------------------------|
| Name *                                 | T1-PKS-MGMT                                                                                                                     |
| Description                            | <input type="text"/>                                                                                                            |
| Tier-0 Router                          | TO-router <a href="#">x</a> <a href="#">▼</a>                                                                                   |
| Edge Cluster                           | edge-cluster-1 <a href="#">x</a> <a href="#">▼</a>                                                                              |
| StandBy Relocation                     | <input checked="" type="checkbox"/> Disable                                                                                     |
| Failover Mode                          | <input type="radio"/> Preemptive <input checked="" type="radio"/> Non-Preemptive                                                |
| Edge Cluster Members <a href="#">i</a> | <a href="#">edge-node-1</a> <a href="#">x</a> <a href="#">edge-node-2</a> <a href="#">x</a> <a href="#">x</a> <a href="#">▼</a> |

---

[CANCEL](#) [ADD](#)

8. Click **Add** and verify.

| Tier-1 Logical Routers                                                                                      |             |                         |                        |                |                |
|-------------------------------------------------------------------------------------------------------------|-------------|-------------------------|------------------------|----------------|----------------|
| <a href="#">+ ADD</a> <a href="#">EDIT</a> <a href="#">DELETE</a> <a href="#">ACTIONS</a> <a href="#">▼</a> |             | Search                  |                        |                |                |
| Logical Router                                                                                              | ID          | Connected Tier-0 Router | High Availability Mode | Transport Zone | Edge Cluster   |
| T1-PKS-MGMT                                                                                                 | f293...9ee6 | TO-router               | Active-Standby         |                | edge-cluster-1 |

9. Select the T1 router and go to **Configuration > Router port**.

| Tier-1 Logical Routers                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |              |                                         |                              |                                                                           |                            |                     |                   |      |                 |              |                |               |            |              |              |             |                              |                                                                           |                            |                     |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|-----------------------------------------|------------------------------|---------------------------------------------------------------------------|----------------------------|---------------------|-------------------|------|-----------------|--------------|----------------|---------------|------------|--------------|--------------|-------------|------------------------------|---------------------------------------------------------------------------|----------------------------|---------------------|
| <a href="#">+ ADD</a> <a href="#">EDIT</a> <a href="#">DELETE</a> <a href="#">ACTIONS</a> <a href="#">▼</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |              | T1-PKS-MGMT                             |                              |                                                                           |                            |                     |                   |      |                 |              |                |               |            |              |              |             |                              |                                                                           |                            |                     |
|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |              | Overview Configuration Routing Services |                              |                                                                           |                            |                     |                   |      |                 |              |                |               |            |              |              |             |                              |                                                                           |                            |                     |
| <b>Logical Router Ports</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |              |                                         |                              |                                                                           |                            |                     |                   |      |                 |              |                |               |            |              |              |             |                              |                                                                           |                            |                     |
| <a href="#">+ ADD</a> <a href="#">EDIT</a> <a href="#">DELETE</a> <a href="#">ACTIONS</a> <a href="#">▼</a> <table border="1"> <thead> <tr> <th>Logical Router ID</th> <th>Type</th> <th>IP Address/mask</th> <th>Connected To</th> <th>Transport Node</th> <th>Relay Service</th> <th>Statistics</th> </tr> </thead> <tbody> <tr> <td>LinkedPor...</td> <td>4d2d...db...</td> <td>Linked Port</td> <td>100.64.48.1/31, fca9:6c79...</td> <td><input checked="" type="radio"/> TO-router<br/>( LinkedPort_T1-PKS-MG... )</td> <td>edge-node-1<br/>edge-node-2</td> <td><a href="#">all</a></td> </tr> </tbody> </table> |              |                                         |                              |                                                                           |                            |                     | Logical Router ID | Type | IP Address/mask | Connected To | Transport Node | Relay Service | Statistics | LinkedPor... | 4d2d...db... | Linked Port | 100.64.48.1/31, fca9:6c79... | <input checked="" type="radio"/> TO-router<br>( LinkedPort_T1-PKS-MG... ) | edge-node-1<br>edge-node-2 | <a href="#">all</a> |
| Logical Router ID                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       | Type         | IP Address/mask                         | Connected To                 | Transport Node                                                            | Relay Service              | Statistics          |                   |      |                 |              |                |               |            |              |              |             |                              |                                                                           |                            |                     |
| LinkedPor...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | 4d2d...db... | Linked Port                             | 100.64.48.1/31, fca9:6c79... | <input checked="" type="radio"/> TO-router<br>( LinkedPort_T1-PKS-MG... ) | edge-node-1<br>edge-node-2 | <a href="#">all</a> |                   |      |                 |              |                |               |            |              |              |             |                              |                                                                           |                            |                     |

10. Click **Add**.

11. Configure the T1 router port as follows:

- Name: T1-PKS-MGMT-port

- ◊ **Logical Switch:** LS-PKS-MGMT

- ◊ **Subnet:** 10.1.1.1/24

### Edit Router Port - T1-PKS-MGMT-port

(?) (X)

| Name *                                                                                                                                                                                                                                                                                                            | T1-PKS-MGMT-port                                                                                                |              |                 |          |    |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|--------------|-----------------|----------|----|
| Description                                                                                                                                                                                                                                                                                                       | <input type="text"/>                                                                                            |              |                 |          |    |
| Type                                                                                                                                                                                                                                                                                                              | Downlink                                                                                                        |              |                 |          |    |
| URPF Mode                                                                                                                                                                                                                                                                                                         | <input checked="" type="radio"/> Strict <input type="radio"/> None                                              |              |                 |          |    |
| Logical Switch                                                                                                                                                                                                                                                                                                    | LS-PKS-MGMT <span>(X) ▾</span>                                                                                  |              |                 |          |    |
| OR Create a New Switch                                                                                                                                                                                                                                                                                            |                                                                                                                 |              |                 |          |    |
| Logical Switch Port                                                                                                                                                                                                                                                                                               | <input checked="" type="radio"/> Attach to new switch port <input type="radio"/> Attach to existing switch port |              |                 |          |    |
| Switch Port Name                                                                                                                                                                                                                                                                                                  | <input type="text"/>                                                                                            |              |                 |          |    |
| <b>Subnets</b><br><span>+ ADD</span> <span>DELETE</span> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">IP Address *</th> <th style="width: 10%;">Prefix Length *</th> </tr> </thead> <tbody> <tr> <td>10.1.1.1</td> <td>24</td> </tr> </tbody> </table> |                                                                                                                 | IP Address * | Prefix Length * | 10.1.1.1 | 24 |
| IP Address *                                                                                                                                                                                                                                                                                                      | Prefix Length *                                                                                                 |              |                 |          |    |
| 10.1.1.1                                                                                                                                                                                                                                                                                                          | 24                                                                                                              |              |                 |          |    |
| <span>CANCEL</span> <span>SAVE</span>                                                                                                                                                                                                                                                                             |                                                                                                                 |              |                 |          |    |

12. Click **Add** and verify.

| Logical Router ID | Type     | IP Address/mask | Connected To             | Transport Node | Relay Service | Statistics |
|-------------------|----------|-----------------|--------------------------|----------------|---------------|------------|
| LinkedPort...     | Downlink | 10.1.1.1/24     | edge-node-1, edge-node-2 |                |               |            |
| T1-PKS-MGMT       | Downlink | 10.1.1.1/24     |                          |                |               |            |

13. Select **Routing** tab.

The screenshot shows the 'Tier-1 Logical Routers' configuration screen. A logical router named 'T1-PKS-MGMT' is selected. The 'Routing' tab is active. Under 'Route Advertisement', the 'Status' is set to 'Enabled'. Other options like 'Advertise All Connected Routes' are set to 'Yes'. There is also a table for 'Advertise Routes' with a single entry for 'Name'.

14. Click **Edit** and configure route advertisement as follows:

- **Status:** Enabled
- **Advertise All Connected Routes:** Yes

### Edit Route Advertisement Configuration

|                                     |                                             |
|-------------------------------------|---------------------------------------------|
| Status                              | <input checked="" type="checkbox"/> Enabled |
| Advertise All Connected Routes      | <input checked="" type="checkbox"/> Yes     |
| Advertise All NAT Routes            | <input type="checkbox"/> No                 |
| Advertise All Static Routes         | <input type="checkbox"/> No                 |
| Advertise All LB VIP Routes         | <input type="checkbox"/> No                 |
| Advertise All LB SNAT IP Routes     | <input type="checkbox"/> No                 |
| Advertise All DNS Forwarder Routes  | <input type="checkbox"/> No                 |
| Advertise All IPsec Local Endpoints | <input type="checkbox"/> No                 |

**CANCEL** **SAVE**

15. Click **Save** and verify.

**Tier-1 Logical Routers**

**T1-PKS-MGMT**

**Route Advertisement | EDIT**

|                                     |            |
|-------------------------------------|------------|
| Status                              | ● Enabled  |
| Advertise All Connected Routes      | ● Yes      |
| Advertise All NAT Routes            | ● No       |
| Advertise All Static Routes         | ● No       |
| Advertise All LB VIP Routes         | ● No       |
| Advertise All LB SNAT IP Routes     | ● No       |
| Advertise All DNS Forwarder Routes  | ● No       |
| Advertise All IPSec Local Endpoints | ● No       |
| Advertised Networks                 | 0 Networks |

## Create NAT Rules

You need to create the following NAT rules on the Tier-0 router for the TKGI Management Plane VMs.

- DNAT: `10.173.62.220` (for example) to access Ops Manager
- DNAT: `10.173.62.221` (for example) to access Harbor
- SNAT: `10.173.62.222` (for example) for all TKGI management plane VM traffic destined to the outside world
- In the NSX Management console, navigate to **Networking > NAT**.
- In the Logical Router field, select the TO-router you defined for TKGI.

**NAT**

Logical Router: TO-router

| ID | Action | Match | Protocol | Source IP | Source Ports | Destination IP | Destination Ports | Transla IP |
|----|--------|-------|----------|-----------|--------------|----------------|-------------------|------------|
|----|--------|-------|----------|-----------|--------------|----------------|-------------------|------------|

- Click **Add**.
- Configure the Ops Manager DNAT rule as follows:
  - **Priority:** `1000`
  - **Action:** `DNAT`
  - **Protocol:** `Any Protocol`
  - **Destination IP:** `10.173.62.220`, for example
  - **Translated IP:** `10.1.1.2`, for example

## New NAT Rule

Priority: 1000

Action\*: DNAT

Protocol:  Any Protocol  Specific Protocol

Source IP:

Destination IP\*: 10.173.62.220

Translated IP\*: 10.1.1.2

Translated Ports:

Applied To:

Status:  Enabled

Logging:  Disabled

Firewall Bypass:

**CANCEL** **ADD**

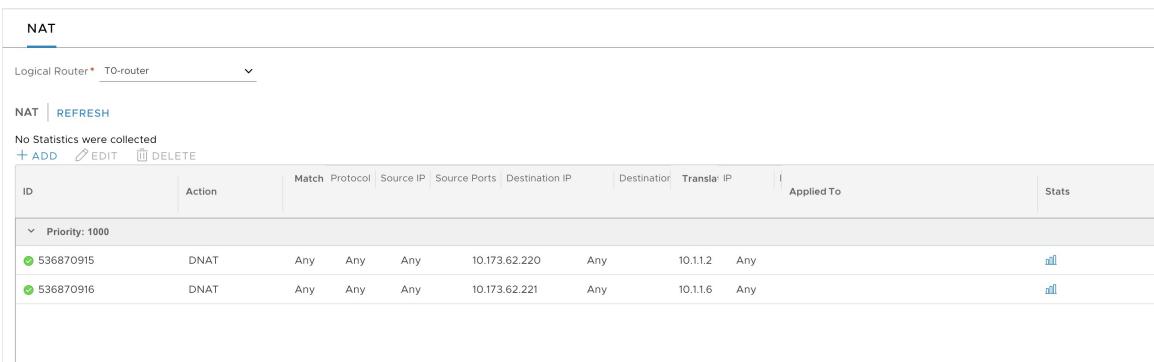
- Click **Add** and verify.

| NAT                          |                               |                                 |       |          |           |               |                |             |               |            |
|------------------------------|-------------------------------|---------------------------------|-------|----------|-----------|---------------|----------------|-------------|---------------|------------|
| Logical Router * TO-router   |                               |                                 |       |          |           |               |                |             |               |            |
| NAT   REFRESH                |                               |                                 |       |          |           |               |                |             |               |            |
| No Statistics were collected |                               |                                 |       |          |           |               |                |             |               |            |
| + ADD                        | <input type="checkbox"/> EDIT | <input type="checkbox"/> DELETE | Match | Protocol | Source IP | Source Ports  | Destination IP | Destination | Translated IP | Applied To |
| ID                           | Action                        |                                 |       |          |           |               |                |             |               | Stats      |
| ▼ Priority: 1000             |                               |                                 |       |          |           |               |                |             |               |            |
| 536870915                    | DNAT                          |                                 | Any   | Any      | Any       | 10.173.62.220 | Any            | 10.1.1.2    | Any           |            |

- Add a second DNAT rule for Harbor by repeating the same operation.

- Priority: 1000
- Action: DNAT
- Protocol: Any Protocol
- Destination IP: 10.173.62.221, for example

- ◊ **Translated IP:** 10.1.1.6, for example
- Verify the creation of the DNAT rules.



The screenshot shows the NAT configuration interface for a logical router named 'T0-router'. The interface includes a header with 'Logical Router' set to 'T0-router', a 'REFRESH' button, and a note 'No Statistics were collected'. Below this is a table with columns: ID, Action, Match, Protocol, Source IP, Source Ports, Destination IP, Destination, Transla[redacted] IP, Applied To, and Stats. Two rows of data are present, both labeled 'Priority: 1000' and '536870915' and '536870916'. The first row has 'Action: DNAT', 'Match: Any', 'Protocol: Any', 'Source IP: 10.173.62.220', 'Destination IP: 10.1.1.2', and 'Translated IP: Any'. The second row has similar values but with 'Destination IP: 10.1.1.6' and 'Translated IP: Any'.

| ID        | Action | Match | Protocol | Source IP | Source Ports  | Destination IP | Destination | Transla[redacted] IP | Applied To | Stats |
|-----------|--------|-------|----------|-----------|---------------|----------------|-------------|----------------------|------------|-------|
| 536870915 | DNAT   | Any   | Any      | Any       | 10.173.62.220 | Any            | 10.1.1.2    | Any                  |            |       |
| 536870916 | DNAT   | Any   | Any      | Any       | 10.173.62.221 | Any            | 10.1.1.6    | Any                  |            |       |

- Create the SNAT rule for the management plane traffic as follows:
  - ◊ **Priority:** 9024
  - ◊ **Action:** SNAT
  - ◊ **Protocol:** Any Protocol
  - ◊ **Source IP:** 10.1.1.0/24, for example
  - ◊ **Translated IP:** 10.173.62.222, for example

## New NAT Rule

Priority: 9024

Action\*: SNAT

Protocol:  Any Protocol  Specific Protocol

Source IP: 10.1.1.0/24

Destination IP:

Translated IP\*: 10.173.62.222

Applied To:

Status:  Enabled

Logging:  Disabled

Firewall Bypass:

**CANCEL** **ADD**

- Verify the creation of the SNAT rule.

| NAT                          |        |       |           |           |               |                |             |             |            |       |
|------------------------------|--------|-------|-----------|-----------|---------------|----------------|-------------|-------------|------------|-------|
| Logical Router* TO-router    |        |       |           |           |               |                |             |             |            |       |
| NAT   REFRESH                |        |       |           |           |               |                |             |             |            |       |
| No Statistics were collected |        |       |           |           |               |                |             |             |            |       |
| ID                           | Action | Match | Protocol  | Source IP | Source Ports  | Destination IP | Destination | Transla* IP | Applied To | Stats |
| ▼ Priority: 1000             |        |       |           |           |               |                |             |             |            |       |
| 536870915                    | DNAT   | Any   | Any       | Any       | 10.173.62.220 | Any            | 10.1.1.2    | Any         |            |       |
| 536870916                    | DNAT   | Any   | Any       | Any       | 10.173.62.221 | Any            | 10.1.1.6    | Any         |            |       |
| ▼ Priority: 9024             |        |       |           |           |               |                |             |             |            |       |
| 536870917                    | SNAT   | Any   | 10.1.1... | Any       | Any           | Any            | 10.173...   | Any         |            |       |

## Next Steps

Configure the NSX-T Password Interval

# Configure VMware NSX Passwords

This topic provides instructions for configure NSX-T passwords after you have installed NSX-T for TKGI.

## Prerequisites

Before completing this section, make sure you have completed the following sections:

- [NSX-T Installation Prerequisites](#)
- [Install and Configure the NSX-T Manager Hosts](#)
- [Generate and Register the NSX-T TLS Certificate and Private Key](#)
- [Create an IP Pool for VTEP](#)
- [Configure Transport Zones](#)
- [Configure vSphere Networking for ESXi Hosts](#)
- [Deploy NSX-T Edge Nodes](#)
- [Deploy NSX-T Transport Nodes](#)
- [Create NSX-T Objects for Kubernetes Clusters Provisioned by TKGI](#)
- [Create NSX-T Objects for TKGI Management Plane Components](#)

## Configure the NSX-T Password Interval (Optional)

The default NSX-T password expiration interval is 90 days. After this period, the NSX-T passwords will expire on all NSX-T Manager Nodes and all NSX-T Edge Nodes. To avoid this, you can extend or remove the password expiration interval, or change the password if needed.



**Note:** For existing Tanzu Kubernetes Grid Integrated Edition deployments, anytime the NSX-T password is changed you must update the BOSH and PKS tiles with the new passwords. See [Adding Infrastructure Password Changes to the Tanzu Kubernetes Grid Integrated Edition Tile](#) for more information.

## Update the NSX-T Manager Password and Password Interval

To update the NSX Manager password, perform the following actions on **one** of the NSX Manager nodes. The changes will be propagated to all NSX Manager nodes.

### SSH into the NSX Manager Node

To manage user password expiration, you use the CLI on one of the NSX Manager nodes.

To access a NSX Manager node, from Unix hosts use the command `ssh USERNAME@IP_ADDRESS_OF_NSX_MANAGER.`

For example:

```
ssh admin@10.196.188.22
```

On Windows, use Putty and provide the IP address for NSX Manager. Enter the user name and password that you defined during the installation of NSX-T.

### Retrieve the Password Expiration Interval

To retrieve the password expiration interval, use the following command:

```
get user USERNAME password-expiration
```

For example:

```
NSX CLI (Manager, Policy, Controller 3.0.0.0.0.15946739). Press ? for help or enter: h
elp
nsx-mgr-1> get user admin password-expiration
Password expires 90 days after last change
```

### Update the Admin Password

To update the user password, use the following command:

```
set user USERNAME password NEW-PASSWORD old-password OLD-PASSWORD
```

For example:

```
set user admin password my-new-pwd old-password my-old-pwd
```

### Set the Admin Password Expiration Interval

To set the password expiration interval, use the following command:

```
set user USERNAME password-expiration PASSWORD-EXPIRATION
```

For example, the following command sets the password expiration interval to 120 days:

```
set user admin password-expiration 120
```

### Remove the Admin Password Expiration Interval

To remove password expiration, use the following command:

```
clear user USERNAME password-expiration
```

For example:

```
clear user admin password-expiration
```

To verify:

```
nsx-mgr-1> clear user admin password-expiration
nsx-mgr-1> get user admin password-expiration
Password expiration not configured for this user
```

### Update the Password for NSX Edge Nodes

To update the NSX Edge Node password, perform the following actions on **each** NSX Edge Node.



**Note:** Unlike the NSX-T Manager nodes, you must update the password or password interval on each Edge Node.

## Enable SSH

SSH on the Edge Node is deactivated by default. You must activate SSH on the Edge Node using the Console from vSphere.

```
start service ssh
set service ssh start-on-boot
```

## SSH to the NSX Edge Node

For example:

```
ssh admin@10.196.188.25
```

## Get the Password Expiration Interval for the Edge Node

For example:

```
nsx-edge> get user admin password-expiration
Password expires 90 days after last change
```

## Update the User Password for the Edge Node

For example:

```
nsx-edge> set user admin password my-new-pwd old-password my-old-pwd
```

## Set the Password Expiration Interval

For example, the following command sets the password expiration interval to 120 days:

```
nsx-edge> set user admin password-expiration 120
```

## Remove the Password Expiration Interval

For example:

```
NSX CLI (Edge 3.0.0.0.0.15946012). Press ? for command list or enter: help
nsx-edge-2> get user admin password-expiration
Password expires 90 days after last change. Current password will expire in 7 days.

nsx-edge-2> clear user admin password-expiration
nsx-edge-2> get user admin password-expiration
```

Password expiration not configured for this user

## Next Step

Once you have completed the installation of NSX-T v3.0, return to the TKGI installation workflow and proceed with the next phase of the process. See [Install Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T Using Ops Manager](#).

## Deploying Ops Manager with VMware NSX for Tanzu Kubernetes Grid Integrated Edition

This topic provides instructions for deploying Ops Manager on VMware vSphere with NSX-T integration for use with VMware Tanzu Kubernetes Grid Integrated Edition.

### Prerequisites

Before deploying Ops Manager with NSX-T for Tanzu Kubernetes Grid Integrated Edition, you must have completed the following tasks:

- [Preparing to Install Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T Data Center](#)
- Depending on your NSX-T version:
  - [NSX-T v3.0: Installing and Configuring NSX-T Data Center v3.0 for Tanzu Kubernetes Grid Integrated Edition](#)
  - [NSX-T v2.5: See the v1.7 documentation:](#)
    - [Installing and Configuring NSX-T Data Center v2.5 for Enterprise PKS and the sequence of topics it includes.](#)
    - [Creating the Tanzu Kubernetes Grid Integrated Edition Management Plane](#)
    - [Create Tanzu Kubernetes Grid Integrated Edition Compute Plane](#)

In addition, review the supported Ops Manager versions for Tanzu Kubernetes Grid Integrated Edition. See [VMware Tanzu Network](#).

Review the known issues for your version of Ops Manager. See one of the following:

- [Ops Manager v2.9 Release Notes](#)
- [Ops Manager v2.10 Release Notes](#)

## Step 1: Generate SSH Key Pair

You cannot deploy Ops Manager without adding a public SSH key in the appropriate field of the **Customize Template** screen. If you do not add a public SSH key, Ops Manager shuts down automatically because it cannot find a key and might enter a reboot loop. For more information, see [Passwords Not Supported for Ops Manager VM on vSphere](#) in the Ops Manager v2.6 release notes.

For instructions on generating the required SSH key pair for installing Ops Manager, refer to the following KB article: [Generate an SSH key pair for installing Ops Manager v2.6 on vSphere](#).

When you add the key value to the **Public SSH Key** field, you must enter the entire public key similar to the format required for [authorized\\_keys](#). For example, the format required is similar to the following:

```
ssh-rsa AAAAB3NzaC1yc2EAAAQJQAAAQEAnZBapWsER/E01hLYvV/rkZe78mUBueZGHx1kw+ByfNbLoA385C
m72L+6qq40yOIH6R42nHN/bynbEHD4Ptess4/s21rLJtTzEWgH9XYnId4sE5f+QTFd2kRtTzZcu8WvFudEIyCI
WjO+o9yvPETs05dE1/3KDn+t9uXxiszrG9Ycb2uNNpmDES+ohm9BQQFmpwFnao+UuQbRXLCcQ3SoE3Ai5Z90+3
PBwm0IByx87/dUuqvVISAJ8yGu2hJobx9PPStFERtUsfx5x+WIu9XIkr15tzxgH9hBDsOS9cVUYJ7kKUUf1yyr
o6ocHyu6TWHJHSJL8Z2FULxMPpqdn+8Xw== my-key
```

## Step 2: Deploy Ops Manager for Tanzu Kubernetes Grid Integrated Edition

1. Before starting, refer to the [Tanzu Kubernetes Grid Integrated Edition Release Notes](#) for supported Ops Manager versions for Tanzu Kubernetes Grid Integrated Edition.
2. Before starting, refer to the known issues in the [Ops Manager Release v2.9 Release Notes](#) or the [Ops Manager Release v2.10 Release Notes](#).
3. Download the Ops Manager for vSphere installation file from the [VMware Tanzu Network](#).
  1. Open a browser to the [Ops Manager](#) download page on the VMware Tanzu Network.
  2. Use the dropdown menu to select the supported Ops Manager release.
  3. Select the **Ops Manager for vSphere** download option. This downloads the **Ops Manager for vSphere** VM template as an OVA file.
4. Log into vCenter using the vSphere Client (HTML5) to deploy the Ops Manager OVA.
5. Select the Resource Pool defined for the Tanzu Kubernetes Grid Integrated Edition Management Plane. See [Create Management Plane](#) in *Installing and Configuring NSX-T Data Center v3.0 for TKG* if you have not defined the Tanzu Kubernetes Grid Integrated Edition Management Resource Pool.
6. Right click the Tanzu Kubernetes Grid Integrated Edition Management Plane Resource Pool and select **Deploy OVF Template**.
7. At the **Select an OVF template** screen:
  - ◊ Click **Browse**.
  - ◊ Select the Ops Manager OVA file you downloaded and click **Open**.
  - ◊ Click **Next**.

## Deploy OVF Template

**1 Select an OVF template**

- 2 Select a name and folder
- 3 Select a compute resource
- 4 Review details
- 5 Select storage
- 6 Ready to complete

**Select an OVF template**

Select an OVF template from remote URL or local file system

Enter a URL to download and install the OVF package from the Internet, or browse to a location accessible from your computer, such as a local hard drive, a network share, or a CD/DVD drive.

URL

http | https://remoteserver-address/filetodeploy.ovf | .ova

Local file

**Choose Files** ops-manager-vsp...-build.229.ova

CANCEL

BACK

NEXT

8. At the **Select Name and folder** screen, enter a name for the Ops Manager VM (or use the default name), select the **Datacenter**, and click **Next**

## Deploy OVF Template

✓ 1 Select an OVF template

2 Select a name and folder

3 Select a compute resource

4 Review details

5 Select storage

6 Ready to complete

Select a name and folder

Specify a unique name and target location

Virtual machine name:

Select a location for the virtual machine.

10.197.79.141

vSAN Datacenter

CANCEL

BACK

NEXT

9. At the **Select a compute resource** screen, select the Tanzu Kubernetes Grid Integrated Edition **Resource Pool** or **Cluster object** and click **Next**.

## Deploy OVF Template

- ✓ 1 Select an OVF template
- ✓ 2 Select a name and folder
- 3 Select a compute resource**

- 4 Review details
- 5 Select storage
- 6 Ready to complete

### Select a compute resource

Select the destination compute resource for this operation

vSAN Datacenter

vSAN Cluster

- 10.197.145.51
- 10.197.145.52
- 10.197.145.53
- 10.197.145.54

- > K8S
- > MGMT
- > PKS

### Compatibility

✓ Compatibility checks succeeded.

CANCEL

BACK

NEXT

10. At the **Review details** screen, confirm the configuration up to this point and click **Next**.

## Deploy OVF Template

- ✓ 1 Select an OVF template
- ✓ 2 Select a name and folder
- ✓ 3 Select a compute resource

### Review details

Verify the template details.

### 4 Review details

- 5 Select storage
- 6 Select networks
- 7 Customize template
- 8 Ready to complete

|               |                                                                                  |
|---------------|----------------------------------------------------------------------------------|
| Publisher     | No certificate present                                                           |
| Product       | Ops Manager                                                                      |
| Version       | 2.7.7-build.229                                                                  |
| Vendor        | Pivotal                                                                          |
| Description   | Pivotal Ops Manager installs and manages Pivotal Platform products and services. |
| Download size | 4.6 GB                                                                           |
| Size on disk  | Unknown (thin provisioned)<br>160.0 GB (thick provisioned)                       |

CANCEL

BACK

NEXT

11. At the **Select Storage** screen, select the desired Datastore, and click **Next**.

## Deploy OVF Template

**5 Select storage**

| Name          | Capacity | Provisioned | Free    |
|---------------|----------|-------------|---------|
| vsanDatastore | 6.99 TB  | 1.39 TB     | 6.53 TB |

**Compatibility**

✓ Compatibility checks succeeded.

**CANCEL** **BACK** **NEXT**



**Warning:** Ops Manager requires a Director VM with at least 8 GB memory.  
For more information, see [Provisioning a Virtual Disk in vSphere](#).

12. At the **Select Networks** screen:

- If you are using vSphere 6.7, select the Tanzu Kubernetes Grid Integrated Edition Management T1 Logical Switch that you defined when [Create Management Plane](#) in *Installing and Configuring NSX-T Data Center v3.0 for TKGI*, and click **Next**.

## Deploy OVF Template

- ✓ 1 Select an OVF template
- ✓ 2 Select a name and folder
- ✓ 3 Select a compute resource
- ✓ 4 Review details
- ✓ 5 Select storage

### 6 Select networks

7 Customize template

8 Ready to complete

#### Select networks

Select a destination network for each source network.

| Source Network | Destination Network |
|----------------|---------------------|
| Network 1      | LS-MGMT-PKS         |

1 items

#### IP Allocation Settings

IP allocation: Static - Manual

IP protocol: IPv4

CANCEL

BACK

NEXT

- ◊ If you are using vSphere 6.5, see [Network Selection for vSphere v6.5](#).

13. At the **Customize template** screen, enter the following information, and click **Next**.

- ◊ **IP Address:** The IP address of the Ops Manager network interface, for example `10.0.0.2` (assuming non-routable NAT-mode).
- ◊ **Netmask:** The network mask for Ops Manager, for example, `255.255.255.0`.
- ◊ **Default Gateway:** The default gateway for Ops Manager to use, for example `10.0.0.1` (assuming non-routable NAT-mode).
- ◊ **DNS:** One or more DNS servers for the Ops Manager VM to use, for example `10.14.7.1`.
- ◊ **NTP Servers:** The IP address of one or more NTP servers for Ops Manager, for example `10.113.60.176`.
- ◊ **Public SSH Key:** (Required) Enter the public SSH key to allow SSH access to the Ops Manager VM. You must enter the entire the public SSH key in the expected format. See [Step 1: Generate SSH Key Pair](#).
- ◊ **Custom hostname:** The hostname for the Ops Manager VM, for example `ops-manager`.

## Deploy OVF Template

- ✓ 1 Select an OVF template
- ✓ 2 Select a name and folder
- ✓ 3 Select a compute resource
- ✓ 4 Review details
- ✓ 5 Select storage
- ✓ 6 Select networks
- 7 Customize template**

8 Ready to complete

### Customize template

Customize the deployment properties of this software solution.

All properties have valid values X

▼ Uncategorized 1 settings

IP Address The IP address for the Ops Manager. Leave blank if DHCP is desired.  
10.0.0.2

▼ Uncategorized 1 settings

Netmask The netmask for the Ops Manager's network. Leave blank if DHCP is desired.  
255.255.255.0

▼ Uncategorized 1 settings

Default Gateway The default gateway address for the Ops Manager's network. Leave blank if DHCP is desired.  
10.0.0.1

▼ Uncategorized 1 settings

CANCEL

## Deploy OVF Template

- ✓ 1 Select an OVF template
- ✓ 2 Select a name and folder
- ✓ 3 Select a compute resource
- ✓ 4 Review details
- ✓ 5 Select storage
- ✓ 6 Select networks
- 7 Customize template**

8 Ready to complete

▼ Uncategorized 1 settings

DNS The domain name servers for the Ops Manager (comma separated). Leave blank if DHCP is desired.  
10.142.7.1,10.142.7.2

▼ Uncategorized 1 settings

NTP Servers Comma-delimited list of NTP servers  
10.128.243.13,10.128.243.14

▼ Uncategorized 1 settings

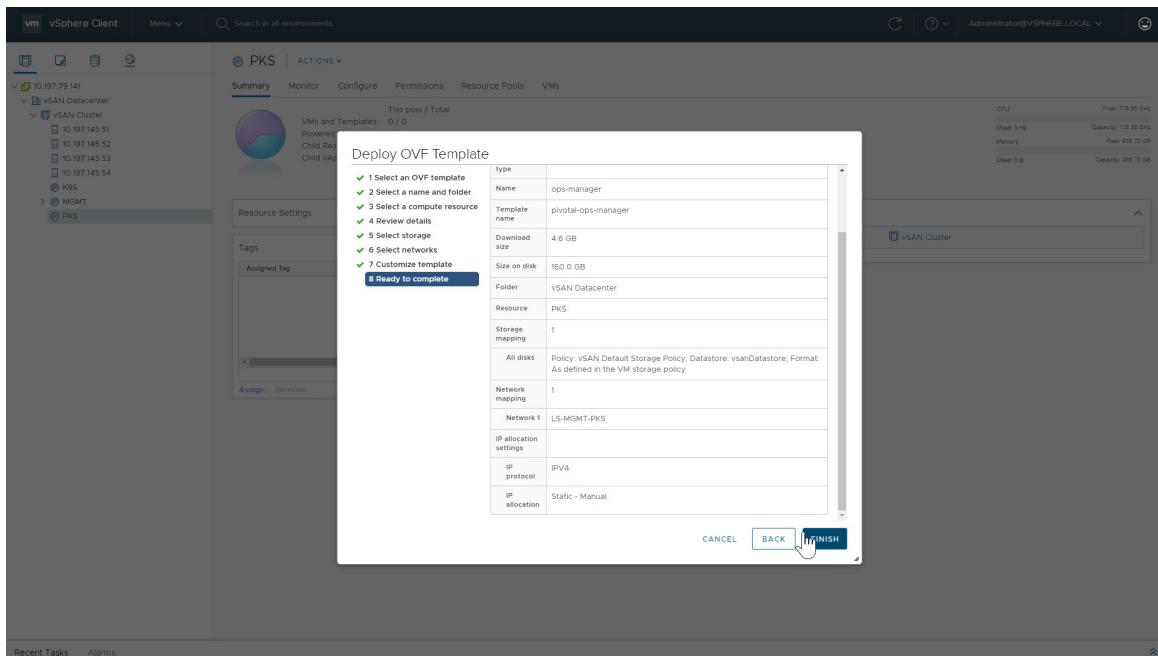
Public SSH Key The Public SSH Key is used to allow SSHing into the Ops Manager with your private ssh key. The username is 'ubuntu'.  
ssh-rsa AAAAB3NzaC1yc2E=

▼ Uncategorized 1 settings

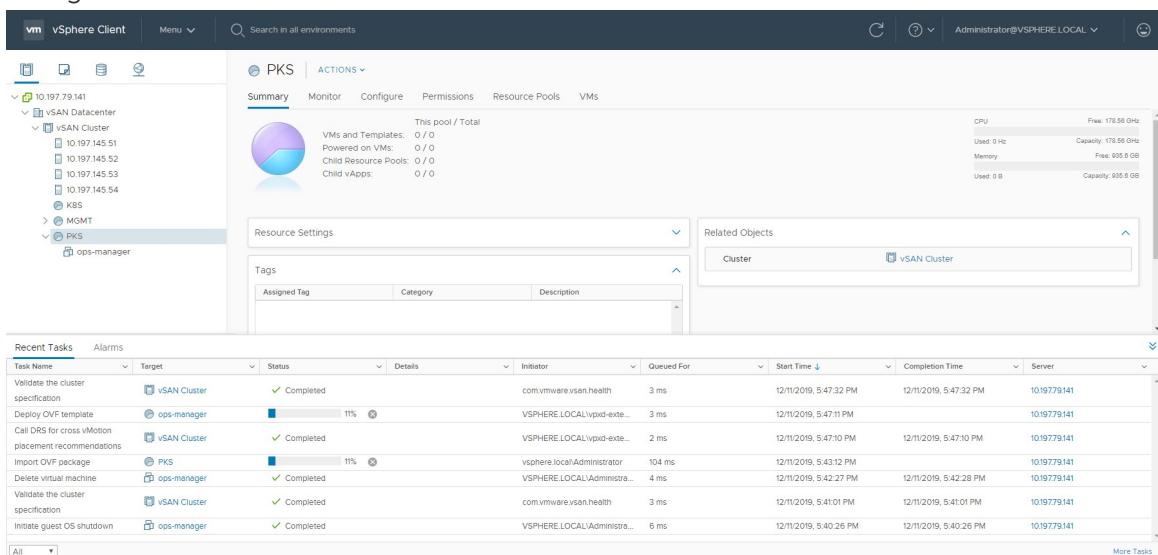
Custom Hostname This will be set as the hostname on the VM. Default: 'pivotal-ops-manager'.  
ops-manager

CANCEL

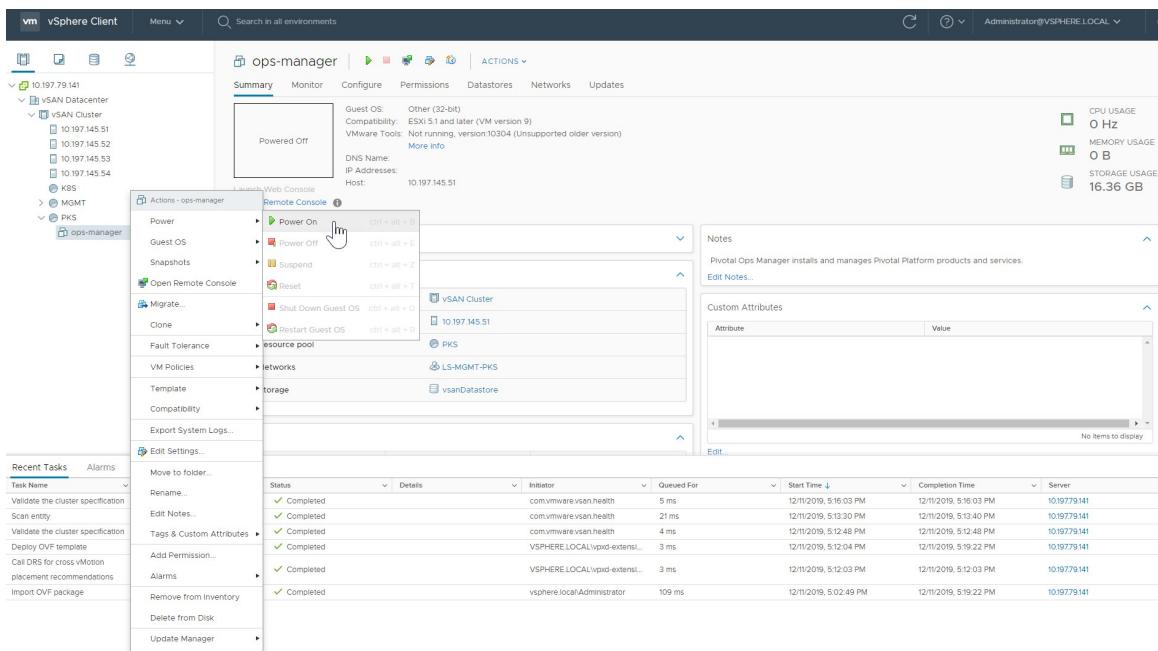
14. At the **Ready to complete** screen, review the configuration settings and click **Finish**. This action begins the OVA import and deployment process.



15. Use the **Recent Tasks** panel at the bottom of the vCenter dashboard to check the progress of the OVA import and deployment. If the import or deployment is unsuccessful, check the configuration for errors.



16. Right-click the Ops Manager VM and click **Power On**.



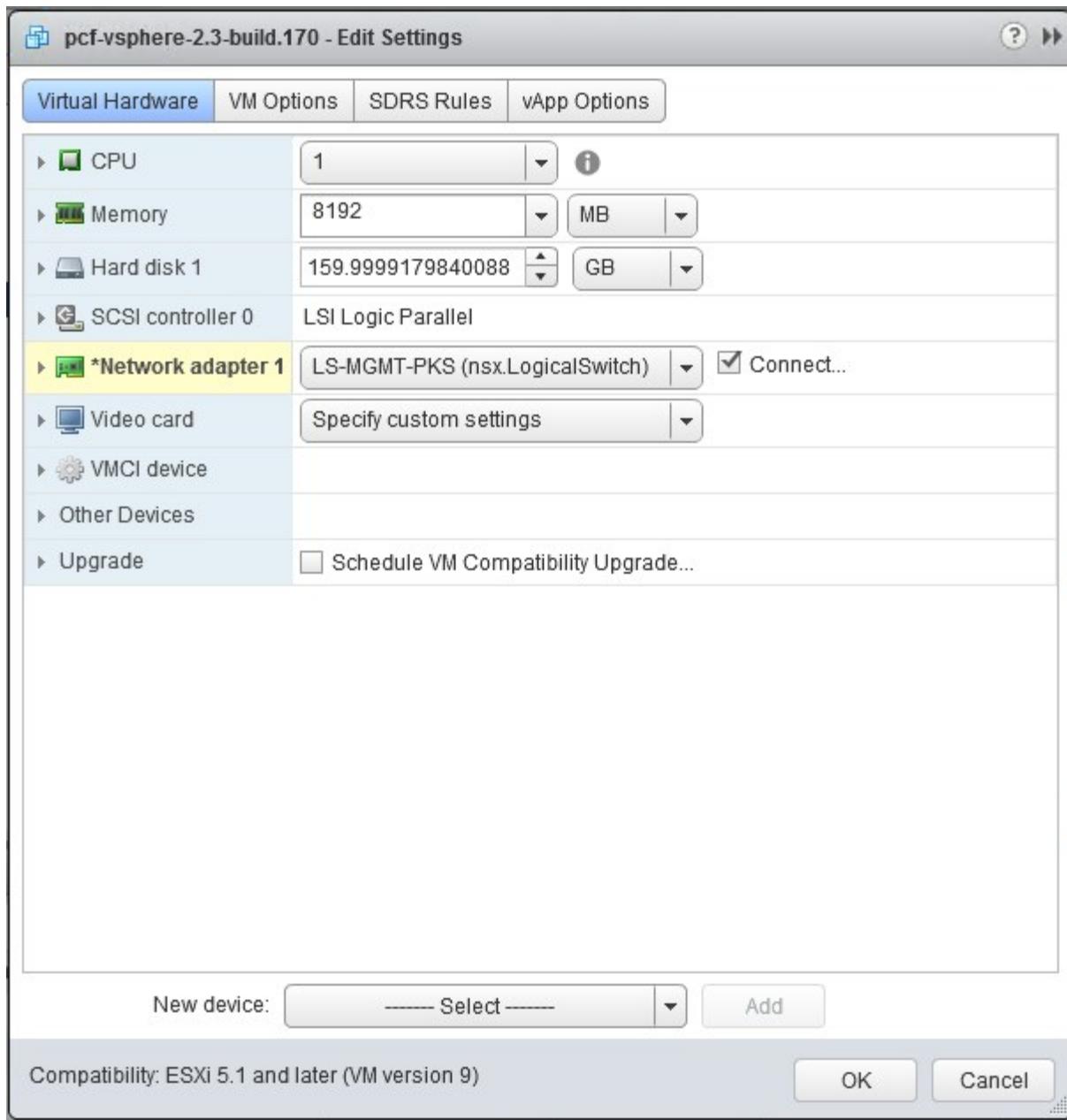
## Network Selection for vSphere v6.5

With VMware vCenter Server 6.5, when initially deploying the Ops Manager OVA, you cannot connect to an NSX-T logical switch. You must first connect to a vSphere Standard (vSS) or vSphere Distributed Switch (vDS). After the OVA deployment is complete, before powering on the Ops Manager VM, connect the network interface to the NSX-T logical switch. The instructions below describe how to do this. This issue is resolved in VMware vCenter Server 6.7. For more information about this issue, see the [VMware Knowledge Base](#).

If you are using vSphere 6.5, at the **Select Networks** screen, select a vSS or vDS port-group such as the standard **VM Network**, and click **Next**.

Complete the remaining deployment steps as described above.

After the OVA deployment completes successfully, right-click the Ops Manager VM and select **Edit Settings**. Change the vNIC connection to use the `nsx.LogicalSwitch` that is defined for the TKGI Management Plane, for example `LS-MGMT-TKGI`.



## Step 3: Configure Ops Manager for Tanzu Kubernetes Grid Integrated Edition

The first time that you start Ops Manager, you must select an authentication system. These instructions use **Internal Authentication**. For configuration details for the **SAML** and **LDAP** options, see [Logging Into Ops Manager with Auth](#) in the Ops Manager documentation.

1. If you are using the [NAT deployment topology](#), create a DNAT rule that maps the Ops Manager private IP to a routable IP. See [Create Management Plane](#) in *Installing and Configuring NSX-T Data Center v3.0 for TKGI* for instructions.
2. If you are using the [No-NAT deployment topology](#), create a DNS entry for the routable IP address that you set for Ops Manager. Use FQDN to log into Ops Manager.



**Note:** Ops Manager security features require you to create a fully qualified domain name to access Ops Manager. See [Installing Ops Manager on](#)

## vSphere.

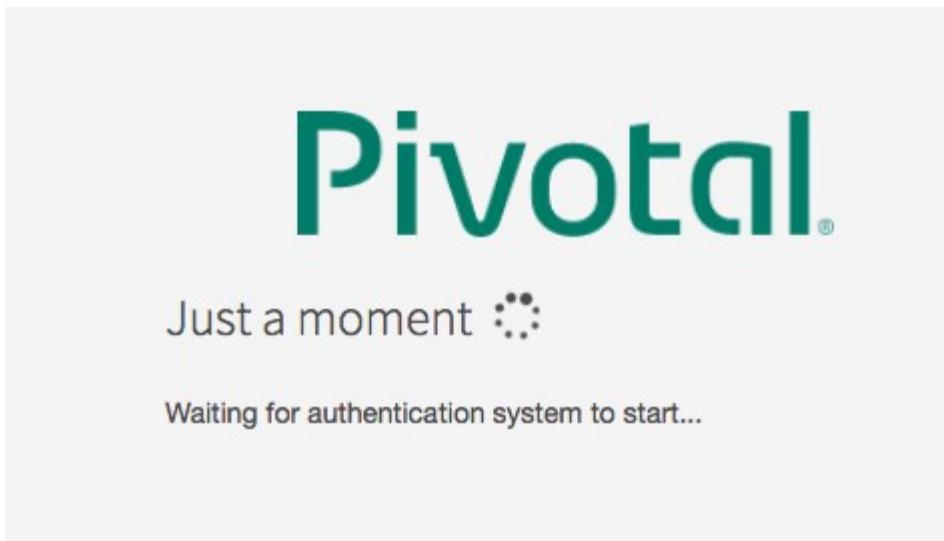
- Navigate to the IP address (NAT mode) or FQDN (No-NAT mode) of your Ops Manager VM in a web browser. The “Welcome to Ops Manager” page should appear.



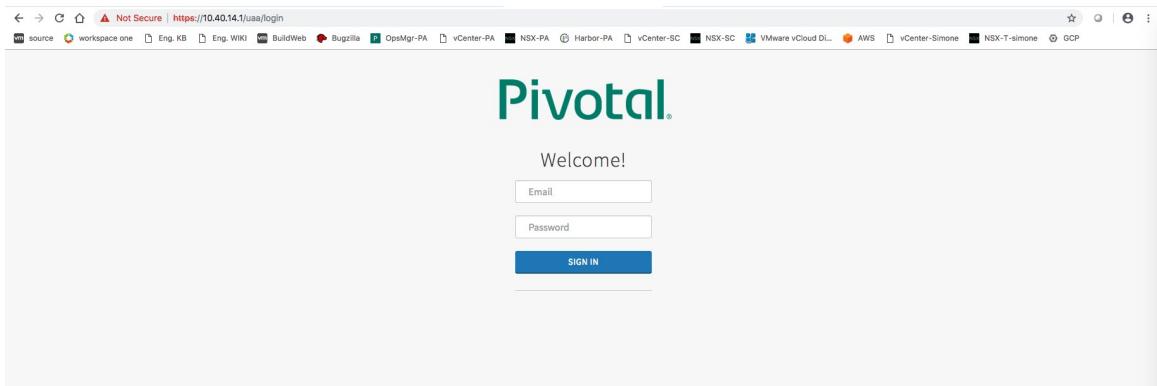
**Note:** It is normal to experience a brief delay before the interface is accessible while the web server and VM start up.

PCF Ops Manager v2.6. ©2013-2019 Pivotal Software, Inc. All Rights Reserved. [API Documentation](#) | [End User License Agreement](#)

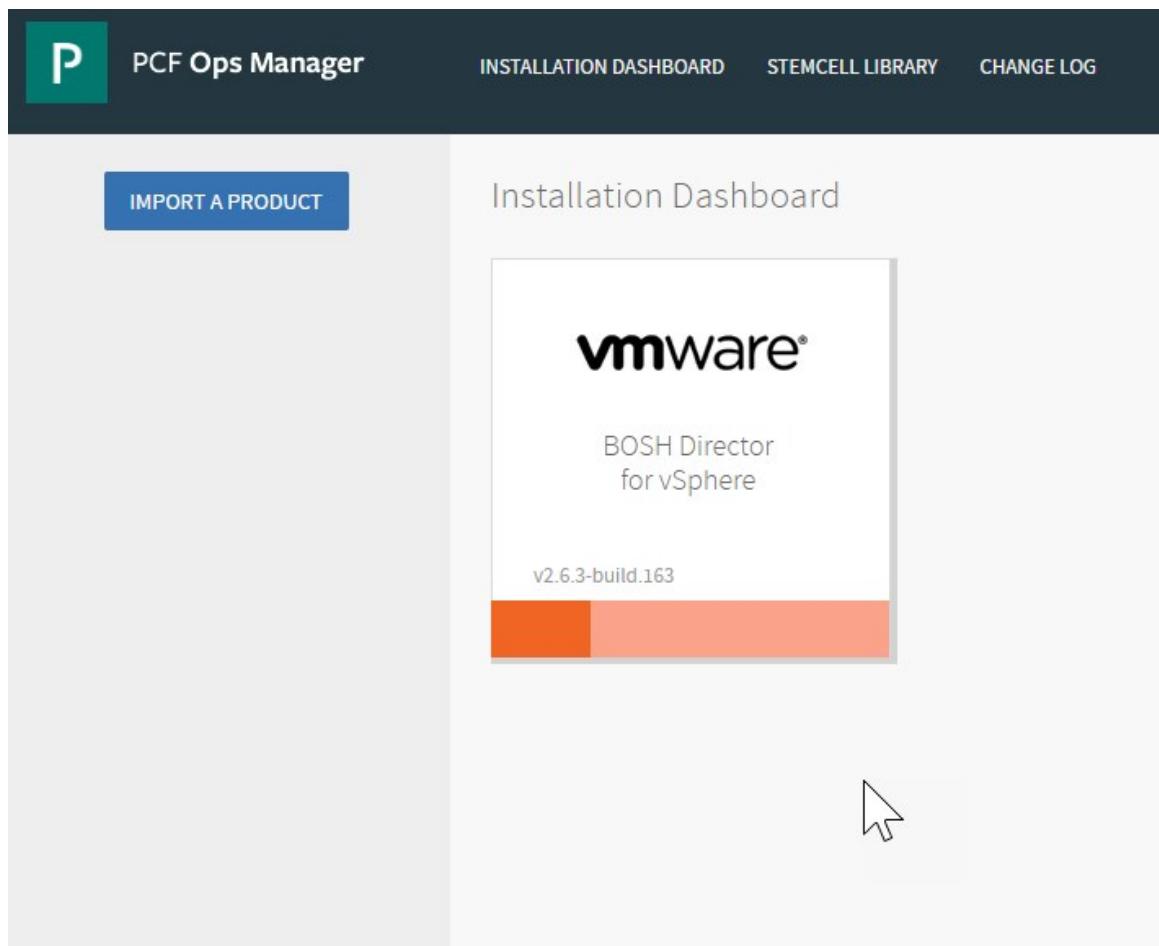
- Select **Internal Authentication** and provide the following information:
  - Username**, **Password**, and **Password confirmation** to create a user with administrative privileges.
  - Decryption passphrase** and the **Decryption passphrase confirmation**. This passphrase encrypts the Ops Manager datastore, and is not recoverable.
  - HTTP proxy** or **HTTPS proxy**, follow the instructions in [Configuring Proxy Settings for the BOSH CPI](#).
- Read the **End User License Agreement**, and select the checkbox to accept the terms.
- Click **Setup Authentication**. It takes a few minutes to initialize the database.



7. Log in to Ops Manager with the username and password that you created.



8. Verify success. You should be able to log in, and you should see the BOSH Director tile is present and ready for configuration, indicated by the orange color.



## Next Step

After you complete this procedure, follow the instructions in [Generate and Register the NSX-T Management SSL Certificate and Private Key](#) in *Installing and Configuring NSX-T Data Center v3.0 for TKG1*.

## Configuring BOSH Director with VMware NSX for Tanzu Kubernetes Grid Integrated Edition

This topic describes how to configure BOSH Director for vSphere with NSX-T integration for Tanzu Kubernetes Grid Integrated Edition (TKGI).

## Prerequisites

Before you begin this procedure, ensure that you have successfully completed all preceding steps for installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T, including:

- Preparing to Install Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T Data Center
- Hardware Requirements for Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T
- Depending on your NSX-T version:

- [NSX-T v3.0: Installing and Configuring NSX-T Data Center v3.0 for Tanzu Kubernetes Grid Integrated Edition](#)
- [NSX-T v2.5: See the v1.7 documentation:](#)
  - [Installing and Configuring NSX-T Data Center v2.5 for Enterprise PKS and the sequence of topics it includes](#)
  - [Creating the Tanzu Kubernetes Grid Integrated Edition Management Plane](#)
  - [Create Tanzu Kubernetes Grid Integrated Edition Compute Plane](#)
- [Deploying Ops Manager with NSX-T for Tanzu Kubernetes Grid Integrated Edition](#)
- [Generate and Register the NSX-T Management SSL Certificate and Private Key in \*Installing and Configuring NSX-T Data Center v3.0 for TKGI\*](#)

## How Ops Manager Accesses NSX-T Manager

To create, delete, and modify NSX-T networking resources, Ops Manager tiles and APIs use a VMware NSX Manager account with the Enterprise Administrator role and permissions.

Users configure Ops Manager to authenticate to NSX Manager for different purposes in different tiles:

- **Tanzu Kubernetes Grid Integrated Edition tile**

The Tanzu Kubernetes Grid Integrated Edition tile uses NSX Manager to create load balancers, providing a Kubernetes service described in the [Create an External Load Balancer](#) section of the Kubernetes documentation.

To configure the **Tanzu Kubernetes Grid Integrated Edition** tile's authentication to NSX Manager, see the topic [Generating and Registering the NSX Manager Superuser Principal Identity Certificate and Key](#).

- **BOSH Director for vSphere tile**

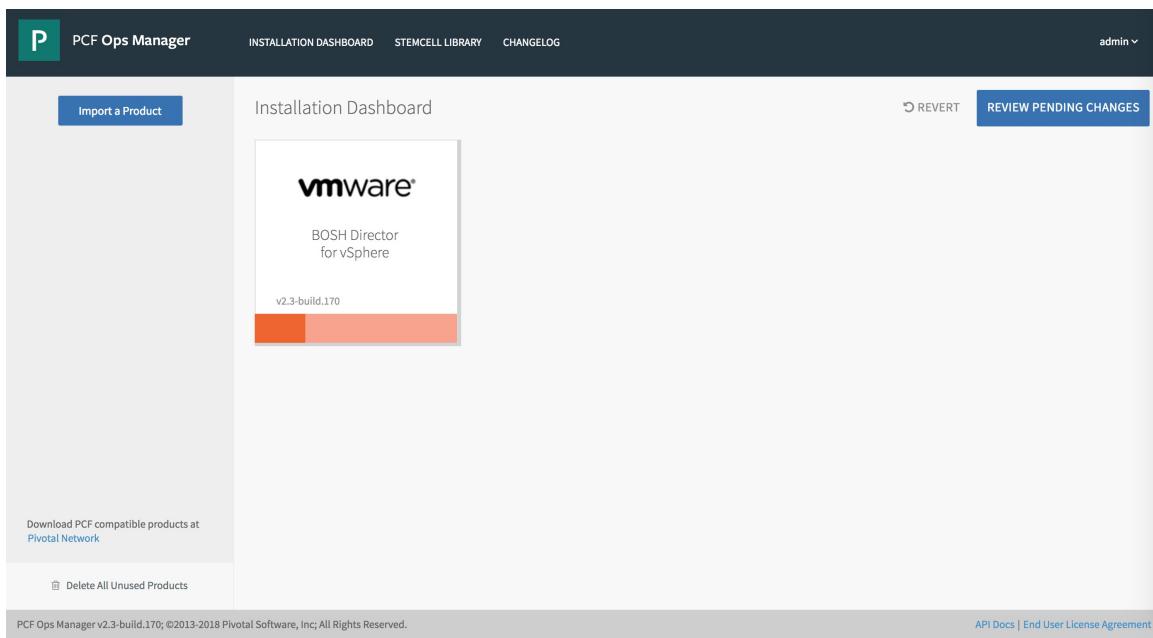
The **BOSH Director for vSphere** tile uses NSX Manager to configure networking and security for external-facing Ops Manager component VMs, such as VMware Tanzu Application Service for VMs routers.

To configure the **BOSH Director for vSphere** tile's authentication to NSX Manager, see [Configure vCenter for Tanzu Kubernetes Grid Integrated Edition](#), below.

## Step 1: Open the BOSH Director Tile

To configure BOSH Director:

1. Log in to Ops Manager with the username and password credentials that you set up in [Configure Ops Manager for Tanzu Kubernetes Grid Integrated Edition](#).
  2. Click the **BOSH Director for vSphere** tile.
-



## Step 2: Configure vCenter for Tanzu Kubernetes Grid Integrated Edition

To configure BOSH Director with your vCenter settings:

1. Select **vCenter Config** in BOSH Director.

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |                                                          |                                                 |                                       |                                               |                                           |                                         |                                                  |       |                                         |               |                                           |                   |                                                          |                   |                                        |                  |                                         |                    |                                   |                                              |                                                |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------|-------------------------------------------------|---------------------------------------|-----------------------------------------------|-------------------------------------------|-----------------------------------------|--------------------------------------------------|-------|-----------------------------------------|---------------|-------------------------------------------|-------------------|----------------------------------------------------------|-------------------|----------------------------------------|------------------|-----------------------------------------|--------------------|-----------------------------------|----------------------------------------------|------------------------------------------------|
| <input checked="" type="radio"/> vCenter Config                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | <input type="radio"/> Director Config                    | <input type="radio"/> Create Availability Zones | <input type="radio"/> Create Networks | <input type="radio"/> Assign AZs and Networks | <input checked="" type="radio"/> Security | <input checked="" type="radio"/> Syslog | <input checked="" type="radio"/> Resource Config |       |                                         |               |                                           |                   |                                                          |                   |                                        |                  |                                         |                    |                                   |                                              |                                                |
| <b>vCenter Config</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |                                                          |                                                 |                                       |                                               |                                           |                                         |                                                  |       |                                         |               |                                           |                   |                                                          |                   |                                        |                  |                                         |                    |                                   |                                              |                                                |
| <table border="1"> <tr> <td>Name*</td> <td><input type="text" value="vCenter-PA"/></td> </tr> <tr> <td>vCenter Host*</td> <td><input type="text" value="10.40.206.61"/></td> </tr> <tr> <td>vCenter Username*</td> <td><input type="text" value="administrator@vsphere.local"/></td> </tr> <tr> <td>vCenter Password*</td> <td><input type="password" value="*****"/></td> </tr> <tr> <td>Datacenter Name*</td> <td><input type="text" value="Datacenter"/></td> </tr> <tr> <td>Virtual Disk Type*</td> <td><input type="text" value="thin"/></td> </tr> <tr> <td>Ephemeral Datastore Names (comma delimited)*</td> <td><input type="text" value="NFS-LAB-DATASTORE"/></td> </tr> </table> |                                                          |                                                 |                                       |                                               |                                           |                                         |                                                  | Name* | <input type="text" value="vCenter-PA"/> | vCenter Host* | <input type="text" value="10.40.206.61"/> | vCenter Username* | <input type="text" value="administrator@vsphere.local"/> | vCenter Password* | <input type="password" value="*****"/> | Datacenter Name* | <input type="text" value="Datacenter"/> | Virtual Disk Type* | <input type="text" value="thin"/> | Ephemeral Datastore Names (comma delimited)* | <input type="text" value="NFS-LAB-DATASTORE"/> |
| Name*                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | <input type="text" value="vCenter-PA"/>                  |                                                 |                                       |                                               |                                           |                                         |                                                  |       |                                         |               |                                           |                   |                                                          |                   |                                        |                  |                                         |                    |                                   |                                              |                                                |
| vCenter Host*                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | <input type="text" value="10.40.206.61"/>                |                                                 |                                       |                                               |                                           |                                         |                                                  |       |                                         |               |                                           |                   |                                                          |                   |                                        |                  |                                         |                    |                                   |                                              |                                                |
| vCenter Username*                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          | <input type="text" value="administrator@vsphere.local"/> |                                                 |                                       |                                               |                                           |                                         |                                                  |       |                                         |               |                                           |                   |                                                          |                   |                                        |                  |                                         |                    |                                   |                                              |                                                |
| vCenter Password*                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          | <input type="password" value="*****"/>                   |                                                 |                                       |                                               |                                           |                                         |                                                  |       |                                         |               |                                           |                   |                                                          |                   |                                        |                  |                                         |                    |                                   |                                              |                                                |
| Datacenter Name*                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | <input type="text" value="Datacenter"/>                  |                                                 |                                       |                                               |                                           |                                         |                                                  |       |                                         |               |                                           |                   |                                                          |                   |                                        |                  |                                         |                    |                                   |                                              |                                                |
| Virtual Disk Type*                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         | <input type="text" value="thin"/>                        |                                                 |                                       |                                               |                                           |                                         |                                                  |       |                                         |               |                                           |                   |                                                          |                   |                                        |                  |                                         |                    |                                   |                                              |                                                |
| Ephemeral Datastore Names (comma delimited)*                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               | <input type="text" value="NFS-LAB-DATASTORE"/>           |                                                 |                                       |                                               |                                           |                                         |                                                  |       |                                         |               |                                           |                   |                                                          |                   |                                        |                  |                                         |                    |                                   |                                              |                                                |
| <small>PCF Ops Manager v2.3-build.170; ©2013-2018 Pivotal Software, Inc; All Rights Reserved.</small>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |                                                          |                                                 |                                       |                                               |                                           |                                         |                                                  |       |                                         |               |                                           |                   |                                                          |                   |                                        |                  |                                         |                    |                                   |                                              |                                                |
| <small>API Docs   End User License Agreement</small>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |                                                          |                                                 |                                       |                                               |                                           |                                         |                                                  |       |                                         |               |                                           |                   |                                                          |                   |                                        |                  |                                         |                    |                                   |                                              |                                                |

2. Enter the following information:

- ❖ **Name:** A name that you provide for your vCenter configuration. This field is used to identify the datacenter configuration in Ops Manager if you are configuring multiple datacenters.
- ❖ **vCenter Host:** The hostname of the vCenter that manages ESXi/vSphere.



**Note:** The FQDN for the vCenter Server cannot contain uppercase

letters.

- ❖ **vCenter Username:** A vCenter username with create and delete privileges for virtual machines (VMs) and folders.
- ❖ **vCenter Password:** The password for the vCenter user specified above.
- ❖ **Datacenter Name:** The name of the datacenter as it appears in vCenter.
- ❖ **Virtual Disk Type:** The Virtual Disk Type to provision for all VMs. For guidance on selecting a virtual disk type, see [vSphere Virtual Disk Types](#).
- ❖ **Ephemeral Datastore Names (comma delimited):** The names of the datastores that store ephemeral VM disks deployed by Ops Manager.
- ❖ **Persistent Datastore Names (comma delimited):** The names of the datastores that store persistent VM disks deployed by Ops Manager.



**Note:** The vSphere datastore type must be Datastore. Tanzu Kubernetes Grid Integrated Edition does not support the use of vSphere Datastore Clusters with or without Storage DRS. For more information, see [Datastores and Datastore Clusters](#) in the vSphere documentation.

3. For Networking, select **NSX Networking**.

Standard vCenter Networking  
 NSX Networking

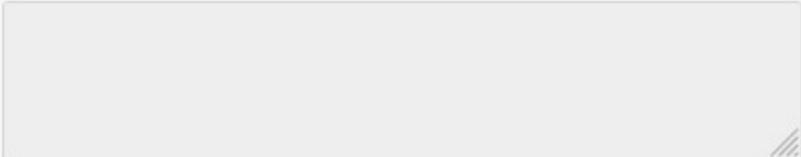
NSX Mode\*  
 NSX-V  
 NSX-T

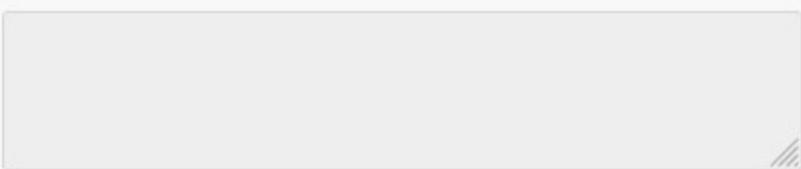
NSX Address\*  
nsxmanager.pks.vmware.local

NSX-T Authentication\*  
 Local User Authentication  
 Certificate Authentication

NSX Username  
admin

NSX Password  
\*\*\*\*\*  
[Change](#)

NSX Manager Principal Identity Certificate  


NSX Manager Principal Identity Private Key  


Use NSX-T Policy API

NSX CA Cert  

```
-----BEGIN CERTIFICATE-----
MIIDwzCCAqgAwIBAgI/GAXqtbkBMMA0GCSqGSIb3DQEBCwUAMHMxDjAiBgNVBAMM
G25zeG1hbmcFnZXlucGtzLnZtd2FyZS5sb2NhDEPMA0GA1UECgwGVk13YXJlMQww
CgYDVQQLANDTkExCzAJBgNVBAYTAiVTMQswCQYDVQQIDAJDQTESMBAGA1UEBwwJ
UGFsbyBBbHRvMB4XDThMDcxNjAzNDg0MVoXDTIyMDcxNjAzNDg0MVowczEkMCIG
-----END CERTIFICATE-----
```

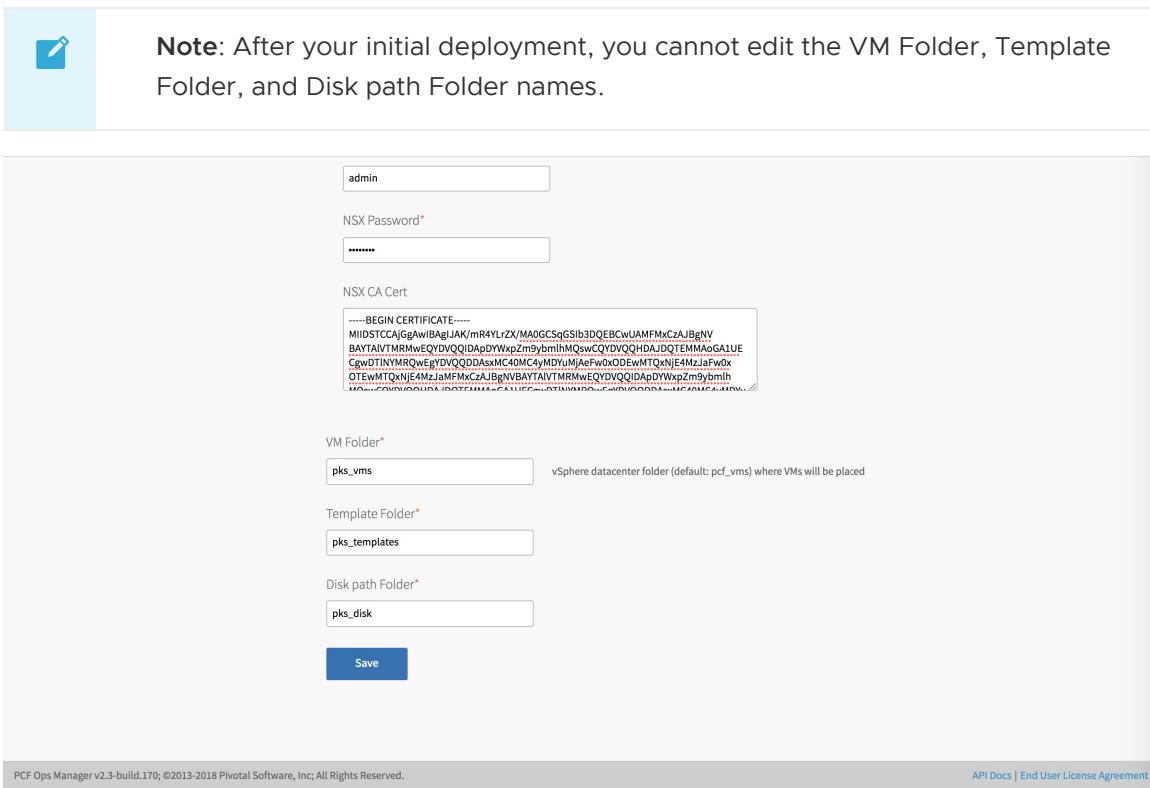


4. Configure NSX networking as follows:

- **NSX Mode:** Select **NSX-T** to use NSX-T networking for dynamically created node networks.
  - **NSX Address:** Enter the FQDN or IP address of the NSX-T Manager, or the VIP address of the NSX-T Management Cluster.
  - **NSX Authentication:** Select the authentication mode, either **Local User Authentication** or **Certificate Authentication**.
  - **NSX Username** and **NSX Password:** If you selected **Local User Authentication**, enter the NSX-T Manager username and password.
  - **NSX Manager Principal Identity Certificate** and **NSX Manager Principal Identity Private Key:** If you selected **Certificate Authentication**, enter the NSX Manager Principal Identity Certificate and Private Key. For more information, see [Generating and Registering the NSX Manager Superuser Principal Identity Certificate and Key](#).
  - **Use NSX-T Policy API:** Select this option to use the NSX-T Policy API instead of the NSX-T Management API. For more information, see [Considerations for Using the NSX-T Policy API with TKGI](#).
  - **NSX CA Cert:** Provide the CA certificate in PEM format that authenticates to the NSX server. Copy the contents of the NSX CA certificate that you generated in [Generate and Register the NSX-T Management SSL Certificate and Private Key](#) to this field.

5. Configure the following folder names:

- **VM Folder:** The vSphere datacenter folder where Ops Manager places VMs.
  - **Template Folder:** The vSphere datacenter folder where Ops Manager places VMs.
  - **Disk path Folder:** The vSphere datastore folder where Ops Manager creates attached disk images. You must not nest this folder.



## 6. Click Save.

PCF Ops Manager - BOSH Director for vSphere

**vCenter Config**

Name\*: vCenter-PA

vCenter Host\*: 10.40.206.61

vCenter Username\*: administrator@vsphere.local

vCenter Password\*:

Add vCenter Config

PCF Ops Manager v2.3-build.170; ©2013-2018 Pivotal Software, Inc. All Rights Reserved.  
https://10.40.14.1/infrastructure/aas\_configurations

API Docs | End User License Agreement

## Step 3: Configure BOSH Director

To configure BOSH Director settings:

- Select **Director Config** in BOSH Director.

Director Config

NTP Servers (comma delimited)\*: 10.113.60.176

JMX Provider IP Address:

Bosh HM Forwarder IP Address:

Enable VM Resurrector Plugin

Enable Post Deploy Scripts

Recreate All VMs  
This will force BOSH to recreate all VMs on the next deploy. Persistent disk will be preserved

Recreate All Persistent Disks  
Checking this box will recreate all Persistent Disks for the Director and all other Tiles

Enable bosh deploy retries  
This will attempt to re-deploy a failed deployment up to 5 times.

PCF Ops Manager v2.3-build.170; ©2013-2018 Pivotal Software, Inc. All Rights Reserved.

API Docs | End User License Agreement

- In the **NTP Servers (comma delimited)** field, enter your NTP server addresses.

**Note:** The NTP server configuration only updates after VM recreation. Ensure that you select the **Recreate VMs deployed by the BOSH Director** checkbox if you modify the value of this field.

- Leave the **JMX Provider IP Address** field blank.

4. Leave the **Bosh HM Forwarder IP Address** field blank.
5. Select the **Enable VM Resurrector Plugin** to enable BOSH Resurrector functionality.
6. Select **Enable Post Deploy Scripts** to run a post-deploy script after deployment. This script allows the job to execute additional commands against a deployment.



**Note:** You must enable post-deploy scripts to install Tanzu Kubernetes Grid Integrated Edition.

7. Select **Recreate VMs deployed by the BOSH Director** to force BOSH to recreate all BOSH-managed VMs on the next deploy. This process does not destroy any persistent disk data.
8. For typical Tanzu Kubernetes Grid Integrated Edition deployments, the default settings for all other BOSH Director configuration parameters are suitable. Optionally you can apply additional configurations to BOSH Director. See [Director Config Pane in Configuring BOSH Director on vSphere](#) in the Ops Manager documentation for details.



**Note:** If you need to be able to remotely access the BOSH Director VM using the BOSH CLI, and you are deploying Tanzu Kubernetes Grid Integrated Edition with NSX-T in a NAT topology, you must provide the **Director Hostname** for BOSH at the time of installation. See [Director Config Pane in Configuring BOSH Director on vSphere](#) in the Ops Manager documentation for details.

9. Click **Save**.

PCF Ops Manager

INSTALLATION DASHBOARD STEMCELL LIBRARY CHANLOGE admin

Settings updated

BOSH Director for vSphere

Settings Status Credentials

vCenter Config Director Config

Director Config

NTP Servers (comma delimited)\*  
10.113.60.176

JMX Provider IP Address

Bosh HM Forwarder IP Address

Enable VM Resurrector Plugin

Enable Post Deploy Scripts

Create Availability Zones

Create Networks

Assign AZs and Networks

Security

Syslog

Resource Config

PCF Ops Manager v2.3-build.170; ©2013-2018 Pivotal Software, Inc; All Rights Reserved. API Docs | End User License Agreement

## Step 4: Create Availability Zones

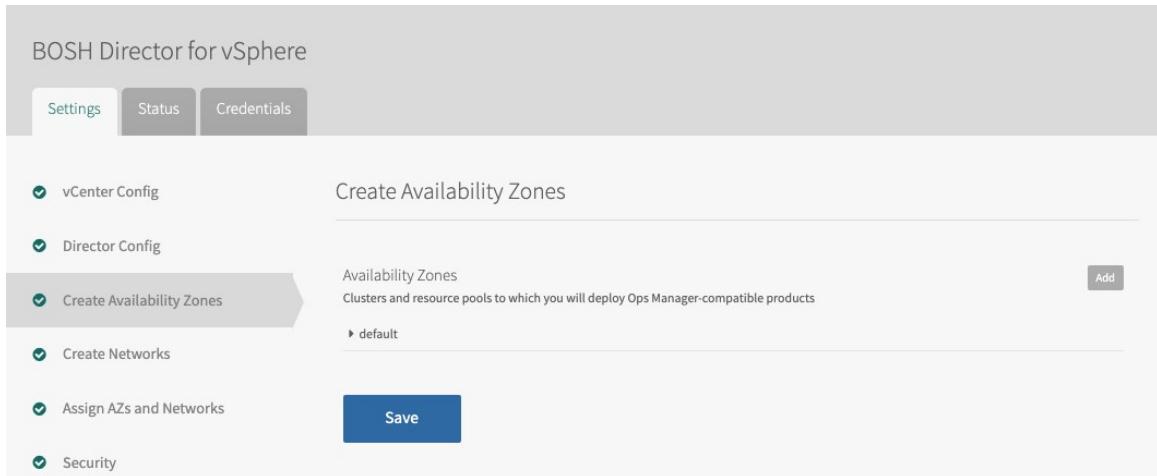
On vSphere with NSX-T, operators define and create Availability Zones (AZs) using vCenter clusters and resource pools. Plans defined in the TKGI tile then use these AZs to enable high availability for TKGI clusters.

The Tanzu Kubernetes Grid Integrated Edition control plane also runs in one of the AZs.

For more information on AZs in TKGI, see [Availability Zones in Tanzu Kubernetes Grid Integrated Edition Architecture](#).

To configure Availability Zones:

1. Select **Create Availability Zones** in BOSH Director.



2. Use the following steps to create one or more Availability Zones for Tanzu Kubernetes Grid Integrated Edition to use:

1. Click **Add** and create the Tanzu Kubernetes Grid Integrated Edition Management AZ.
2. Enter a unique **Name** for the Availability Zone, such as **AZ-MGMT**.
3. Select the IaaS configuration (vSphere/vCenter).
4. Enter the name of an existing vCenter **Cluster** to use as an Availability Zone, such as **COMP-Cluster-1**.
5. Enter the name of the Tanzu Kubernetes Grid Integrated Edition Management **Resource Pool** in the vCenter cluster that you specified above, such as **RP-MGMT-TKGI**. The jobs running in this Availability Zone share the CPU and memory resources defined by the pool.
6. Click **Add Cluster** and create at least one Tanzu Kubernetes Grid Integrated Edition Compute AZ.
7. Specify the **Cluster** and the **Resource Pool**, such as **RP-TKGI-AZ**. Alternatively, specify the **Cluster** and the **Host Group**. See [Using vSphere Host Group](#) for more information.
8. (Optional) If you are using a host group with vSAN stretched clusters, set the **VM-Host Affinity Rule** dropdown to **SHOULD**. This setting maintains high availability by letting TKGI restart VMs in another host group if their AZ fails. TKGI ignores this setting if the vSAN cluster has no host group configured.

For more information, see [Ability to Set the VM-Host Affinity Rule to “Should” for Clusters in vSphere] (<https://docs.pivotal.io/ops-manager/2-9/release-notes.html#vm-affinity-rule>) in the *Ops Manager v2.9 Release Notes*.

9. Add additional clusters as necessary. Click the trash icon to delete a cluster. The first cluster cannot be deleted.

The screenshot shows two identical configuration interfaces for creating availability zones, one on top of the other. Both interfaces have a sidebar on the left with the following items:

- vCenter Config
- Director Config
- Create Availability Zones** (highlighted with a grey arrow pointing to it)
- Create Networks
- Assign AZs and Networks
- Security
- BOSH DNS Config
- Syslog
- Resource Config

The main area is titled "Create Availability Zones" and contains the following fields:

**Availability Zones**  
Clusters and resource pools to which you will deploy Ops Manager-compatible products

default

AZ-MGMT

Name\*  A unique name for this availability zone

IaaS Configuration\*

**Clusters**

Cluster\*

Resource Pool

Host Group

VM-Host Affinity Rule

**Add Cluster**

**Add**

**Create Availability Zones**

Availability Zones  
Clusters and resource pools to which you will deploy Ops Manager-compatible products

default

AZ-COMP-1

Name\*  Delete

IaaS Configuration\*

**Clusters**

Cluster\*

Resource Pool

Host Group

VM-Host Affinity Rule

**Add Cluster**

**Add**

**Create Availability Zones**

Availability Zones  
Clusters and resource pools to which you will deploy Ops Manager-compatible products

Name\*  Add

IaaS Configuration\*

Clusters

Cluster \*  Add Cluster

Resource Pool

Host Group

VM-Host Affinity Rule

---

**Create Availability Zones**

Availability Zones  
Clusters and resource pools to which you will deploy Ops Manager-compatible products

Name\*  Add

IaaS Configuration\*

Clusters

Cluster \*  Add Cluster

Resource Pool

Host Group

VM-Host Affinity Rule

**Save**

3. Click **Save**.

## Step 5: Create Networks

You must configure and create BOSH Director networking.



**Note:** If you are using [No-NAT with Virtual Switch \(VSS/VDS\) Topology](#), create the infrastructure network and optionally the TKGI network by following the instructions in [Create Networks Pane in Configuring BOSH Director on vSphere](#) in the Ops Manager documentation. While completing the steps in [Create Networks Pane](#), do not create the `services` network. With TKGI on NSX-T, NSX-T manages the dynamically created networks.

To configure BOSH Director networking:

1. Select **Create Networks** in BOSH Director.
-

BOSH Director for vSphere

Settings Status Credentials

vCenter Config Director Config Create Availability Zones

Create Networks

Warning: Pivotal recommends keeping the IP settings throughout the life of your installation. Ops Manager may prevent you from changing them in the future. Contact Pivotal support for help completing such a change.

Verification Settings

Enable ICMP checks

Networks

Add Network

One or many IP ranges upon which your products will be deployed

Save

Security Assign AZs and Networks Syslog Resource Config

PCF Ops Manager v2.3-build.170; ©2013-2018 Pivotal Software, Inc; All Rights Reserved. API Docs | End User License Agreement

2. Select **Enable ICMP checks** to enable ICMP on your networks. Ops Manager uses ICMP checks to confirm that components within your network are reachable.
3. Click **Add Network**.

NET-MGMT-PKS

Name\* NET-MGMT-PKS

Subnets

vSphere Network Name\* LS-MGMT-PKS

CIDR\* 10.0.0.0/24

Reserved IP Ranges 10.0.0.1-10.0.0.2

DNS\* 10.20.20.1

Gateway\* 10.0.0.1

Availability Zones\*  AZ-MGMT  AZ-COMP-1  AZ-COMP-2

PCF Ops Manager v2.3-build.170; ©2013-2018 Pivotal Software, Inc; All Rights Reserved. API Docs | End User License Agreement

4. Create the following network:

- ❖ **NET-MGMT-TKGI**: Network for Ops Manager, BOSH Director, and Tanzu Kubernetes Grid Integrated Edition components. This network maps to the NSX logical switch created for the Tanzu Kubernetes Grid Integrated Edition Management Network. See [Create Management Plane](#) in *Installing and Configuring NSX-T Data Center v3.0 for TKGI*.



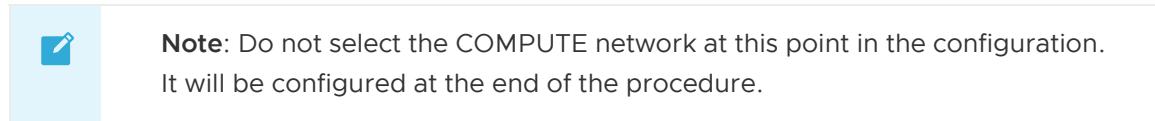
**Note:** NSX-T automatically creates the service network to be used by the control plane and worker nodes (VMs) for Kubernetes clusters managed by Tanzu Kubernetes Grid Integrated Edition. You should

not manually create this network.

Use the following values as a guide when you define the network in BOSH. Replace the IP addresses with ranges you defined for the [Create Management Plane](#) in *Installing and Configuring NSX-T Data Center v3.0 for TKGI*. Reserve any IP addresses from the subnet that are already in use, such as the IP for Ops Manager and subnet gateway.

| Infrastructure | Field                | Configuration     |
|----------------|----------------------|-------------------|
| Network        |                      |                   |
|                | Name                 | NET-MGMT-TKGI     |
|                | vSphere Network Name | LS-MGMT-TKGI      |
|                | CIDR                 | 10.0.0.0/24       |
|                | Reserved IP Ranges   | 10.0.0.1-10.0.0.2 |
|                | DNS                  | 10.20.20.1        |
|                | Gateway              | 10.0.0.1          |

- Select the **AZ-MGMT** Availability Zone to use with the **NET-MGMT-TKGI** network.



- Click **Save**.

The screenshot shows the PCF Ops Manager interface for configuring BOSH Director settings. The main navigation bar includes 'PCF Ops Manager', 'INSTALLATION DASHBOARD', 'STEMCELL LIBRARY', 'CHANGELOG', and a user account 'admin'. Below the header, a green banner indicates 'Settings updated'. The main content area is titled 'BOSH Director for vSphere' and has tabs for 'Settings', 'Status', and 'Credentials'. Under 'Settings', the 'Create Networks' section is active, showing fields for 'CIDR' (10.0.0.0/24), 'DNS' (10.20.20.1), and 'Gateway' (10.0.0.1). A note states: 'Warning: Pivotal recommends keeping the IP settings throughout the life of your installation. Ops Manager may prevent you from changing them in the future. Contact Pivotal support for help completing such a change.' Below this, the 'Verification Settings' section includes 'Enable ICMP checks' (checked). The 'Networks' section lists 'NET-MGMT-PKS' with an 'Add Network' button. Other configuration sections include 'vCenter Config', 'Director Config', 'Create Availability Zones', 'Assign AZs and Networks' (which is currently inactive), 'Security', 'Syslog', and 'Resource Config'. A large blue 'Save' button is located at the bottom of the configuration pane. The footer of the interface includes the copyright notice 'PCF Ops Manager v2.3-build.170; ©2013-2018 Pivotal Software, Inc; All Rights Reserved.' and links to 'API Docs' and 'End User License Agreement'.

## Step 6: Assign AZs and Networks

To configure the AZs and the Network for BOSH Director:

- Select **Assign AZs and Networks** in BOSH Director.

The screenshot shows the 'PCF Ops Manager' interface for 'BOSH Director for vSphere'. The top navigation bar includes 'INSTALLATION DASHBOARD', 'STEMCELL LIBRARY', and 'CHANGELOG'. A user 'admin' is logged in. The main section is titled 'Assign AZs and Networks'. It displays a list of configuration items: 'vCenter Config' (checked), 'Director Config' (checked), 'Create Availability Zones' (checked), 'Create Networks' (checked), 'Assign AZs and Networks' (highlighted in grey), 'Security' (checked), 'Syslog' (checked), and 'Resource Config' (checked). Below this is a note: 'The BOSH Director is a single instance. Choose the availability zone in which to place that instance. It is highly recommended that you backup this VM on a regular basis to preserve settings.' It shows 'Singleton Availability Zone' set to 'AZ-MGMT' and 'Network' set to 'NET-MGMT-PKS'. A 'Save' button is at the bottom.

2. Use the drop-down menu to select a **Singleton Availability Zone**. The Ops Manager Director installs in this Availability Zone. For Tanzu Kubernetes Grid Integrated Edition, this will be the **AZ-MGMT** availability zone.
3. Use the drop-down menu to select a **Network** for BOSH Director. BOSH Director runs on the Tanzu Kubernetes Grid Integrated Edition Management Plane network. Select the **NST-MGMT-TKGI** network.
4. Click **Save**.

The screenshot shows the 'PCF Ops Manager' interface for 'BOSH Director for vSphere'. The top navigation bar includes 'INSTALLATION DASHBOARD', 'STEMCELL LIBRARY', and 'CHANGELOG'. A user 'admin' is logged in. A green header bar indicates a successful action: 'Successfully assigned Network and Availability Zone'. The main section is titled 'Assign AZs and Networks'. It displays a list of configuration items: 'vCenter Config' (checked), 'Director Config' (checked), 'Create Availability Zones' (checked), 'Create Networks' (checked), 'Assign AZs and Networks' (highlighted in grey), 'Security' (checked), 'Syslog' (checked), and 'Resource Config' (checked). Below this is a note: 'The BOSH Director is a single instance. Choose the availability zone in which to place that instance. It is highly recommended that you backup this VM on a regular basis to preserve settings.' It shows 'Singleton Availability Zone' set to 'AZ-MGMT' and 'Network' set to 'NET-MGMT-PKS'. A 'Save' button is at the bottom. At the bottom of the page, a footer notes 'PCF Ops Manager v2.3-build.170; ©2013-2018 Pivotal Software, Inc; All Rights Reserved.' and links to 'API Docs' and 'End User License Agreement'.

## Step 7: Configure Security

To configure a BOSH Director certificate and password:

1. Select **Security** in BOSH Director.
2. In **Trusted Certificates**, enter a custom certificate authority (CA) certificate to insert into your

organization's certificate trust chain. This feature allows all BOSH-deployed components in your deployment to trust a custom root certificate.

If you are using self-signed CAs for the infrastructure components (NSX-T, vCenter), you need to add every CA of every component your deployment might connect to. In other words, the bundle must include all certificates for any component that connects to or from BOSH.

If you are using a private Docker registry, such as VMware Harbor, use this field to enter the certificate for the registry. See [Integrating Harbor Registry with Tanzu Kubernetes Grid Integrated Edition](#) for details.

3. Choose **Generate passwords** or **Use default BOSH password**. Use the **Generate passwords** option for increased security.
4. Click **Save**. To view your saved Director password, click the **Credentials** tab.

## Step 8: Configure BOSH DNS

To configure BOSH Director DNS:

1. Select **BOSH DNS Config** in BOSH Director.
2. (Optional) In **Excluded Recursors**, enter a list of prohibited recursor addresses.
3. (Optional) In **Recursor Timeout**, enter a time limit for contacting the connected recursors. This includes dialing, writing, and reading from the recursor. If any of these actions exceeds the time limit you set, the action fails.



**Note:** This time limit must include one of the Go parse duration time units. For example, entering `5s` sets the timeout limit to five seconds. For more information about supported time units, see [func ParseDuration](#) in the Go Programming Language documentation.

4. (Optional) In **Handlers**, enter a list of custom domain handlers in JSON format.
5. Click **Save**.

## Step 9: Configure Logging

To configure BOSH Director logging:

1. Select **Syslog** in BOSH Director.
2. (Optional) To send BOSH Director system logs to a remote server, select **Yes**.
3. In the **Address** field, enter the IP address or DNS name for the remote server.
4. In the **Port** field, enter the port number that the remote server listens on.
5. In the **Transport Protocol** dropdown menu, select **TCP** or **UDP**. This selection determines which transport protocol is used to send the logs to the remote server.
6. (Optional) Select the **Enable TLS** checkbox to send encrypted logs to remote server with TLS. After you select the checkbox, perform the following steps:

1. Enter either the name or SHA1 fingerprint of the remote peer in **Permitted Peer**.
2. Enter the SSL certificate for the remote server in **SSL Certificate**.



**Note:** For an optimal security configuration, enable TLS encryption when you are forwarding logs. Logs can contain sensitive information, such as cloud provider credentials.

7. (Optional) Enter an integer in **Queue Size**. This value specifies the number of log messages held in the buffer. The default value is 100,000.
8. (Optional) Select the checkbox to **Forward Debug Logs** to an external source. This option is deselected by default. If you select it, you might generate a large amount of log data.
9. (Optional) Enter configuration details for rsyslog in the **Custom rsyslog Configuration** field. This field requires the rainerscript syntax.
10. Click **Save Syslog Settings**.

## Step 10: Configure Resources

To configure BOSH Director resources:

1. Select **Resource Config** in BOSH Director.
2. Adjust any values as necessary for your deployment. Under the **Instances**, **Persistent Disk Type**, and **VM Type** fields, choose **Automatic** from the drop-down menu to allocate the recommended resources for the job. If the **Persistent Disk Type** field reads **None**, the job does not require persistent disk space.



**Note:** Ops Manager requires a Director VM with at least 8 GB memory.



**Note:** If you set a field to **Automatic** and the recommended resource allocation changes in a future version, Ops Manager automatically uses the updated recommended allocation.

3. Click **Save**.

## Step 11: (Optional) Add Custom VM Extensions

Use the Ops Manager API to add custom properties to your VMs such as associated security groups and load balancers.

For more information, see [Managing Custom VM Extensions](#).

## Step 12: Deploy BOSH

To deploy BOSH:

1. Go to the Ops Manager **Installation Dashboard**.

The screenshot shows the PCF Ops Manager interface. At the top, there's a navigation bar with a teal logo, the text "PCF Ops Manager", and links for "INSTALLATION DASHBOARD", "STEMCELL LIBRARY", and "CHANGELOG". On the far right, it says "admin" with a dropdown arrow. Below the navigation, there's a button "Import a Product". The main area is titled "Installation Dashboard" and features a large card for the "BOSH Director for vSphere" product. The card displays the "vmware" logo, the product name, its version "v2.3-build.170", and a prominent teal button at the bottom. To the left of the card, there's a link "Download PCF compatible products at Pivotal Network" and a link "Delete All Unused Products". At the bottom of the dashboard, it says "PCF Ops Manager v2.3-build.170; ©2013-2018 Pivotal Software, Inc; All Rights Reserved." and includes links for "API Docs" and "End User License Agreement".

**2. Click Review Pending Changes.**

The screenshot shows the "Review Pending Changes" screen. The top navigation bar is identical to the previous one. The main content area is titled "Review Pending Changes". It lists the "BOSH Director" product under "Select All Products", which is checked. The product card shows the "vmware" logo, the name "BOSH Director", and the version "Version 2.3-build.170". Below the product card, there's a section titled "Depends on" with the message "No Dependencies". On the right side of the screen, there's a large blue button labeled "APPLY CHANGES". At the bottom of the screen, it says "PCF Ops Manager 2.3-build.170; ©2013-2018 Pivotal Software, Inc; All Rights Reserved." and includes links for "API Docs" and "End User License Agreement".

**3. Click Apply Changes.**

PCF Ops Manager

INSTALLATION DASHBOARD STEMCELL LIBRARY CHANLOG admin ▾

## Applying Changes

0% Hide verbose output

**Installing BOSH**

- Uploading runtime config releases to the director
- Updating BOSH director with 2.0 cloud config
- Updating CPI configs
- Updating Internal UAA Configuration
- Putting Tile Credentials into CredHub
- Cleaning up BOSH director

```
===== 2018-10-15 17:14:50 UTC Running "/usr/local/bin/bosh --no-color --non-interactive --tty create-env /var/tempest/workspaces/default/deployments/bosh.yml"
Deployment manifest: '/var/tempest/workspaces/default/deployments/bosh.yml'
Deployment state: '/var/tempest/workspaces/default/deployments/bosh-state.json'

Started validating
Validating release 'bosh'... Finished (00:00:01)
Validating release 'bosh-vsphere-cpi'... Finished (00:00:00)
Validating release 'uaa'... Finished (00:00:05)
```

PCF Ops Manager 2.3-build.170; ©2013-2018 Pivotal Software, Inc. All Rights Reserved. API Docs | End User License Agreement

#### 4. Confirm changes applied successfully.

PCF Ops Manager

INSTALLATION DASHBOARD STEMCELL LIBRARY CHANLOG admin ▾

## Applying Changes

100%  Changes Applied

Your changes were successfully applied.  
We recommend that you export a backup of this installation from the actions menu.

CLOSE RETURN TO DASHBOARD Hide verbose output

**Installing BOSH**

- Uploading runtime config releases to the director
- Updating BOSH director with 2.0 cloud config
- Updating CPI configs
- Updating Internal UAA Configuration
- Putting Tile Credentials into CredHub
- Cleaning up BOSH director

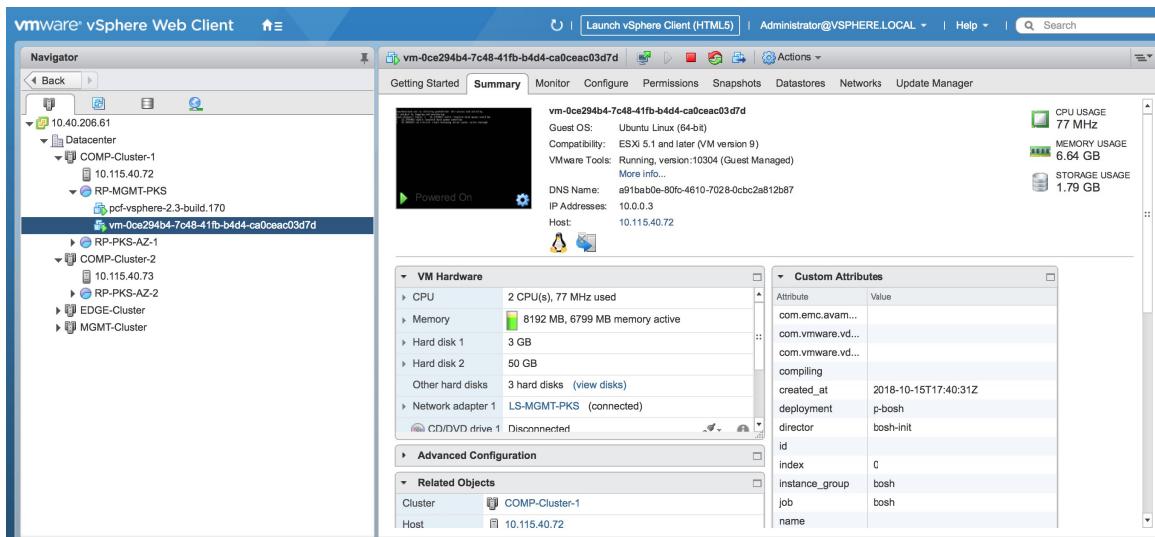
```
Succeeded
===== 2018-10-15 17:49:45 UTC Finished "/usr/local/bin/bosh --no-color --non-interactive --tty --environment=10.0.0.3 update-cpi-config /tmp/cpi_configs.yml20181015-810-1m4gczz"; Duration: 0s; Exit Status: 0
===== 2018-10-15 17:49:46 UTC Running "/usr/local/bin/bosh --no-color --non-interactive --tty --environment=10.0.0.3 clean-up"
Using environment '10.0.0.3' as client 'ops_manager'
Task 2
Task 2 | 17:49:46 | Deleting dns blobs: DNS blobs (00:00:00)

Task 2 Started Mon Oct 15 17:49:46 UTC 2018
Task 2 Finished Mon Oct 15 17:49:46 UTC 2018
Task 2 Duration 00:00:00
Task 2 done

Succeeded
===== 2018-10-15 17:49:46 UTC Finished "/usr/local/bin/bosh --no-color --non-interactive --tty --environment=10.0.0.3 clean-up"; Duration: 0s; Exit Status: 0
Exited with 0.
```

PCF Ops Manager 2.3-build.170; ©2013-2018 Pivotal Software, Inc. All Rights Reserved. API Docs | End User License Agreement

#### 5. Check BOSH VM. Log in to vCenter and check for the `p-bosh` VM deployment in the Tanzu Kubernetes Grid Integrated Edition Management resource pool.



## Step 13: Update Network Availability Zones

After successfully deploying BOSH, ensure that both the Management AZ and the Compute AZs appear in the Tanzu Kubernetes Grid Integrated Edition tile Plans.

To ensure that the Management AZ and the Compute AZs are included in the [NET-MGMT-TKGI](#) network you defined above:

1. Return to the BOSH tile and click **Create Networks**.

| Attribute        | Value                |
|------------------|----------------------|
| com.emc.avam...  |                      |
| com.vmware.vd... |                      |
| com.vmware.vd... |                      |
| compiling        |                      |
| created_at       | 2018-10-15T17:40:31Z |
| deployment       | p-bosh               |
| director         | bosh-init            |
| id               |                      |
| index            | 0                    |
| instance_group   | bosh                 |
| job              | bosh                 |
| name             |                      |

2. Edit the network ([NET-MGMT-TKGI](#)) and each COMPUTE AZ.

The screenshot shows the 'Create Network' configuration page. The network name is set to 'NET-MGMT-PKS'. Subnets are defined with 'vSphere Network Name' as 'LS-MGMT-PKS', 'CIDR' as '10.0.0.0/24', and 'Reserved IP Ranges' as '10.0.0.1-10.0.0.2'. DNS is set to '10.20.20.1', and the gateway is '10.0.0.1'. Availability zones selected are AZ-MGMT, AZ-COMP-1, and AZ-COMP-2. A note at the bottom states: 'Warning: Pivotal recommends keeping the IP settings throughout the life of your installation. Ops Manager may prevent you from changing them in the future. Contact Pivotal support for help completing such a change.'

3. Click Save.

The screenshot shows the 'Create Networks' section of the BOSH Director configuration. It includes a warning about keeping IP settings. Under 'Verification Settings', 'Enable ICMP checks' is checked. In the 'Networks' section, a network named 'NET-MGMT-PKS' is listed. A 'Save' button is visible at the bottom. A note at the bottom states: 'One or many IP ranges upon which your products will be deployed'.

4. Review pending changes, and click **Apply Changes** to redeploy BOSH.

## Next Step

Generate and Register the NSX Manager Superuser Principal Identity Certificate and Key for Tanzu Kubernetes Grid Integrated Edition.

## Generating and Registering the VMware NSX Manager Superuser Principal Identity Certificate and Key

This topic describes how to generate and register the NSX-T Manager superuser principal identity certificate and key. You do this in two situations:

- You are preparing to install Tanzu Kubernetes Grid Integrated Edition (TKGI) on vSphere with

NSX-T. For instructions, see [Generating the Certificate and Key for Installation](#).

- You need to rotate the NSX-T Manager certificate and key for an existing TKGI installation. For instructions, see [Rotate the Principal Identity Certificate and Key](#).

The NSX-T Manager superuser for TKGI has the Enterprise Administrator role and permissions. See [Role-Based Access Control](#) in the VMware documentation for more information.

## Installation Prerequisites

Review these prerequisites if you are installing TKGI on vSphere with NSX-T for the first time.

If you are rotating the NSX-T Manager certificate for an existing TKGI installation, see [Rotate the Principal Identity Certificate and Key](#), below.

- [Preparing to Install Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T Data Center](#)
- [Hardware Requirements for Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T](#)
- Depending on your NSX-T version:
  - ◊ [NSX-T v3.0: Installing and Configuring NSX-T Data Center v3.0 for Tanzu Kubernetes Grid Integrated Edition](#)
  - ◊ [NSX-T v2.5: See the v1.7 documentation:](#)
    - [Installing and Configuring NSX-T Data Center v2.5 for Enterprise PKS and the sequence of topics it includes](#)
    - [Creating the Tanzu Kubernetes Grid Integrated Edition Management Plane](#)
    - [Create Tanzu Kubernetes Grid Integrated Edition Compute Plane](#)
- [Deploying Ops Manager with NSX-T for Tanzu Kubernetes Grid Integrated Edition](#)
- Generate and Register the NSX-T Management SSL Certificate and Private Key in *Installing and Configuring NSX-T Data Center v3.0 for TKGI*
- [Configuring BOSH Director with NSX-T for Tanzu Kubernetes Grid Integrated Edition](#)

## How Ops Manager Accesses NSX-T Manager

To create, delete, and modify NSX-T networking resources, Ops Manager tiles and APIs use a VMware NSX Manager account with the Enterprise Administrator role and permissions.

Users configure Ops Manager to authenticate to NSX Manager for different purposes in different tiles:

- **Tanzu Kubernetes Grid Integrated Edition tile**

The Tanzu Kubernetes Grid Integrated Edition tile uses NSX Manager to create load balancers, providing a Kubernetes service described in the [Create an External Load Balancer](#) section of the Kubernetes documentation.

To configure the **Tanzu Kubernetes Grid Integrated Edition** tile's authentication to NSX Manager, see [About the NSX Manager Superuser Principal Identity](#), below.

- **BOSH Director for vSphere tile**

The **BOSH Director for vSphere** tile uses NSX Manager to configure networking and security for external-facing Ops Manager component VMs, such as VMware Tanzu Application Service for VMs routers.

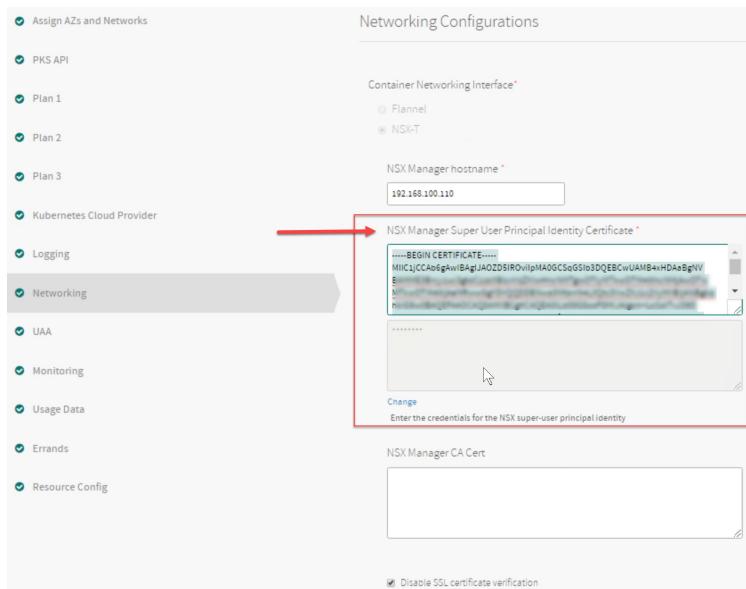
To configure the **BOSH Director for vSphere** tile's authentication to NSX Manager, see [Configure vCenter for Tanzu Kubernetes Grid Integrated Edition](#) in *Configuring BOSH Director with NSX-T for Tanzu Kubernetes Grid Integrated Edition*.

## About the NSX-T Manager Super User Principal Identity

The TKGI API accesses the NSX-T Manager through an Enterprise Administrator account. This superuser account lets TKGI use NSX-T to create, delete, and modify networking resources for Kubernetes cluster nodes.

When you configure Tanzu Kubernetes Grid Integrated Edition with NSX-T as the container networking interface, you must provide the certificate and private key for the NSX-T Manager Enterprise Administrator account in the **Networking** pane of the Tanzu Kubernetes Grid Integrated Edition tile.

See the **NSX Manager Super User Principal Identity Certificate** field in the following screenshot:



[View a larger version of this image.](#)

For more information, see the **Networking** section of *Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T*.

## Generating the Certificate and Key for Installation

There are two options for generating the principal identity certificate and private key:

- [Option A](#): Run a script on a Linux host with OpenSSL installed that generates the certificate and private key. For more information, see [Option A: Generate and Register the Certificate and Key Using Scripts](#) below.

- **Option B:** Use the automatic **Generate RSA Certificate** option in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Option B: Generate and Register the Certificate and Key Using the Tanzu Kubernetes Grid Integrated Edition Tile](#) below.

After you have generated the principal identity certificate and key, you must register both with the NSX-T Manager using an HTTPS POST operation on the NSX-T API. There is no user interface for this operation.

## Option A: Generate and Register the Certificate and Key Using a Script

This option uses a Bash shell script to generate and register the NSX-T Manager superuser principal identity certificate and key. When you configure TKGI for deployment, copy and paste the contents of `pks-nsx-t-superuser.crt` and `pks-nsx-t-superuser.key` to the **NSX Manager Super User Principal Identity Certificate** field in the **Networking** pane of the Tanzu Kubernetes Grid Integrated Edition tile.



**Note:** The Linux VM must have OpenSSL installed and have network access to the NSX-T Manager. For example, you can use the TKGI client VM where you install the TKGI CLI.

### Step 1: Generate and Register the Certificate and Key

You must generate a certificate and private key, and create the Super User Principal Identity with the certificate in the NSX-T Manager.

To create the Super User Principal Identity, create and run the `create_certificate_pi.sh` script:

1. Log in to a Linux VM in your Tanzu Kubernetes Grid Integrated Edition environment.
2. Create an empty file using `vi create_certificate_pi.sh` or `nano create_certificate_pi.sh`.
3. Modify the file you created to have the following script contents:

```
#!/bin/bash
#create_certificate_pi.sh

NSX_MANAGER="NSX-MANAGER-IP"
NSX_USER="NSX-MANAGER-USERNAME"

PI_NAME="pks-nsx-t-superuser"
NSX_SUPERUSER_CERT_FILE="pks-nsx-t-superuser.crt"
NSX_SUPERUSER_KEY_FILE="pks-nsx-t-superuser.key"
NODE_ID=$(cat /proc/sys/kernel/random/uuid)

stty -echo
printf "Password: "
read NSX_PASSWORD
stty echo

Create a certificate and key for the PI
openssl req \
-newkey rsa:2048 \
```

```

-x509 \
-nodes \
-keyout "$NSX_SUPERUSER_KEY_FILE" \
-new \
-out "$NSX_SUPERUSER_CERT_FILE" \
-subj /CN="$PI_NAME" \
-extensions client_server_ssl \
-config <(
 cat /etc/ssl/openssl.cnf \
 <(printf '[client_server_ssl]\nextendedKeyUsage = clientAuth\n')
) \
-sha256 \
-days 730

Define the payload of the request
pi_request=$(cat <<END
{
 "display_name": "$PI_NAME",
 "name": "$PI_NAME",
 "role": "enterprise_admin",
 "certificate_pem": "$(awk 'NF {sub(/\r/, ""); printf "%s\\n",$0;}' ./"
$NSX_SUPERUSER_CERT_FILE")",
 "node_id": "$NODE_ID"
}
END
)

Create an API request to register a name-certificate combination
curl -k -X POST \
 "https://{$NSX_MANAGER}/api/v1/trust-management/principal-identities/with-c
ertificate" \
 -u "$NSX_USER:$NSX_PASSWORD" \
 -H 'content-type: application/json' \
 -d "$pi_request"

List all PIs
curl -k -X GET \
 "https://{$NSX_MANAGER}/api/v1/trust-management/principal-identities" \
 --cert $(pwd)"/$NSX_SUPERUSER_CERT_FILE" \
 --key $(pwd)"/$NSX_SUPERUSER_KEY_FILE"

```

Where:

- ◊ **NSX-MANAGER-IP** is the IP address of the NSX Management Cluster VIP or NSX Management Load Balancer IP.
  - ◊ **NSX-MANAGER-USERNAME** is the Username for NSX Manager.
4. Save the `create_certificate_pi.sh` file.
  5. Run the script using `bash create_certificate_pi.sh`.
  6. When prompted, enter the `NSX_MANAGER_PASSWORD` for the NSX-T user you specified in the script.
  7. Verify results:
    - ◊ Confirm the certificate, `pks-nsx-t-superuser.crt`, and private key, `pks-nsx-t-
superuser.key`, are generated in the directory where you ran the script.

- ◊ Confirm the principal identity associated with the certificate was created in NSX-T.
- ◊ Confirm the principal identity `pks-nsx-t-superuser` is registered with the role `Enterprise Admin` on the NSX-T Manager **System > Users > Role Assignments** screen.

| User/User Group Name             | Roles            |
|----------------------------------|------------------|
| <code>pks-nsx-t-superuser</code> | Enterprise Admin |
| <code>nsx_policy</code>          | Enterprise Admin |
| <code>audit</code>               | Auditor          |
| <code>admin</code>               | Enterprise Admin |

[View a larger version of this image.](#)

## Option B: Generate and Register the Certificate and Key Using the Tanzu Kubernetes Grid Integrated Edition Tile

### Step 1: Generate the Certificate and Key

To generate the certificate and key automatically in the **Networking** pane in the Tanzu Kubernetes Grid Integrated Edition tile, follow the steps below:

1. Navigate to the **Networking** pane in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Networking](#) in *Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T Integration*.
2. Click **Generate RSA Certificate** and provide a wildcard domain. For example, `*.nsx.tkgi.vmware.local`.

### Step 2: Copy the Certificate and Key to the Linux VM

To copy the certificate and key you generated to a Linux VM, follow the steps below:



**Note:** The Linux VM must have OpenSSL installed and have network access to the NSX-T Manager. For example, you can use the TKGI client VM where you install the TKGI CLI.

1. On the Linux VM you want to use to register the certificate, create a file named `pks-nsx-t-superuser.crt`. Copy the generated certificate into the file.
2. On the Linux VM you want to use to register the key, create a file named `pks-nsx-t-superuser.key`. Copy the generated private key into the file.

3. Save both files.

### Step 3: Export Environment Variables

On the Linux VM where you created the certificate and key files, export the environment variables below. Change the `NSX_MANAGER_IP`, `NSX_MANAGER_USERNAME`, and `NSX_MANAGER_PASSWORD` values to match your environment. Use the NSX-T Management Cluster VIP or load balancer for the `NSX_MANAGER_IP`.

```
export NSX_MANAGER="NSX_MANAGER_IP"
export NSX_USER="NSX_MANAGER_USERNAME"
export NSX_PASSWORD='NSX_MANAGER_PASSWORD'
export PI_NAME="pks-nsx-t-superuser"
export NSX_SUPERUSER_CERT_FILE="pks-nsx-t-superuser.crt"
export NSX_SUPERUSER_KEY_FILE="pks-nsx-t-superuser.key"
export NODE_ID=$(cat /proc/sys/kernel/random/uuid)
```

### Step 4: Register the Certificate

1. On the same Linux VM, run the following commands to register the principal identity with NSX-T Manager:

```
$ pi_request=$(cat <<END
{
 "display_name": "$PI_NAME",
 "name": "$PI_NAME",
 "role": "enterprise_admin",
 "certificate_pem": "$(awk 'NF {sub(/\\r/, ""); printf "%s\\n", $0;}' ./
$NSX_SUPERUSER_CERT_FILE")",
 "node_id": "$NODE_ID"
}
END
)
```

```
$ curl -k -X POST \
 "https://${NSX_MANAGER}/api/v1/trust-management/principal-identities/with-c
ertificate" \
-u "$NSX_USER:$NSX_PASSWORD" \
-H 'content-type: application/json' \
-d "$pi_request"
```

### Step 6: Verify the Certificate and Key

To verify that the certificate and key can be used with NSX-T, run the following command:

```
$ curl -k -X GET \
"https://${NSX_MANAGER}/api/v1/trust-management/principal-identities" \
--cert $(pwd) /"$NSX_SUPERUSER_CERT_FILE" \
--key $(pwd) /"$NSX_SUPERUSER_KEY_FILE"
```

## Rotate the Principal Identity Certificate and Key

To rotate the NSX-T Principal Identity super user certificate, see [How to renew the nsx-t-superuser-certificate used by Principal Identity user] (<https://kb.vmware.com/s/article/80355>) in the VMware Knowledge Base.

## Next Installation Step

If you have completed this procedure as part of installing TKGI for the first time, proceed to [Installing TKGI on vSphere with NSX-T](#).

## Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with VMware NSX

This topic describes how to install and configure VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) on vSphere with NSX-T integration.

## Prerequisites

Before you begin this procedure, ensure that you have successfully completed all preceding steps for installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T, including:

- [Preparing to Install Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T Data Center](#)
- [Installing and Configuring NSX-T Data Center v3.0 for Tanzu Kubernetes Grid Integrated Edition](#)
- [Configuring NSX-T Data Center v3.1 Transport Zones and Edge Node Switches for Tanzu Kubernetes Grid Integrated Edition](#)
- [Deploying Ops Manager with NSX-T for Tanzu Kubernetes Grid Integrated Edition](#)
- [Generate and Register the NSX-T Management SSL Certificate and Private Key in \*Installing and Configuring NSX-T Data Center v3.0 for TKGI\*](#)
- [Configuring BOSH Director with NSX-T for Tanzu Kubernetes Grid Integrated Edition](#)
- [Generating and Registering the NSX Manager Superuser Principal Identity Certificate and Key for Tanzu Kubernetes Grid Integrated Edition](#)

## Step 1: Install Tanzu Kubernetes Grid Integrated Edition

To install Tanzu Kubernetes Grid Integrated Edition, do the following:

1. Download the product file from [VMware Tanzu Network](#).
2. Navigate to <https://YOUR-OPS-MANAGER-FQDN/> in a browser to log in to the Ops Manager Installation Dashboard.
3. Click **Import a Product** to upload the product file.
4. Under **Tanzu Kubernetes Grid Integrated Edition** in the left column, click the plus sign to add this product to your staging area.

## Step 2: Configure Tanzu Kubernetes Grid Integrated Edition

Click the orange **Tanzu Kubernetes Grid Integrated Edition** tile to start the configuration process.



**Note:** Configuration of NSX-T or Flannel **cannot** be changed after initial installation and configuration of Tanzu Kubernetes Grid Integrated Edition.

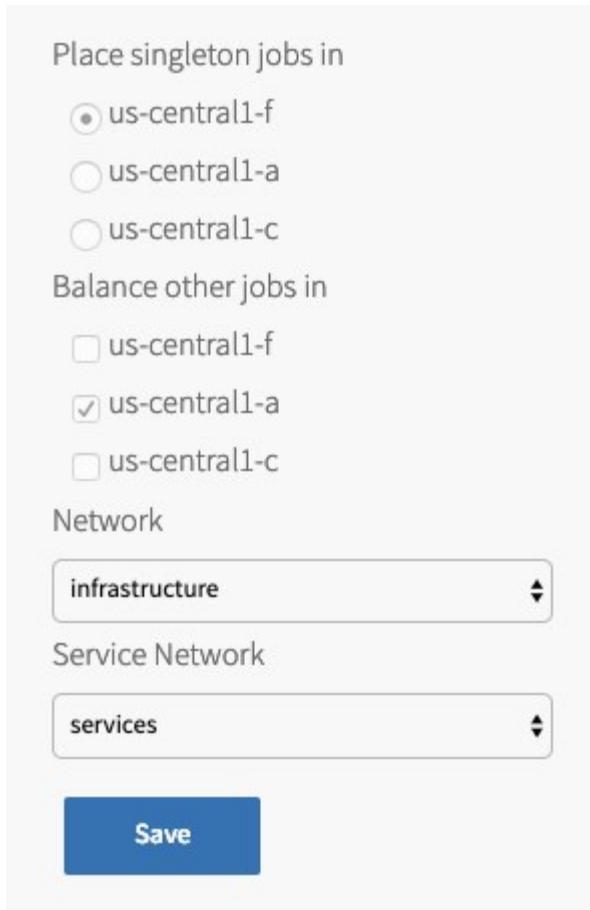


**WARNING:** When you configure the Tanzu Kubernetes Grid Integrated Edition tile, do not use spaces in any field entries. This includes spaces between characters as well as leading and trailing spaces. If you use a space in any field entry, the deployment of Tanzu Kubernetes Grid Integrated Edition fails.

## Assign AZs and Networks

To configure the availability zones (AZs) and networks used by the Tanzu Kubernetes Grid Integrated Edition control plane:

1. Click **Assign AZs and Networks**.
2. Under **Place singleton jobs in**, select the availability zone (AZ) where you want to deploy the TKGI API and TKGI Database VMs.



- Under **Balance other jobs in**, select the AZ for balancing other Tanzu Kubernetes Grid Integrated Edition control plane jobs.



**Note:** You must specify the **Balance other jobs in** AZ, but the selection has no effect in the current version of Tanzu Kubernetes Grid Integrated Edition.

- Under **Network**, select the TKGI Management Network linked to the `ls-tkgi-mgmt` NSX-T logical switch you created in the [Create Networks Page](#) step of *Configuring BOSH Director with NSX-T for Tanzu Kubernetes Grid Integrated Edition*. This provides network placement for Tanzu Kubernetes Grid Integrated Edition component VMs, such as the TKGI API and TKGI Database VMs.
- Under **Service Network**, your selection depends on whether you are installing a new Tanzu Kubernetes Grid Integrated Edition deployment or upgrading from a previous version of Tanzu Kubernetes Grid Integrated Edition.
  - If you are deploying Tanzu Kubernetes Grid Integrated Edition with NSX-T for the first time, select the TKGI Management Network that you specified in the **Network** field. You do not need to create or define a service network because Tanzu Kubernetes Grid Integrated Edition creates the service network for you during the installation process.
  - If you are upgrading from a previous version of Tanzu Kubernetes Grid Integrated Edition, then select the **Service Network** linked to the `ls-tkgi-service` NSX-T logical switch that Tanzu Kubernetes Grid Integrated Edition created for you during installation. The service network provides network placement for existing on-demand

Kubernetes cluster service instances that were created by the Tanzu Kubernetes Grid Integrated Edition broker.

6. Click **Save**.

## TKGI API

Perform the following steps:

1. Click **TKGI API**.
2. Under **Certificate to secure the TKGI API**, provide a certificate and private key pair.

The screenshot shows the 'TKGI API Service' configuration page. It includes fields for 'Certificate PEM' and 'Private Key PEM' (both empty), a 'Generate RSA Certificate' button, an 'API Hostname (FQDN)' field containing 'tkgi.api.example.com', a 'Worker VM Max in Flight' field containing '4', and a large blue 'Save' button.

TKGI API Service

Certificate to secure the TKGI API \*

Certificate PEM

Private Key PEM

Generate RSA Certificate

API Hostname (FQDN) \*

tkgi.api.example.com

Worker VM Max in Flight \*

4

Save

The certificate that you supply must cover the specific subdomain that routes to the TKGI API VM with TLS termination on the ingress. If you use UAA as your OIDC provider, this certificate must be a proper certificate chain and have a SAN field.

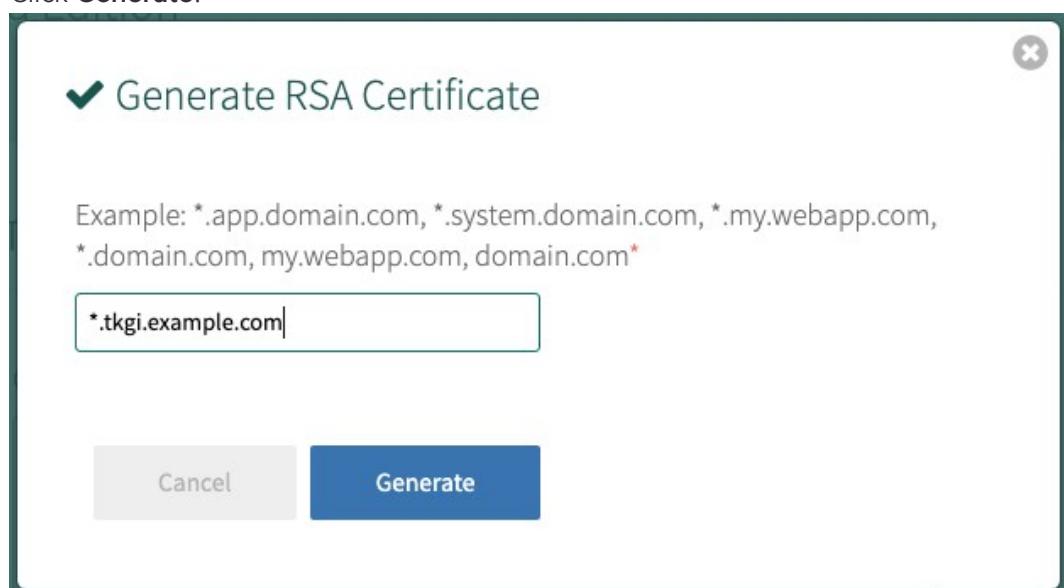


**Warning:** TLS certificates generated for wildcard DNS records only work for a single domain level. For example, a certificate generated for `*.tkgi.EXAMPLE.com` does not permit communication to `*.api.tkgi.EXAMPLE.com`. If the certificate does not contain the correct FQDN for the TKGI API, calls to the API will fail.

You can enter your own certificate and private key pair, or have Ops Manager generate one for you.

To generate a certificate using Ops Manager:

1. Click **Generate RSA Certificate** for a new install or **Change** to update a previously-generated certificate.
2. Enter the domain for your API hostname. This must match the domain you configure under **TKGI API > API Hostname (FQDN)** below, in the same pane. It can be a standard FQDN or a wildcard domain.
3. Click **Generate**.



3. Under **API Hostname (FQDN)**, enter the FQDN that you registered to point to the TKGI API load balancer, such as `api.tkgi.example.com`. To retrieve the public IP address or FQDN of the TKGI API load balancer, log in to your IaaS console.



**Note:** The FQDN for the TKGI API must not contain uppercase letters or trailing whitespace.

4. Under **Worker VM Max in Flight**, enter the maximum number of non-canary worker instances to create or update in parallel within an availability zone.

This field sets the `max_in_flight` variable value. The `max_in_flight` setting limits the number of component instances the TKGI CLI creates or starts simultaneously when running `tkgi create-cluster` or `tkgi update-cluster`. By default, `max_in_flight` is set to `4`, limiting the TKGI CLI to creating or starting a maximum of four component instances in parallel.

5. Click **Save**.

## Plans

A plan defines a set of resource types used for deploying a cluster.

### Activate a Plan

You must first activate and configure **Plan 1**,

and afterwards you can activate up to twelve additional, optional, plans.

To activate and configure a plan, perform the following steps:

1. Click the plan that you want to activate.



**Note:** Plans 11, 12, and 13 support Windows worker-based Kubernetes clusters on vSphere with NSX-T, and are a beta feature on vSphere with Flannel. To configure a Windows worker plan see [Plans](#) in *Configuring Windows Worker-Based Kubernetes Clusters* for more information.

2. Select **Active** to activate the plan and make it available to developers deploying clusters.

## Configuration for Plan 1

Select 'Active' to allow users of the PKS CLI to create a cluster using this template plan.

### Plan\*

Active

### Name \*

small

### Description \*

Example: This plan will configure a lightweight kubernetes cluster. Not recommended for production workloads.

### Master/ETCD Node Instances ( min: 1, max: 5 ) \*

1

### Master/ETCD VM Type\*

Automatic: medium.disk (cpu: 2, ram: 4 GB, disk: 32 GB)



### Master Persistent Disk Type\*

Automatic: 10 GB



### Master/ETCD Availability Zones \*

us-central1-f

us-central1-a

us-central1-c

3. Under **Name**, provide a unique name for the plan.
4. Under **Description**, edit the description as needed. The plan description appears in the Services Marketplace, which developers can access by using the TKGI CLI.
5. Under **Master/ETCD Node Instances**, select the default number of Kubernetes control

plane/etcdb nodes to provision for each cluster. You can enter 1, 3, or 5.



**Note:** If you deploy a cluster with multiple control plane/etcdb node VMs, confirm that you have sufficient hardware to handle the increased load on disk write and network traffic. For more information, see [Hardware recommendations](#) in the etcd documentation.

In addition to meeting the hardware requirements for a multi-control plane node cluster, we recommend configuring monitoring for etcd to monitor disk latency, network latency, and other indicators for the health of the cluster. For more information, see [Configuring Telegraf in TKGI](#).



**WARNING:** To change the number of control plane/etcdb nodes for a plan, you must ensure that no existing clusters use the plan. Tanzu Kubernetes Grid Integrated Edition does not support changing the number of control plane/etcdb nodes for plans with existing clusters.

6. Under **Master/ETCD VM Type**, select the type of VM to use for Kubernetes control plane/etcdb nodes. For more information, including control plane node VM customization options, see the [Control Plane Node VM Size](#) section of *VM Sizing for Tanzu Kubernetes Grid Integrated Edition Clusters*.
7. Under **Master Persistent Disk Type**, select the size of the persistent disk for the Kubernetes control plane node VM.
8. Under **Master/ETCD Availability Zones**, select one or more AZs for the Kubernetes clusters deployed by Tanzu Kubernetes Grid Integrated Edition. If you select more than one AZ, Tanzu Kubernetes Grid Integrated Edition deploys the control plane VM in the first AZ and the worker VMs across the remaining AZs. If you are using multiple control plane nodes, Tanzu Kubernetes Grid Integrated Edition deploys the control plane and worker VMs across the AZs in round-robin fashion.



**Note:** Tanzu Kubernetes Grid Integrated Edition does not support changing the AZs of existing control plane nodes.

9. Under **Maximum number of workers on a cluster**, set the maximum number of Kubernetes worker node VMs that Tanzu Kubernetes Grid Integrated Edition can deploy for each cluster. Enter any whole number in this field.

Maximum number of workers on a cluster (min: 1) \*

Worker Node Instances (min: 1) \*

Worker VM Type\*

Automatic: medium.disk (cpu: 2, ram: 4 GB, disk: 32 GB)

Worker Persistent Disk Type\*

Automatic: 50 GB

Worker Availability Zones \*

- us-central1-f
- us-central1-a
- us-central1-c

10. Under **Worker Node Instances**, specify the default number of Kubernetes worker nodes the TKGI CLI provisions for each cluster. The **Worker Node Instances** setting must be less than, or equal to, the **Maximum number of workers on a cluster** setting.

For high availability, create clusters with a minimum of three worker nodes, or two per AZ if you intend to use PersistentVolumes (PVs). For example, if you deploy across three AZs, you should have six worker nodes. For more information about PVs, see [PersistentVolumes in Maintaining Workload Uptime](#). Provisioning a minimum of three worker nodes, or two nodes per AZ is also recommended for stateless workloads.

For more information about creating clusters, see [Creating Clusters](#).



**Note:** Changing a plan's **Worker Node Instances** setting does not alter the number of worker nodes on existing clusters. For information about scaling an existing cluster, see [Scale Horizontally by Changing the Number of Worker Nodes Using the TKGI CLI in Scaling Existing Clusters](#).

11. Under **Worker VM Type**, select the type of VM to use for Kubernetes worker node VMs. For more information, including worker node VM customization options, see [Worker Node VM Number and Size in VM Sizing for Tanzu Kubernetes Grid Integrated Edition Clusters](#).



**Note:** Tanzu Kubernetes Grid Integrated Edition requires a **Worker VM Type** with an ephemeral disk size of 32 GB or more.

12. Under **Worker Persistent Disk Type**, select the size of the persistent disk for the Kubernetes worker node VMs.
13. Under **Worker Availability Zones**, select one or more AZs for the Kubernetes worker nodes. Tanzu Kubernetes Grid Integrated Edition deploys worker nodes equally across the AZs you select.
14. Under **Kubelet customization - system-reserved**, enter resource values that Kubelet can use to reserve resources for system daemons. For example, `memory=250Mi`, `cpu=150m`. For more information about system-reserved values, see the [Kubernetes documentation](#).

#### Kubelet customization - system-reserved

#### Kubelet customization - eviction-hard

#### Errand VM Type\*

Automatic: t3.micro (cpu: 1, ram: 1 GB, disk: 8 GB) 

15. Under **Kubelet customization - eviction-hard**, enter threshold limits that Kubelet can use to evict pods when they exceed the limit. Enter limits in the format `EVICTION-SIGNAL=QUANTITY`. For example, `memory.available=100Mi`, `nodefs.available=10%`, `nodefs.inodesFree=5%`. For more information about eviction thresholds, see the [Kubernetes documentation](#).



**WARNING:** Use the Kubelet customization fields with caution. If you enter values that are invalid or that exceed the limits the system supports, Kubelet might fail to start. If Kubelet fails to start, you cannot create clusters.

16. Under **Errand VM Type**, select the size of the VM that contains the errand. The smallest instance possible is sufficient, as the only errand running on this VM is the one that applies the **Default Cluster App** YAML configuration.
17. (Optional) Under **(Optional) Add-ons - Use with caution**, enter additional YAML configuration to add custom workloads to each cluster in this plan. You can specify multiple files using `---` as a separator. For more information, see [Adding Custom Linux Workloads](#).

## (Optional) Add-ons - Use with caution

Allow Privileged

## Admission Plugins

- PodSecurityPolicy
- SecurityContextDeny

18. (Optional) Select the **Allow Privileged** option to allow users to either create Pods with privileged containers or create clusters with resizable persistent volumes using a manually installed vSphere CSI driver. For more information about privileged mode, see [Pods](#) in the Kubernetes documentation.

**Allow Privileged** is not required if clusters use the automatically installed vSphere CSI driver for vSphere CNS. For information about using the automatically installed vSphere CSI driver, see [Storage](#) below and [Deploying Cloud Native Storage \(CNS\) on vSphere](#).



**Note:** Enabling the [Allow Privileged](#) option means that all containers in the cluster will run in privileged mode. [Pod Security Policy](#) provides a [privileged](#) parameter that can be used to activate or deactivate Pods running in privileged mode. As a best practice, if you activate [Allow Privileged](#), define PSP to limit which Pods run in privileged mode. If you are implementing PSP for privileged pods, you must activate [Allow Privileged](#) mode.

19. (Optional) Activate or deactivate one or more admission controller plugins: [PodSecurityPolicy](#) and [SecurityContextDeny](#). For more information see [Using Admission Control Plugins for Tanzu Kubernetes Grid Integrated Edition Clusters](#).



**Note:** To use PodSecurityPolicy features, you must use Ops Manager v2.10.17 or later.

20. (Optional) Under **Node Drain Timeout(mins)**, enter the timeout in minutes for the node to drain pods. If you set this value to [0](#), the node drain does not terminate.

Node Drain Timeout(mins) ( min: 0, max: 1440 )

Pod Shutdown Grace Period (seconds) ( min: -1, max: 86400 )

Force node to drain even if it has running pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or StatefulSet.

Force node to drain even if it has running DaemonSet-managed pods.

Force node to drain even if it has running running pods using emptyDir.

Force node to drain even if pods are still running after timeout.

21. (Optional) Under **Pod Shutdown Grace Period (seconds)**, enter a timeout in seconds for the node to wait before it forces the pod to terminate. If you set this value to `-1`, the default timeout is set to the one specified by the pod.
22. (Optional) To configure when the node drains, activate the following:
  - Force node to drain even if it has running pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or StatefulSet.
  - Force node to drain even if it has running DaemonSet-managed pods.
  - Force node to drain even if it has running running pods using emptyDir.
  - Force node to drain even if pods are still running after timeout.



**Warning:** If you select **Force node to drain even if pods are still running after timeout**, the node halts all running workloads on pods. Before enabling this configuration, set **Node Drain Timeout** to a value greater than `0`.

For more information about configuring default node drain behavior, see [Worker Node Hangs Indefinitely](#) in [Troubleshooting](#).

23. Click **Save**.

## Deactivate a Plan

To deactivate a plan, perform the following steps:

1. Click the plan that you want to deactivate.

2. Select **Inactive**.
3. Click **Save**.

## Kubernetes Cloud Provider

In the procedure below, you use credentials for vCenter master VMs. You must have provisioned the service account with the correct permissions. For more information, see [Create the Master Node Service Account](#) in *Preparing vSphere Before Deploying Tanzu Kubernetes Grid Integrated Edition*.

To configure your Kubernetes cloud provider settings, follow the procedure below:

1. Click **Kubernetes Cloud Provider**.
2. Under **Choose your IaaS**, select **vSphere**.

Choose your IaaS\*

GCP  
 vSphere

vCenter Master Credentials \*

user@example.com

.....

vCenter Host \*

vcenter-example.com

Datacenter Name \*

example-dc

Datastore Name \*

example-ds

Stored VM Folder \*

pkd\_vms

3. Ensure the values in the following procedure match those in the **vCenter Config** section of the Ops Manager tile:
  1. Enter your **vCenter Master Credentials**. Enter the vCenter Server username using the format `user@domainname`, for example: “`user@example.com`”. For more information about the master node service account, see [Preparing vSphere Before](#)

## Deploying Tanzu Kubernetes Grid Integrated Edition.



**Warning:** The vSphere Container Storage Plug-in will not function if you do not specify the domain name for active directory users.

2. Enter your **vCenter Host**. For example, `vcenter-example.com`.



**Note:** The FQDN for the vCenter Server cannot contain uppercase letters.

3. Enter your **Datacenter Name**. For example, `example-dc`.
4. Enter your **Datastore Name**. For example, `example-ds`. Populate **Datastore Name** with the Persistent Datastore name configured in your **BOSH Director** tile under **vCenter Config > Persistent Datastore Names**.

The **Datastore Name** field should contain a single Persistent datastore.



**Note:** The vSphere datastore type must be Datastore. Tanzu Kubernetes Grid Integrated Edition does not support the use of vSphere Datastore Clusters with or without Storage DRS. For more information, see [Datastores and Datastore Clusters](#) in the vSphere documentation.



**Note:** The **Datastore Name** is the default datastore used if the Kubernetes cluster `StorageClass` does not define a `StoragePolicy`. Do not enter a datastore that is a list of BOSH Job/VMDK datastores. For more information, see [PersistentVolume Storage Options on vSphere](#).



**Note:** For multi-AZ and multi-cluster environments, your **Datastore Name** should be a shared Persistent datastore available to each vSphere cluster. Do not enter a datastore that is local to a single cluster. For more information, see [PersistentVolume Storage Options on vSphere](#).

5. Enter the **Stored VM Folder** so that the persistent stores know where to find the VMs. To retrieve the name of the folder, navigate to your BOSH Director tile, click **vCenter Config**, and locate the value for **VM Folder**. The default folder name is `pks_vms`.
4. Click **Save**.

## Networking

To configure networking, do the following:

1. Click **Networking**.

2. Under **Container Networking Interface**, select **NSX-T**.

**Networking Configurations**

**Container Networking Interface\***

Flannel  
 NSX-T

**NSX Manager hostname \***

**NSX Manager Super User Principal Identity Certificate \***  

```
-----BEGIN CERTIFICATE-----
MIIC1jCCAb6gAwIBAgIJAM7XLuOmmJKeMA0GCSqGSIb3DQEBCwUAMB4xHDAaBgNV
BAMME3Brct1uc3gttC1zdXBlcnVzZXlwHncNMTgxMDlyMTcwOTE3WhcNMjAxMDIx
MTcwOTE3WjAeMRwwGgYDVQQDDBNwa3MtbnN4LXQtc3VwZXJ1c2VyMIIBljANBgkq
nkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEArCjTVoBCfZBsAtRI/jlkhDVAH61j97cW

```

Change

**NSX Manager CA Cert**

```
-----BEGIN CERTIFICATE-----
MIIDTzCCAjegAwIBAgIJAikUwk/zSSSLMA0GCSqGSIb3DQEBCwUAMFUxCzAJBgNV
BAYTAIVTMRMwEQYDVQQIDAپDYWxpZm9ybmlhMQswCQYDVQQHDAJDQTEMMAoGA1
UE
CgwDTINYMRYwFAYDVQQDDA0xMC4xOTYuMTg4LjIxMB4XDTE4MTAyMDAwMTAxNFoX

```

Disable SSL certificate verification

NAT mode

1. For **NSX Manager hostname**, enter the hostname or IP address of your NSX-T Manager.
2. For **NSX Manager Super User Principal Identity Certificate**, copy and paste the contents and private key of the Principal Identity certificate you created in [Generating and Registering the NSX Manager Superuser Principal Identity Certificate and Key](#).
3. For **NSX Manager CA Cert**, copy and paste the contents of the NSX-T Manager CA certificate you created in [Generate and Register the NSX-T Management SSL Certificate and Private Key](#). Use this certificate and key to connect to the NSX-T Manager.
4. The **Disable SSL certificate verification** checkbox is **not** selected by default. In

order to deactivate TLS verification, select the checkbox. If you did not enter a CA certificate, or if your CA certificate is self-signed, you can deactivate TLS verification.



**Note:** The **NSX Manager CA Cert** field and the **Disable SSL certificate verification** option are intended to be mutually exclusive. If you deactivate SSL certificate verification, leave the CA certificate field blank. If you enter a certificate in the **NSX Manager CA Cert** field, do not deactivate SSL certificate verification. If you populate the certificate field and deactivate certificate validation, insecure mode takes precedence.

5. If you are using a NAT deployment topology, leave the **NAT mode** checkbox selected. If you are using a No-NAT topology, clear this checkbox. For more information, see [NSX-T Deployment Topologies for Tanzu Kubernetes Grid Integrated Edition](#).
6. If you are using the NSX-T Policy API, select **Policy API mode**.
7. Configure the NSX-T networking objects, including the **Pods IP Block ID**, **Nodes IP Bock ID**, **T0 Router ID**, **Floating IP Pool ID**, **Nodes DNS**, **vSphere Cluster Names**, and **Kubernetes Service Network CIDR Range**. Each of the these fields are described in more detail beneath the example screenshots. If you are using the NSX-T Policy API, you must have created the **Pods IP Block ID**, **Nodes IP Bock ID**, **T0 Router ID**, and **Floating IP Pool ID** objects using the NSX-T Policy API. See [Create NSX-T Objects for Kubernetes Clusters Using the Policy Interface](#).

|                                                            |                                                               |
|------------------------------------------------------------|---------------------------------------------------------------|
| <input type="radio"/> Kubernetes Cloud Provider            | <input type="checkbox"/> Disable SSL certificate verification |
| <input checked="" type="checkbox"/> Networking             | <input checked="" type="checkbox"/> NAT mode                  |
| <input checked="" type="checkbox"/> UAA                    | <input type="checkbox"/> Policy API mode                      |
| <input checked="" type="checkbox"/> Host Monitoring        | Pods IP Block ID *                                            |
| <input checked="" type="checkbox"/> In-Cluster Monitoring  | Nodes IP Block ID *                                           |
| <input checked="" type="checkbox"/> Tanzu Mission Control  | T0 Router ID *                                                |
| <input checked="" type="checkbox"/> About the CEIP Program | Floating IP Pool ID *                                         |
| <input checked="" type="checkbox"/> Storage                | Nodes DNS *                                                   |
| <input checked="" type="checkbox"/> Errands                | vSphere Cluster Names *                                       |
| <input checked="" type="checkbox"/> Resource Config        | Kubernetes Service Network CIDR Range *                       |
|                                                            | <input type="text" value="10.100.200.0/24"/>                  |
|                                                            | TKGI Operation Timeout *                                      |
|                                                            | <input type="text" value="60000"/>                            |

[View a larger version of this image.](#)

- **Pods IP Block ID:** Enter the UUID of the IP block to be used for Kubernetes pods. Tanzu Kubernetes Grid Integrated Edition allocates IP addresses for the pods when they are created in Kubernetes. Each time a namespace is created in Kubernetes, a subnet from this IP block is allocated. The current subnet size that is created is /24, which means a maximum of 256 pods can be created per namespace.
- **Nodes IP Block ID:** Enter the UUID of the IP block to be used for Kubernetes nodes. Tanzu Kubernetes Grid Integrated Edition allocates IP addresses for the nodes when they are created in Kubernetes. The node networks are created on a separate IP address space from the pod networks. The current subnet size that is created is /24, which means a maximum of 256 nodes can be created per cluster.  
For more information, including sizes and the IP blocks to avoid using, see [Plan IP Blocks](#) in *Preparing NSX-T Before Deploying Tanzu Kubernetes Grid Integrated Edition*.
- **TO Router ID:** Enter the `t0-tkgi` TO router UUID. Locate this value in the NSX-T UI router overview.
- **Floating IP Pool ID:** Enter the `ip-pool-vips` ID that you created for load balancer VIPs. For more information, see [Plan Network CIDRs](#) in *Network Planning for Installing Tanzu Kubernetes Grid Integrated Edition with NSX-T*. Tanzu Kubernetes Grid Integrated Edition uses the floating IP pool to allocate IP addresses to the load balancers created for each of the clusters. The load balancer routes the API requests to the control plane nodes and the data plane.
- **Nodes DNS:** Enter one or more Domain Name Servers used by the Kubernetes nodes.
- **vSphere Cluster Names:** Enter a comma-separated list of the vSphere clusters where you will deploy Kubernetes clusters. The NSX-T pre-check errand uses this field to verify that the hosts from the specified clusters are available in NSX-T. You can specify clusters in this format:  
`cluster1,cluster2,cluster3`.
- **Kubernetes Service Network CIDR Range:** Specify an IP address and subnet size depending on the number of Kubernetes services that you plan to deploy within a single Kubernetes cluster, for example: `10.100.200.0/24`. The IP address used here is internal to the cluster and can be anything, such as `10.100.200.0`. A /24 subnet provides 256 IPs. If you have a cluster that requires more than 256 IPs, define a larger subnet, such as `/20`.
- Under **TKGI Operation Timeout**, enter the timeout in milliseconds for TKGI-API operation. When needed, increase the **TKGI Operation Timeout** setting to avoid timeouts during cluster deletion in large-scale NSX-T environments. To determine the optimal Operation Timeout setting, see [Cluster Deletion Fails in General Troubleshooting](#).

3. (Optional) Configure a global proxy for all outgoing HTTP and HTTPS traffic from your

Kubernetes clusters and the TKGI API server. See [Using Proxies with Tanzu Kubernetes Grid Integrated Edition on NSX-T](#) for instructions on how to enable a proxy.

4. Under **Allow outbound internet access from Kubernetes cluster vms (IaaS-dependent)**, ignore the **Enable outbound internet access** checkbox.
5. Click **Save**.

## UAA

To configure the UAA server:

1. Click **UAA**.
2. Under **TKGI API Access Token Lifetime**, enter a time in seconds for the TKGI API access token lifetime. This field defaults to **600**.

**UAA Configuration**

---

PKS API Access Token Lifetime (in seconds) \*

600

PKS API Refresh Token Lifetime (in seconds) \*

21600

PKS Cluster Access Token Lifetime (in seconds) \*

600

PKS Cluster Refresh Token Lifetime (in seconds) \*

21600

3. Under **TKGI API Refresh Token Lifetime**, enter a time in seconds for the TKGI API refresh token lifetime. This field defaults to **21600**.
4. Under **TKGI Cluster Access Token Lifetime**, enter a time in seconds for the cluster access token lifetime. This field defaults to **600**.
5. Under **TKGI Cluster Refresh Token Lifetime**, enter a time in seconds for the cluster refresh token lifetime. This field defaults to **21600**.



**Note:** VMware recommends using the default UAA token timeout values. By default, access tokens expire after ten minutes and refresh tokens expire after six hours.

6. Under **Configure created clusters to use UAA as the OIDC provider**, select **Enabled** or **Disabled**. This is a global default setting for TKGI-provisioned clusters. For more information, see [OIDC Provider for Kubernetes Clusters](#).

To configure Tanzu Kubernetes Grid Integrated Edition to use UAA as the OIDC provider:

- Under **Configure created clusters to use UAA as the OIDC provider**, select **Enabled**.

Configure created clusters to use UAA as the OIDC provider.\*

Disabled  
 Enabled

UAA OIDC Groups Claim \*

UAA OIDC Groups Prefix \*

UAA OIDC Username Claim \*

UAA OIDC Username Prefix \*

TKGI cluster client redirect URIs

Configure your UAA user account store with either internal or external authentication mechanisms \*

Internal UAA  
 LDAP Server  
 SAML Identity Provider

**Save**

- For **UAA OIDC Groups Claim**, enter the name of your groups claim. This is used to set a user's group in the JSON Web Token (JWT) claim. The default value is `roles`.
- For **UAA OIDC Groups Prefix**, enter a prefix for your groups claim. This prevents conflicts with existing names. For example, if you enter the prefix `oidc:`, UAA creates a group name like `oidc:developers`. The default value is `oidc:`.
- For **UAA OIDC Username Claim**, enter the name of your username claim. This is

used to set a user's username in the JWT claim. The default value is `user_name`. Depending on your provider, you can enter claims besides `user_name`, like `email` or `name`.

5. For **UAA OIDC Username Prefix**, enter a prefix for your username claim. This prevents conflicts with existing names. For example, if you enter the prefix `oidc:`, UAA creates a username like `oidc:admin`. The default value is `oidc:`.



**Warning:** VMware recommends adding OIDC prefixes to prevent users and groups from gaining unintended cluster privileges. If you change the above values for a pre-existing Tanzu Kubernetes Grid Integrated Edition installation, you must change any existing role bindings that bind to a username or group. If you do not change your role bindings, developers cannot access Kubernetes clusters. For instructions, see [Managing Cluster Access and Permissions](#).

7. (Optional) For **TKGI cluster client redirect URIs**, enter one or more comma-delimited UAA redirect URIs. Configure **TKGI cluster client redirect URIs** to assign persistent UAA `cluster_client_redirect_uri` URIs to your clusters. UAA redirect URIs configured in the **TKGI cluster client redirect URIs** field persist through cluster updates and TKGI upgrades.
8. Select one of the following options:
  - ◊ To use an internal user account store for UAA, select **Internal UAA**. Click **Save** and continue to [\(Optional\) Host Monitoring](#).
  - ◊ To use LDAP for UAA, select **LDAP Server** and continue to [Connecting Tanzu Kubernetes Grid Integrated Edition to an LDAP Server](#).
  - ◊ To use SAML for UAA, select **SAML Identity Provider** and continue to [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#).

## (Optional) Host Monitoring

In **Host Monitoring**, you can configure monitoring of nodes and VMs using Syslog, VMware vRealize Log Insight (vRLI) Integration, or Telegraf.

## Configure TKGI Monitoring Features on Host

Enable Syslog for TKGI?\*

No  
 Yes

Enable VMware vRealize Log Insight Integration?\*

No  
 Yes

Enable Telegraf Outputs?\*

No  
 Yes

**Save**

You can configure one or more of the following:

- **Syslog:** To configure Syslog, see [Syslog](#) below. Syslog forwards log messages from all BOSH-deployed VMs to a syslog endpoint.
- **VMware vRealize Log Insight (vRLI) Integration:** To configure VMware vRealize Log Insight (vRLI) Integration, see [VMware vRealize Log Insight Integration](#) below. The vRLI integration pulls logs from all BOSH jobs and containers running in the cluster, including node logs from core Kubernetes and BOSH processes, Kubernetes event logs, and pod `stdout` and `stderr`.
- **Telegraf:** To configure Telegraf, see [Configuring Telegraf in TKGI](#). The Telegraf agent sends metrics from TKGI API, control plane node, and worker node VMs to a monitoring service, such as Wavefront or Datadog.

For more information about these components, see [Monitoring TKGI and TKGI-Provisioned Clusters](#).

### Syslog

To configure Syslog for all BOSH-deployed VMs in Tanzu Kubernetes Grid Integrated Edition:

1. Click **Host Monitoring**.
2. Under **Enable Syslog for TKGI**, select **Yes**.
3. Under **Address**, enter the destination syslog endpoint.

4. Under **Port**, enter the destination syslog port.
5. Under **Transport Protocol**, select a transport protocol for log forwarding.
6. (Optional) To enable TLS encryption during log forwarding, complete the following steps:
  1. Ensure **Enable TLS** is selected.



**Note:** Logs might contain sensitive information, such as cloud provider credentials. VMware recommends that you enable TLS encryption for log forwarding.

2. Under **Permitted Peer**, provide the accepted fingerprint (SHA1) or name of remote peer. For example, `*.YOUR-LOGGING-SYSTEM.com`.
3. Under **TLS Certificate**, provide a TLS certificate for the destination syslog endpoint.



**Note:** You do not need to provide a new certificate if the TLS certificate for the destination syslog endpoint is signed by a Certificate Authority (CA) in your BOSH certificate store.

7. (Optional) Under **Max Message Size**, enter a maximum message size for logs that are forwarded to a syslog endpoint. By default, the **Max Message Size** field is 10,000 characters.
8. (Optional) Under **Custom Rsyslog Configuration**, enter your RSyslog rules configuration using RainerScript syntax. For more information, see [RainerScript](#) in the RSyslog documentation. For example RSyslog rule configurations, see [Example Custom Rules](#) in the Syslog BOSH GitHub repository.
9. Click **Save**.

## VMware vRealize Log Insight Integration



**Note:** Before you configure the vRLI integration, you must have a vRLI license and vRLI must be installed, running, and available in your environment. You need to provide the live instance address during configuration. For instructions and additional information, see the [vRealize Log Insight documentation](#).

By default, vRLI logging is deactivated. To configure vRLI logging:

1. Under **Enable VMware vRealize Log Insight Integration?**, select **Yes**.

Enable VMware vRealize Log Insight Integration?\*

No  
 Yes

Host \*

Enable SSL?

Disable SSL certificate validation

CA certificate

Rate limiting \*

2. Under **Host**, enter the IP address or FQDN of the vRLI host.
3. (Optional) Select the **Enable SSL?** checkbox to encrypt the logs being sent to vRLI using SSL.
4. Choose one of the following SSL certificate validation options:
  - To skip certificate validation for the vRLI host, select the **Disable SSL certificate validation** checkbox. Select this option if you are using a self-signed certificate in order to simplify setup for a development or test environment.



**Note:** Deactivating certificate validation is not recommended for production environments.

- To enable certificate validation for the vRLI host, clear the **Disable SSL certificate validation** checkbox.
5. (Optional) If your vRLI certificate is not signed by a trusted CA root or other well known certificate, enter the certificate in the **CA certificate** field. Locate the PEM of the CA used to sign the vRLI certificate, copy the contents of the certificate file, and paste them into the field. Certificates must be in PEM-encoded format.
  6. Under **Rate limiting**, enter a time in milliseconds to change the rate at which logs are sent to the vRLI host. The rate limit specifies the minimum time between messages before the

fluentd agent begins to drop messages. The default value `0` means that the rate is not limited, which suffices for many deployments.



**Note:** If your deployment is generating a high volume of logs, you can increase this value to limit network traffic. Consider starting with a lower value, such as `10`, then tuning to optimize for your deployment. A large number might result in dropping too many log entries.

7. Click **Save**. These settings apply to any clusters created after you have saved these configuration settings and clicked **Apply Changes**. If the **Upgrade all clusters errand** has been enabled, these settings are also applied to existing clusters.



**Note:** The Tanzu Kubernetes Grid Integrated Edition tile does not validate your vRLI configuration settings. To verify your setup, look for log entries in vRLI.

## (Optional) In-Cluster Monitoring

In **In-Cluster Monitoring**, you can configure one or more observability components and integrations that run in Kubernetes clusters and capture logs and metrics about your workloads. For more information, see [Monitoring Workers and Workloads](#).

## Configure PKS Monitoring Features Deployed in Cluster

Wavefront Integration\*

- No  
 Yes

Deploy cAdvisor\*

- No  
 Yes
- Enable Metric Sink Resources  
 Enable Log Sink Resources  
 Enable node exporter on workers

**Save**

To configure in-cluster monitoring:

- To configure Wavefront, see [Wavefront](#).
- To configure cAdvisor, see [VMware vRealize Operations Management Pack for Container Monitoring](#).
- To configure sink resources, see:
  - [Metric Sink Resources](#)
  - [Log Sink Resources](#)

You can enable both log and metric sink resources or only one of them.

### Wavefront

You can monitor Kubernetes clusters and pods metrics externally using the integration with [Wavefront by VMware](#).



**Note:** Before you configure Wavefront integration, you must have an active Wavefront account and access to a Wavefront instance. You provide your Wavefront access token during configuration. For additional information, see the [Wavefront](#)

documentation.

To use Wavefront with Windows worker-based clusters, developers must install Wavefront to their clusters manually, using Helm.

To enable and configure Wavefront monitoring:

1. In the Tanzu Kubernetes Grid Integrated Edition tile, select **In-Cluster Monitoring**.
2. Under **Wavefront Integration**, select **Yes**.
3. Under **Wavefront URL**, enter the URL of your Wavefront subscription. For example:

```
https://try.wavefront.com/api
```

4. Under **Wavefront Access Token**, enter the API token for your Wavefront subscription.
5. (Optional) For installations that require a proxy server for outbound Internet access, enable access by entering values for **HTTP Proxy Host**, **HTTP Proxy Port**, **Proxy username**, and **Proxy password**.
6. Click **Save**.

The Tanzu Kubernetes Grid Integrated Edition tile does not validate your Wavefront configuration settings. To verify your setup, look for cluster and pod metrics in Wavefront.

## VMware vRealize Operations Management Pack for Container Monitoring

You can monitor Tanzu Kubernetes Grid Integrated Edition Kubernetes clusters with VMware vRealize Operations Management Pack for Container Monitoring.

To integrate Tanzu Kubernetes Grid Integrated Edition with VMware vRealize Operations Management Pack for Container Monitoring, you must deploy a container running [cAdvisor](#) in your TKGI deployment.

cAdvisor is an open source tool that provides monitoring and statistics for Kubernetes clusters.

To deploy a cAdvisor container:

1. Select **In-Cluster Monitoring**.
2. Under **Deploy cAdvisor**, select **Yes**.
3. Click **Save**.

For more information about integrating this type of monitoring with TKGI, see the [VMware vRealize Operations Management Pack for Container Monitoring User Guide](#) and [Release Notes](#) in the VMware documentation.

## Metric Sink Resources

You can configure TKGI-provisioned clusters to send Kubernetes node metrics and pod metrics to metric sinks. For more information about metric sink resources and what to do after you enable them in the tile, see [Sink Resources in Monitoring Workers and Workloads](#).

To enable clusters to send Kubernetes node metrics and pod metrics to metric sinks:

1. In **In-Cluster Monitoring**, select **Enable Metric Sink Resources**. If you enable this checkbox,

Tanzu Kubernetes Grid Integrated Edition deploys Telegraf as a `DaemonSet`, a pod that runs on each worker node in all your Kubernetes clusters.

2. (Optional) To enable Node Exporter to send worker node metrics to metric sinks of kind `ClusterMetricSink`, select **Enable node exporter on workers**. If you enable this checkbox, Tanzu Kubernetes Grid Integrated Edition deploys Node Exporter as a `DaemonSet`, a pod that runs on each worker node in all your Kubernetes clusters.

For instructions on how to create a metric sink of kind `ClusterMetricSink` for Node Exporter metrics, see [Create a ClusterMetricSink Resource for Node Exporter Metrics](#) in *Creating and Managing Sink Resources*.

3. Click **Save**.

## Log Sink Resources

You can configure TKGI-provisioned clusters to send Kubernetes API events and pod logs to log sinks. For more information about log sink resources and what to do after you enable them in the tile, see [Sink Resources](#) in *Monitoring Workers and Workloads*.

To enable clusters to send Kubernetes API events and pod logs to log sinks:

1. Select **Enable Log Sink Resources**. If you enable this checkbox, Tanzu Kubernetes Grid Integrated Edition deploys Fluent Bit as a `DaemonSet`, a pod that runs on each worker node in all your Kubernetes clusters.
2. Click **Save**.

## Tanzu Mission Control

Tanzu Mission Control integration lets you monitor and manage Tanzu Kubernetes Grid Integrated Edition clusters from the Tanzu Mission Control console, which makes the Tanzu Mission Control console a single point of control for all Kubernetes clusters. For more information about Tanzu Mission Control, see the [VMware Tanzu Mission Control](#) home page.

To integrate Tanzu Kubernetes Grid Integrated Edition with Tanzu Mission Control:

1. Confirm that the TKGI API VM has internet access and can connect to [cna.tmc.cloud.vmware.com](https://cna.tmc.cloud.vmware.com) and the other outbound URLs listed in the [What Happens When You Attach a Cluster](#) section of the Tanzu Mission Control Product documentation.
2. Navigate to the **Tanzu Kubernetes Grid Integrated Edition** tile > the **Tanzu Mission Control** pane and select **Yes** under **Tanzu Mission Control Integration**.

Tanzu Mission Control Integration\*

No  
 Yes

Tanzu Mission Control URL \*

VMware Cloud Services API Token \*

Tanzu Mission Control Cluster Group \*

Tanzu Mission Control Cluster Name Prefix \*

**Save**

3. Configure the fields below:

- **Tanzu Mission Control URL:** Enter the Org URL of your Tanzu Mission Control subscription, without a trailing slash (/). For example, `YOUR-ORG.tmc.cloud.vmware.com`.
- **VMware Cloud Services API token:** Enter your API token to authenticate with VMware Cloud Services APIs. You can retrieve this token by logging in to [VMware Cloud Services](#) and viewing your account information.
- **Tanzu Mission Control Cluster Group:** Enter the name of a Tanzu Mission Control cluster group.

The name can be `default` or another value, depending on your role and access policy:

- `Org Member` users in VMware cloud services have a `service.admin` role in Tanzu Mission Control. These users:
  - By default, can create and attach clusters only in the `default` cluster group.
  - Can create and attach clusters to other cluster groups after an `organization.admin` user grants them the `clustergroup.admin` or

`clustergroup.edit` role for those groups.

- `Org Owner` users in VMware cloud services have `organization.admin` permissions in Tanzu Mission Control. These users:
  - Can create cluster groups.
  - Can grant `clustergroup` roles to `service.admin` users through the Tanzu Mission Control Access Policy view.

For more information about role and access policy, see [Access Control](#) in the VMware Tanzu Mission Control Product documentation.

- ◊ **Tanzu Mission Control Cluster Name Prefix:** Enter a name prefix for identifying the Tanzu Kubernetes Grid Integrated Edition clusters in Tanzu Mission Control.

4. Click **Save**.



**Warning:** After the Tanzu Kubernetes Grid Integrated Edition tile is deployed with a configured cluster group, the cluster group cannot be updated.



**Note:** When you upgrade your Kubernetes clusters and have Tanzu Mission Control integration enabled, existing clusters will be attached to Tanzu Mission Control.

## VMware CEIP

Tanzu Kubernetes Grid Integrated Edition-provisioned clusters send usage data to the TKGI control plane for storage. The VMware Customer Experience Improvement Program (CEIP) provides the option to also send the cluster usage data to VMware to improve customer experience.

To configure Tanzu Kubernetes Grid Integrated Edition CEIP Program settings:

1. Click **CEIP**.
2. Review the information about the CEIP.

## About the CEIP Program

VMware's Customer Experience Improvement Program ("CEIP") provides VMware with information that enables VMware to improve its products and services, to fix problems, and to advise you on how best to deploy and use our products. As part of the CEIP, VMware collects technical information about your organization's use of VMware products and services on a regular basis in association with your organization's VMware license key(s). This information does not personally identify any individual.

Customers who participate in the CEIP receive proactive support benefits that include a weekly report based on telemetry data. Contact your Customer Success Manager to subscribe to this report. You can view a sample report at <https://via.vmware.com/TKGI>.

Additional information regarding the data collected through CEIP and the purposes for which it is used by VMware is set forth in the Trust & Assurance Center at <http://www.vmware.com/trustvmware/ceip.html>. If you prefer not to participate in VMware's CEIP for this product, you should uncheck the box below. You may join or leave VMware's CEIP for this product at any time.

Join the VMware Customer Experience Improvement Program\*

- No
- Yes

Please enter your Entitlement Account Number or your Tanzu Customer Number. We need this information to generate a report for you

Please enter a label for this TKGI Installation

Assign a label to this TKGI Installation for use in your reports

Please select how you will be using this TKGI Installation \*

- Demo or Proof-of-concept
- Development or Pre-production
- Production
- I do not wish to provide this information

**Save**

[View a larger version of this image.](#)

3. If you wish to participate in CEIP, select **Yes**. Otherwise, select **No**.
4. If you selected the **Yes**, complete the following:
  - ◊ (Optional) Enter your entitlement account number or Tanzu customer number. If you are a VMware customer, you can find your entitlement account number in your **Account Summary** on [my.vmware.com](https://my.vmware.com). If you are a Pivotal customer, you can find your Pivotal Customer Number in your Pivotal Order Confirmation email.
  - ◊ (Optional) Enter a descriptive name for your TKGI installation. The label you assign to this installation will be used in CEIP reports to identify the environment.
5. To provide information about the purpose for this installation, select an option.

Please select how you will be using this TKGI Installation \*

- Demo or Proof-of-concept
- Development or Pre-production
- Production
- I do not wish to provide this information

6. Click **Save**.

## Storage

In **Storage Configurations**, you can configure vSphere CNS settings.

## Storage Configurations

vSphere CSI Driver Integration (Enable automatic installation of vSphere CSI driver on all clusters)\*

- No
- Yes (Warning: Manually deployed vSphere CSI driver must be removed after enabling this feature!)

**Save**

To configure vSphere CNS:

1. Select **Storage**.
2. (Optional) To enable automatic installation of the vSphere CSI driver on all clusters, select **Yes**.



**Warning:** If you have existing clusters with a manually deployed vSphere CSI driver, you must remove the manually deployed driver after enabling this feature. For more information, see [Deploying Cloud Native Storage \(CNS\) on vSphere](#).

## Errands

Errands are scripts that run at designated points during an installation.

To configure which post-deploy and pre-delete errands run for Tanzu Kubernetes Grid Integrated Edition:

1. Make a selection in the dropdown next to each errand.



**Note:** We recommend that you use the default settings for all errands except for the **NSX-T validation** and **Run smoke tests** errands.

2. (Optional) Set the **NSX-T validation** errand to **On**.

This errand verifies the NSX-T objects.

3. (Optional) Set the **Run smoke tests** errand to **On**.

This errand uses the TKGI CLI to create a Kubernetes cluster and then delete it. If the creation or deletion fails, the errand fails and the installation of the Tanzu Kubernetes Grid Integrated Edition tile is aborted.

4. (Optional) To ensure that all of your cluster VMs are patched, configure the **Upgrade all clusters errand** errand to **On**.

Updating the Tanzu Kubernetes Grid Integrated Edition tile with a new Linux stemcell and

the **Upgrade all clusters errand** enables triggers the rolling of every Linux VM in each Kubernetes cluster. Similarly, updating the Tanzu Kubernetes Grid Integrated Edition tile with a new Windows stemcell triggers the rolling of every Windows VM in your Kubernetes clusters.



**Note:** VMware recommends that you review the Tanzu Network metadata and confirm stemcell version compatibility before using the Tanzu Network APIs to update the stemcells in your automated pipeline. For more information, see the [API reference](#).

## Resource Config

To modify the resource configuration of Tanzu Kubernetes Grid Integrated Edition, follow the steps below:

1. Select **Resource Config**.
2. For each job, review the **Automatic** values in the following fields:
  - ◊ **INSTANCES**: Tanzu Kubernetes Grid Integrated Edition defaults to the minimum configuration. If you want a highly available configuration (beta), scale the number of VM instances as follows:
    1. To configure your Tanzu Kubernetes Grid Integrated Edition database for high availability (beta), increase the **INSTANCES** value for **TKGI Database** to **3**.
    2. To configure your Tanzu Kubernetes Grid Integrated Edition API and UAA for high availability (beta), increase the **INSTANCES** value for **TKGI API** to **2** or more.



**Warning:** High availability mode is a beta feature. Do not scale your **TKGI API** or **TKGI Database** to more than one instance in production environments.



**Note:** On vSphere with NSX-T, you must manually deploy an NSX-T load balancer so that you can select it as part of the resource configuration. For more information, see [Provisioning an NSX-T Load Balancer for the TKGI API Server](#).

- ◊ **VM TYPE**: By default, the **TKGI Database** and **TKGI API** jobs are set to the same **Automatic** VM type. If you want to adjust this value, we recommend that you select the same VM type for both jobs.



**Note:** The **Automatic VM TYPE** values match the recommended resource configuration for the **TKGI API** and **TKGI Database** jobs.

- ◊ **PERSISTENT DISK TYPE**: By default, the **TKGI Database** and **TKGI API** jobs are set

to the same persistent disk type. If you want to adjust this value, you can change the persistent disk type for each of the jobs independently. Using the same persistent disk type for both jobs is not required.

3. Under each job, leave **NSX-T CONFIGURATION** and **NSX-V CONFIGURATION** blank.



**Warning:** To avoid workload downtime, use the resource configuration recommended in [About Tanzu Kubernetes Grid Integrated Edition Upgrades](#) and [Maintaining Workload Uptime](#).

## Step 3: Apply Changes

After configuring the Tanzu Kubernetes Grid Integrated Edition tile, follow the steps below to deploy the tile:

1. Return to the Ops Manager Installation Dashboard.
2. Click **Review Pending Changes**. Select the product that you intend to deploy and review the changes. For more information, see [Reviewing Pending Product Changes](#).
3. Click **Apply Changes**.

## Step 4: Install the TKGI and Kubernetes CLIs

The TKGI CLI and the Kubernetes CLI help you interact with your Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes clusters and Kubernetes workloads. To install the CLIs, follow the instructions below:

- [Installing the TKGI CLI](#)
- [Installing the Kubernetes CLI](#)

## Step 5: Verify NAT Rules

If you are using NAT mode, verify that you have created the required NAT rules for the Tanzu Kubernetes Grid Integrated Edition Management Plane. See [Create Management Plane](#) in *Installing and Configuring NSX-T Data Center v3.0 for TKGI* for details.

In addition, for NAT and No-NAT modes, verify that you created the required NAT rule for Kubernetes control plane nodes to access NSX-T Manager. For details, see [Create IP Blocks and Pool for Compute Plane](#) in *Installing and Configuring NSX-T Data Center v3.0 for TKGI*.

If you want your developers to be able to access the TKGI CLI from their external workstations, create a DNAT rule that maps a routable IP address to the TKGI API VM. This must be done after Tanzu Kubernetes Grid Integrated Edition is successfully deployed and it has an IP address. See [Create Management Plane](#) in *Installing and Configuring NSX-T Data Center v3.0 for TKGI* for details.

## Step 6: Configure Authentication for Tanzu Kubernetes Grid Integrated Edition

Follow the procedures in [Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users](#) on

vSphere in *Installing Tanzu Kubernetes Grid Integrated Edition > vSphere*.

## Next Steps

After installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T integration, complete the following tasks:

- Integrate VMware Harbor with Tanzu Kubernetes Grid Integrated Edition to store and manage container images. For more information, see [Integrating VMware Harbor Registry with Tanzu Kubernetes Grid Integrated Edition](#).
- [Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on vSphere](#).
- [Creating an Tanzu Kubernetes Grid Integrated Edition Cluster](#).

## Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on vSphere

This topic describes how to create admin users in VMware Tanzu Kubernetes Grid Integrated Edition with User Account and Authentication (UAA). Creating at least one admin user is a necessary step during the initial set up of Tanzu Kubernetes Grid Integrated Edition.

### Overview

UAA is the identity management service for Tanzu Kubernetes Grid Integrated Edition. Tanzu Kubernetes Grid Integrated Edition includes a UAA server, which is hosted on the TKGI API VM.

To interact with the UAA server, you can use the UAA Command Line Interface (UAAC). You can either run UAAC commands from the Ops Manager VM or install UAAC on your local workstation.

### Prerequisites

Before setting up admin users for Tanzu Kubernetes Grid Integrated Edition, you must have one of the following:

- SSH access to the Ops Manager VM
- A machine that can connect to your TKGI API VM

### Step 1: Connect to the TKGI API VM

You can connect to the TKGI API VM from the Ops Manager VM or from a different machine such as your local workstation.

#### Option 1: Connect through the Ops Manager VM

You can connect to TKGI API VM by logging in to the Ops Manager VM through SSH.

To SSH into the Ops Manager VM on vSphere, do the following:

1. Locate the credentials that were used to import the Ops Manager `.ova` or `.ovf` file into your virtualization system. You configured these credentials when you installed Ops Manager and used them to complete the [Prepare vSphere](#) steps in *Deploying Ops Manager on vSphere*.

2. Change the permissions for your private SSH key by running the following command:

```
chmod 600 PRIVATE-KEY
```

Where `PRIVATE-KEY` is the name of your private SSH key.

3. SSH into the Ops Manager VM by running the following command:

```
ssh -i PRIVATE-KEY ubuntu@OPS-MANAGER-FQDN
```

Where `OPS-MANAGER-FQDN` is the fully qualified domain name (FQDN) of Ops Manager.

For example:

```
$ ssh -i id_rsa ubuntu@my-opsmanager-fqdn.example.com
```

4. Proceed to the [Log In as a UAA Admin](#) section to create admin users with UAAC.

## Option 2: Connect through a Non-Ops Manager Machine

To connect to the TKGI API VM and run UAA commands, do the following:

1. Install UAAC on your machine. For example:

```
gem install cf-uaac
```

2. Download a copy of your Ops Manager root CA certificate to the machine. To download the certificate, do the following:
  1. In a web browser, navigate to the FQDN of Ops Manager and log in.
  2. In Ops Manager, navigate to **Settings** in the drop-down menu under your username.
  3. Click **Advanced Options**.
  4. On the **Advanced Options** configuration page, click **Download Root CA Cert**.
  5. Move the certificate to a secure location on your machine and record the path.
3. Proceed to the [Log In as a UAA Admin](#) section to create admin users with UAAC.

## Step 2: Log In as a UAA Admin

Before creating TKGI users, you must log in to the UAA server as a UAA admin. To log in to the UAA server, do the following:

1. Retrieve the UAA management admin client secret:
  1. In a web browser, navigate to the Ops Manager **Installation Dashboard** and click the **Tanzu Kubernetes Grid Integrated Edition** tile.
  2. Click the **Credentials** tab.
  3. Click **Link to Credential** next to **Pks Uaa Management Admin Client** and copy the value of `secret`.

2. Target your UAA server by running the following command:

```
uaac target https://TKGI-API:8443 --ca-cert CERTIFICATE-PATH
```

Where:

- TKGI-API is the domain name of your TKGI API server. You entered this domain name in the **Tanzu Kubernetes Grid Integrated Edition** tile > **TKGI API** > **API Hostname (FQDN)**.
- CERTIFICATE-PATH is the path to your Ops Manager root CA certificate. Provide this certificate to validate the TKGI API certificate with SSL.
  - If you are logged in to the Ops Manager VM, specify `/var/tempest/workspaces/default/root_ca_certificate` as the path. This is the default location of the root certificate on the Ops Manager VM.
  - If you downloaded the Ops Manager root CA certificate to your machine, specify the path where you stored the certificate.

For example:

```
$ uaac target api.tkgi.example.com:8443 -ca-cert /var/tempest/workspaces/default/root_ca_certificate
```



**Note:** If you receive an `Unknown key: Max-Age = 86400` warning message, you can ignore it because it has no impact.

3. Authenticate with UAA by running the following command:

```
uaac token client get admin -s ADMIN-CLIENT-SECRET
```

Where `ADMIN-CLIENT-SECRET` is your UAA management admin client secret that you retrieved in a previous step. The client username is `admin`.

## Step 3: Assign Tanzu Kubernetes Grid Integrated Edition Cluster Scopes

The `pks.clusters.manage` and `pks.clusters.admin` UAA scopes grant users the ability to create and manage Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition. For information about UAA scopes in Tanzu Kubernetes Grid Integrated Edition, see [UAA Scopes for Tanzu Kubernetes Grid Integrated Edition Users](#).

To create Tanzu Kubernetes Grid Integrated Edition users with the `pks.clusters.manage` or `pks.clusters.admin` UAA scope, perform one or more of the following procedures based on the needs of your deployment:

- To assign TKGI cluster scopes to an individual user, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an Individual User](#). Follow this procedure if you selected

**Internal UAA** when you configured **UAA** in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#).

- To assign TKGI cluster scopes to an LDAP group, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an External LDAP Group](#). Follow this procedure if you selected **LDAP Server** when you configured **UAA** in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Installing Tanzu Kubernetes Grid Integrated Edition TKGI on vSphere](#).
- To assign TKGI cluster scopes to a SAML group, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an External SAML Group](#). Follow this procedure if you selected **SAML Identity Provider** when you configured **UAA** in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Installing Tanzu Kubernetes Grid Integrated Edition TKGI on vSphere](#).
- To assign TKGI cluster scopes to a client, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to a Client](#).

## Next Step

After you create admin users in Tanzu Kubernetes Grid Integrated Edition, the admin users can create and manage Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition. For more information, see [Managing Kubernetes Clusters and Workloads](#).

## Advanced Configurations for Tanzu Kubernetes Grid Integrated Edition on vSphere with VMware NSX

This topic lists the sections to follow when installing VMware Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T Data Center.

## Post-Installation NSX-T Configurations

After you have installed Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T, refer to the following sections for additional NSX-T configuration options:

- [Deploying an NSX-T Load Balancer](#)
- [Defining and Using Network Profiles](#)
- [Using Proxies with Tanzu Kubernetes Grid Integrated Edition on NSX-T](#)
- [Configuring Ingress Resources and Load Balancer Services](#)
- [Isolating Tenants](#)
- [Implementing a Multi-Foundation Tanzu Kubernetes Grid Integrated Edition Deployment](#)

## Provisioning a VMware NSX Load Balancer for the TKGI API Server

This topic describes how to deploy an NSX-T load balancer for the Tanzu Kubernetes Grid Integrated Edition API Server.

## About the NSX-T Load Balancer for the TKGI API Server

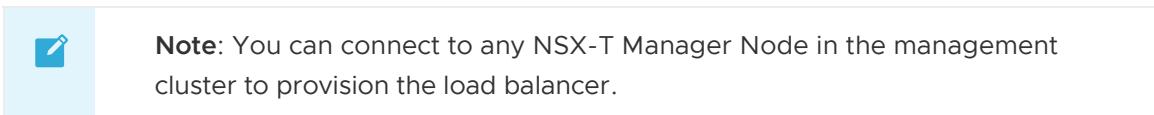
If you deploy Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T with the TKGI API in high-availability mode, you must configure an NSX-T load balancer for the TKGI API traffic. For more information, see [Load Balancers in Tanzu Kubernetes Grid Integrated Edition Deployments on vSphere with NSX-T](#).

To provision an NSX-T load balancer for the TKGI API Server VM, complete the following steps.

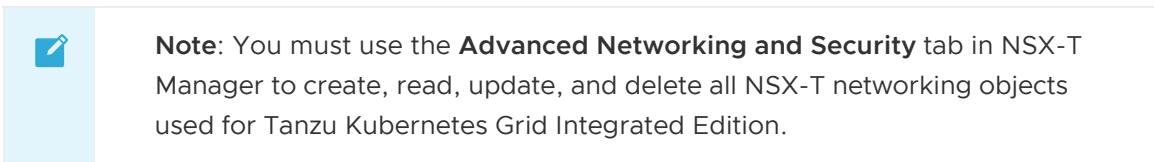
### Step 1: Create NSGroup

If you are using a Dynamic Server Pool, create an NSGroup as described in this step. If you are using a Static Server Pool, skip this step and proceed to Step 2.

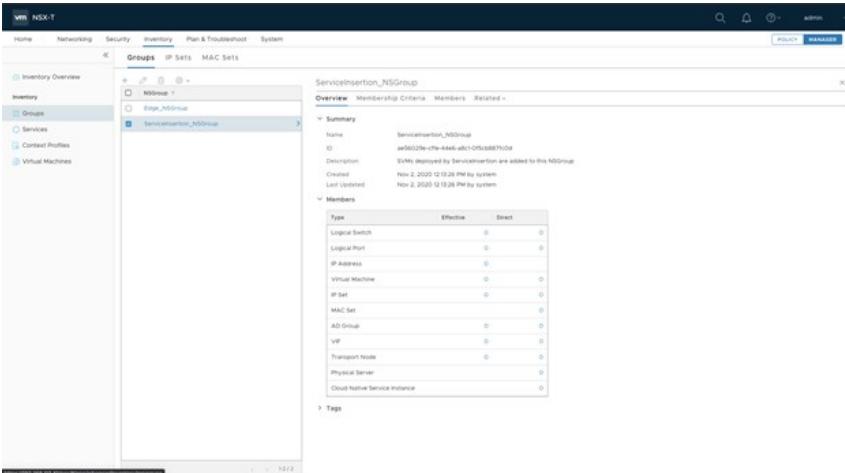
1. Log in to an NSX-T Manager Node.



2. Select the **Advanced Networking & Security** tab.



3. Select **Inventory > Groups**.



4. Click **+ADD** to add a new NSGroup.
5. Enter a name for the NSGroup, for example `tkgi-api`.



6. Click **ADD**.

## Step 2: Create Two Virtual Servers

The TKGI API Server virtual machine hosts two server processes and exposes two ports: the TKGI API Server on port 9021, and the UAA server on port 8443. Each NSX-T Virtual Server listens on one port. Thus, you need two Virtual Servers, one for the TKGI API server and the other for UAA.

If you deploy your Tanzu Kubernetes Grid Integrated Edition using [No-NAT with Virtual Switch \(VSS/VDS\) Topology](#), You only need to deploy ONE virtual Server

### Create a Virtual Server for the TKGI API Server

1. In NSX-T Manager, select **Networking > Load Balancing > Virtual Servers**.
2. Click Add.
3. Configure General Properties for the Virtual Server.
  - Name: `tkgi-api-server`, for example
  - Application Types: Choose **Layer 7 TCP**. Or, you can choose Layer 4, in which case you don't have to configure SSL for the virtual server. For [No-NAT with Virtual Switch \(VSS/VDS\) Topology](#), Choose Layer 4
  - Application Profile: `default-http-lb-app-profile`
  - Access Log: `Disabled`
  - Click **Next**
4. Configure Virtual Server Identifiers.
  - IP Address: Enter an IP address from the floating pool, such as `192/168.160.108`
  - Port: **9021** (for the TKGS API Server). For [No-NAT with Virtual Switch \(VSS/VDS\) Topology](#), Set Port: **9021,8443\***
  - Click **Next**
5. Configure Server Pool and Rules.
  - Click **Create a New Server Pool**
  - Name the server pool, for example `tkgi-api-server`
  - Load Balancing Algorithm: **ROUND\_ROBIN** (for example)
  - Click **Next**
6. Configure SNAT Translation for the Server Pool:

- ◊ Translation Mode: Auto Map. For [No-NAT with Virtual Switch \(VSS/VDS\) Topology](#), Translation Mode: **IP List**. IP Address: Same IP address entered inside Virtual Server Identifiers.
  - ◊ Click **Next**
7. OPTION 1: Configure Pool Members for the Static Server Pool:
- ◊ Membership Type: **Static**
  - ◊ Leave members empty. This will be added automatically later when you apply changes in Ops Manager.
  - ◊ Click **Next**
8. OPTION 2: Configure Pool Members for the Dynamic Server Pool:
- ◊ Membership Type: **Dynamic**
  - ◊ Set NSGroup as the NSGroup name created in Step 1, such as **tkgi-api**
  - ◊ Set Max Group IP Address List to 3, since we can only have up to 3 TKGI API instances
  - ◊ Click **Next** For [No-NAT with Virtual Switch \(VSS/VDS\) Topology](#),
  - ◊ Membership Type: **Static**
  - ◊ Static Membership: Add your Tanzu Kubernetes Grid Integrated Edition API Server one by one, leave port column empty
  - ◊ Click **Next**
9. Configure Health Monitors for the virtual server:
- ◊ Click **Create A New Active Monitor**
  - ◊ Set **Name**, for example **tkgi-api-server**
  - ◊ Set Health Check Protocol **LBHttpsMonitor**
  - ◊ Set the port as **9021**
  - ◊ Leave other fields as default
  - ◊ Click **Next**
10. Configure Health Check Parameters:
- ◊ Choose **High Security for SSL Ciphers**
  - ◊ Select **TLS\_V1\_2** as SSL Protocols
  - ◊ Set HTTP Method as **GET**
  - ◊ Set HTTP Request URL as **/actuator/health**
  - ◊ Set HTTP Request Header as **200**
  - ◊ Click **Finish**
11. Configure **New Server Pool > Health Monitors**:
- ◊ Set Active Health Monitor as the what was created: **tkgi-api-server**
  - ◊ Click **Finish**
12. Configure the Virtual Server:

- ◊ Set the Default Server Pool to what you just created: **tkgi-api-server**
- ◊ Click **Next**

13. Persistence Profiles is optional. Click **Next**.
14. Configure Client Side SSL (only if L7 Application Type is selected). Use the default certificates. Click **Next**.
15. Configure Server Side SSL (only if L7 Application Type is selected). Use the default certificates. Click **Finish**.

The screenshot shows the VMware NSX-T Management interface. The main window displays the 'Virtual Servers' tab under the 'Load Balancing' section. A message indicates 'No Virtual Servers found.' Below this, there is a table header with columns: Name, Protocol, IP/Port, Default Server Pool, Load Balancer, and Operational State.

The bottom half of the screen shows the 'Add New Virtual Server' dialog, divided into two tabs: 'General Properties' and 'Virtual Server Identifiers'. The 'General Properties' tab is active, showing the following fields:

- Name:** pks-api-server
- Description:** (empty)
- Load Balancer Application Profile:** (description text about application profiles)
- Application Type:** Layer 7 (radio button selected)
- Application Profile:** default-http-lb-app-profile
- Access Log:** Disabled (checkbox)

At the bottom of this tab are 'CANCEL' and 'NEXT' buttons. The 'Virtual Server Identifiers' tab shows the following fields:

- IP Address:** 192.168.160.108
- Port:** 9021 (with a note: 'Specify port (e.g. 8080) or port range (e.g. 80-90) or both separated by comma (e.g. 8080, 80-90, 20)')
- Protocol:** TCP
- Advanced Properties:** Maximum Concurrent Connection and Maximum New Connection Rate (both empty)
- Default Pool Member Port:** (empty with a note: 'Specify port (e.g. 8080) or port range (e.g. 80-90) or both separated by comma (e.g. 8080, 80-90, 20)')

At the bottom of this tab are 'CANCEL', 'BACK', and 'NEXT' buttons.

**Add New Virtual Server**

1 General Properties  
2 Virtual Server Identifiers  
**3 Server Pool and Rules**  
4 Load Balancing Profiles  
A Persistence Profiles  
B Client Side SSL  
C Server Side SSL

**Server Pool and Rules**

Default Server Pool  Create A New Server Pool  
Sorry Server Pool  Create A New Server Pool

Load Balancer Rules

HTTP Request Rewrite Phase 0

+ ADD  CLONE  DELETE  MOVE UP  MOVE DOWN

| Priority        | Match Conditions | Match Strategy | Actions |
|-----------------|------------------|----------------|---------|
| No Rules found. |                  |                |         |

> HTTP Request Forwarding Phase 0

CANCEL BACK NEXT

**Add New Server Pool**

1 General Properties  
2 SNAT Translation  
3 Pool Members  
4 Health Monitors

**General Properties**

Name  Description   
Load Balancing Algorithm

Advanced Properties

TCP Multiplexing  Disabled   
Maximum Multiplexing Connections

CANCEL NEXT

**Add New Server Pool**

1 General Properties  
**2 SNAT Translation**  
3 Pool Members  
4 Health Monitors

**SNAT Translation**

Three Modes based on the topology are supported. In case of Inline deployment of Load Balancer, use Transparent (NO\_SNAT) to preserve original Client IP and Port. Auto Map mode uses LB interface IP and ephemeral port. In scenarios where both Clients and Pool Members are attached to the same Logical Router, SNAT (Auto Map or IP List) must be used.

Translation Mode \*  Transparent  Auto Map  IP List

CANCEL BACK NEXT

Add New Server Pool

- [1 General Properties](#)
- [2 SNAT Translation](#)
- [3 Pool Members](#)
- [4 Health Monitors](#)

### Pool Members

Pool Members can either be Static members that allows you to add IPs and Ports of individual servers or Dynamic Members as defined by NSGroup Membership Criteria. The admin state in case of the Dynamic Members can be set after Server Pool creation in the Members section of the Server Pool. Currently only IPv4 addressing is supported.

Membership Type  Static  Dynamic

Static Membership

| Name                   | IP | Port | Weight | State | Backup Member | Max. Concurrent Connection |
|------------------------|----|------|--------|-------|---------------|----------------------------|
| No Pool Members found. |    |      |        |       |               |                            |

[+ ADD](#) [CLONE](#) [DELETE](#)

[COLUMNS](#) [Pool Members](#)

[CANCEL](#) [BACK](#) [NEXT](#)

Add New Server Pool

- [1 General Properties](#)
- [2 SNAT Translation](#)
- [3 Pool Members](#)
- [4 Health Monitors](#)

### Pool Members

Pool Members can either be Static members that allows you to add IPs and Ports of individual servers or Dynamic Members as defined by NSGroup Membership Criteria. The admin state in case of the Dynamic Members can be set after Server Pool creation in the Members section of the Server Pool. Currently only IPv4 addressing is supported.

Membership Type  Static  Dynamic

Dynamic Membership

NSGroup \*

Max Group IP Address List

Limits the max number of pool members to the specified value. If not specified, allows the dynamic pool to grow up to the load balancer max pool member capacity.

NOTE: Pools using NSGroup must use Active Health Monitors with "Monitoring Port" specified

[CANCEL](#) [BACK](#) [NEXT](#)

Add New Server Pool

- [1 General Properties](#)
- [2 SNAT Translation](#)
- [3 Pool Members](#)
- [4 Health Monitors](#)

### Health Monitors

Minimum Active Members

Active Health Monitor  Create A New Active Monitor

Passive Health Monitor  Create A New Passive Monitor

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

Add New Active Health Monitor

- 1 Monitor Properties**
- 2 Health Check Parameters**

### Monitor Properties

|                             |                                             |
|-----------------------------|---------------------------------------------|
| Name *                      | <input type="text" value="pks-api-server"/> |
| Description                 | <input type="text"/>                        |
| Health Check Protocol *     | LbHttpsMonitor                              |
| Monitoring Port             | <input type="text" value="9021"/>           |
| Monitoring Interval (sec) * | <input type="text" value="5"/>              |
| Fall Count *                | <input type="text" value="3"/>              |
| Rise Count *                | <input type="text" value="3"/>              |
| Timeout Period (sec) *      | <input type="text" value="15"/>             |

CANCEL NEXT

Add New Active Health Monitor

- Monitor Properties**
- Health Check Parameters**

### SSL and HTTP Health Check Parameters

against the received response; and the Pool Member is considered healthy only if there is a complete match.

#### HTTP Request Configuration

| HTTP Method          | GET                                                                                                                                                                                                                                                                     |             |              |   |  |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|--------------|---|--|
| HTTP Request URL     | /actuator/health                                                                                                                                                                                                                                                        |             |              |   |  |
| HTTP Request Version | HTTP_VERSION_1_1                                                                                                                                                                                                                                                        |             |              |   |  |
| HTTP Request Headers | + ADD <span style="color: #0056b3;">DELETE</span> <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th>Header Name</th> <th>Header Value</th> </tr> </thead> <tbody> <tr> <td> X</td> <td></td> </tr> </tbody> </table> | Header Name | Header Value | X |  |
| Header Name          | Header Value                                                                                                                                                                                                                                                            |             |              |   |  |
| X                    |                                                                                                                                                                                                                                                                         |             |              |   |  |
| HTTP Request Body    | <input type="text"/>                                                                                                                                                                                                                                                    |             |              |   |  |

#### HTTP Response Configuration

|                    |     |
|--------------------|-----|
| HTTP Response Code | 200 |
|--------------------|-----|

CANCEL BACK FINISH

Add New Server Pool

- 1 General Properties**
- 2 SNAT Translation**
- 3 Pool Members**
- 4 Health Monitors**

### Health Monitors

|                        |                                                                                                                                 |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------|
| Minimum Active Members | <input type="text" value="1"/>                                                                                                  |
| Active Health Monitor  | <input type="text" value="pks-api-service"/> <span style="color: #0056b3; font-size: small;">Create A New Active Monitor</span> |
| Passive Health Monitor | <input type="text"/> <span style="color: #0056b3; font-size: small;">Create A New Passive Monitor</span>                        |

CANCEL BACK FINISH

**Add New Virtual Server**

**Server Pool and Rules**

Default Server Pool: **pks-api-server** Create A New Server Pool  
 pks-api-server  
 10.0.2.14:443

Sorry Server Pool: Create A New Server Pool

Load Balancer Rules

HTTP Request Rewrite Phase 0

+ ADD  CLONE  DELETE  MOVE UP  MOVE DOWN

| Priority        | Match Conditions | Match Strategy | Actions |
|-----------------|------------------|----------------|---------|
| No Rules found. |                  |                |         |

> HTTP Request Forwarding Phase 0

CANCEL BACK NEXT

**Add New Virtual Server**

**Persistence Profiles**

Persistence Profiles

Persistence:  Disabled

loadBalancing.common.emptyText

Create A New Source Persistence Profile

Cookie Persistence

Create A New Cookie Persistence Profile

CANCEL BACK NEXT

**Edit Virtual Server**

**Client Side SSL**

Client Side SSL:  Enabled

Client SSL Profile: **nsx-default-client-ssl-profile** Create A New Client SSL Profile

Default Certificate: **pka-nsx-certificate**

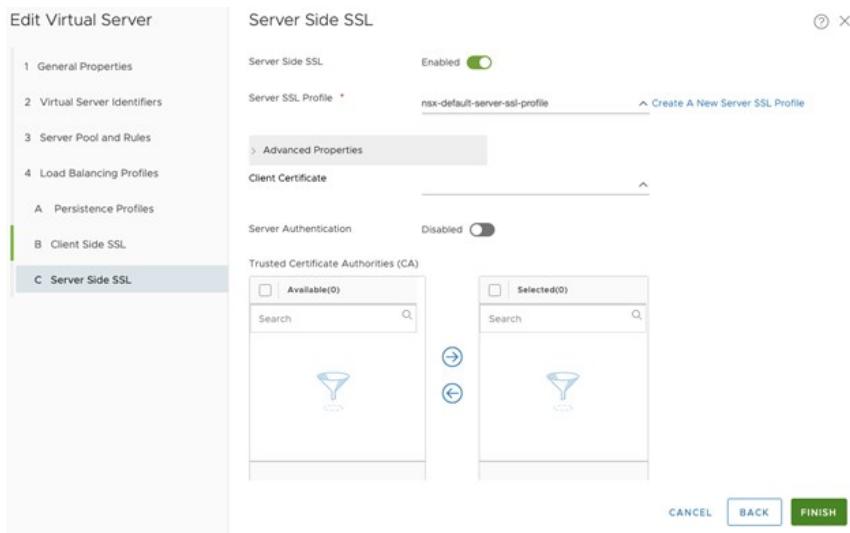
> Advanced Properties

SNI Certificates

| Available(8)                                           | Selected(0)              |
|--------------------------------------------------------|--------------------------|
| <input type="checkbox"/> S16d2c54-8877-4258-9a15-      | <input type="checkbox"/> |
| <input type="checkbox"/> APIH-AR certificate for node  | <input type="checkbox"/> |
| <input type="checkbox"/> LocalManager                  | <input type="checkbox"/> |
| <input type="checkbox"/> mp-cluster certificate for nc | <input type="checkbox"/> |
| <input type="checkbox"/> pks-6be14ec6-9834-4528-a      | <input type="checkbox"/> |

Mandatory Client Authentication:  Enabled

CANCEL BACK NEXT



## Create a Virtual Server for the UAA Server

Skip this if you deploy as [No-NAT with Virtual Switch \(VSS/VDS\) Topology](#)

1. Select the tkgs-api-server Virtual Server you created and click **Clone**.
2. Click **Edit** to change the content and configure the virtual server for UAA.
3. Configure General Properties
  - ◊ Set the Name as **tkgi-api-uaa**
  - ◊ Click **Next**
4. Configure Virtual Server Identifiers
  - ◊ Set the Port to **8443**
  - ◊ Click **Next**
5. Configure Server Pool and Rules
  - ◊ Click Create A New Server Pool
  - ◊ Set name to **tkgi-api-uaa**
  - ◊ Click **Next**
6. Configure SNAT Translation for the Server Pool.
  - ◊ Pool Members should be the same that you configured for the tkgi-api vritual server
  - ◊ Click **Next**
7. Configure Health Monitors:
  - ◊ Click Create A New Active Monitor
  - ◊ Set Name, for example **tkgi-api-uaa**
  - ◊ Set Health Check Protocol **LBHttpsMonitor**
  - ◊ Set port as **8443**
  - ◊ Leave other fields as default
  - ◊ Click **Next**
8. Configure Health Check Parameters:

- ◊ Choose High Security for SSL Ciphers
- ◊ Select TLS\_V1\_2 as SSL Protocols
- ◊ Set HTTP Request URL as /healthz
- ◊ Set HTTP Request Header as 200
- ◊ Click **Finish**

9. Configure a New Server Pool

- ◊ Set Active Health Monitor to tkgi-api-uaa
- ◊ Click **Finish**

10. Edit Virtual Server

- ◊ Set Default Server Pool as tkgi-api-uaa
- ◊ Click **Next**

11. Persistence Profiles is optional. Click **Next**.

12. Configure Client Side SSL (only if L7 Application Type is selected). Use the default certificates. Click **Next**.

13. Configure Server Side SSL (only if L7 Application Type is selected). Use the default certificates. Click **Finish**.

General Properties

Name \* pks-api-uaa

Description

Load Balancer Application Profile

Application Type \* Layer 7

Application Profile \* default-http-lb-app-profile

Access Log Enabled

CANCEL NEXT

Virtual Server Identifiers

IP Address \* 192.168.160.108

Port \* 8443

Protocol TCP

Advanced Properties

Maximum Concurrent Connection

Maximum New Connection Rate

Default Pool Member Port

CANCEL BACK NEXT

**Edit Virtual Server**

**Server Pool and Rules**

1 General Properties

2 Virtual Server Identifiers

**3 Server Pool and Rules**

4 Load Balancing Profiles

A Persistence Profiles

B Client Side SSL

C Server Side SSL

Default Server Pool **pks-api-server** Create A New Server Pool

Advanced Properties

Sorry Server Pool Create A New Server Pool

Load Balancer Rules

HTTP Request Rewrite Phase 0

+ ADD  CLONE  DELETE  MOVE UP  MOVE DOWN

| <input type="checkbox"/> Priority | Match Conditions | Match Strategy | Actions |
|-----------------------------------|------------------|----------------|---------|
| No Rules found.                   |                  |                |         |

HTTP Request Forwarding Phase 0

CANCEL BACK NEXT

**Add New Server Pool**

**General Properties**

SNAT Translation

Pool Members

Health Monitors

Name \* **pks-api-usa**

Description

Load Balancing Algorithm **ROUND\_ROBIN**

Advanced Properties

TCP Multiplexing  Disabled

Maximum Multiplexing Connections **6**

CANCEL NEXT

**Add New Server Pool**

**SNAT Translation**

Three Modes based on the topology are supported. In case of inline deployment of Load Balancer, use Transparent (NO\_SNAT) to preserve original Client IP and Port. Auto Map mode uses LB Interface IP and ephemeral port. In scenarios where both Clients and Pool Members are attached to the same Logical Router, SNAT (Auto Map or IP List) must be used.

Translation Mode \*  Transparent  Auto Map  IP List

CANCEL BACK NEXT

Add New Server Pool

- [General Properties](#)
- [SNAT Translation](#)
- [Pool Members](#)
- [Health Monitors](#)

### Pool Members

Pool Members can either be Static members that allows you to add IPs and Ports of individual servers or Dynamic Members as defined by NSGroup Membership Criteria. The admin state in case of the Dynamic Members can be set after Server Pool creation in the Members section of the Server Pool. Currently only IPv4 addressing is supported.

Membership Type  Static  Dynamic

Dynamic Membership

NSGroup \*

Max Group IP Address List

Limits the max number of pool members to the specified value. If not specified, allows the dynamic pool to grow up to the load balancer max pool member capacity.

NOTE: Pools using NSGroup must use Active Health Monitors with "Monitoring Port" specified

CANCEL BACK NEXT

Add New Server Pool

- [1 General Properties](#)
- [2 SNAT Translation](#)
- [3 Pool Members](#)
- [4 Health Monitors](#)

### Health Monitors

Minimum Active Members

Active Health Monitor  Create A New Active Monitor

Passive Health Monitor  Create A New Passive Monitor

CANCEL BACK FINISH

Add New Active Health Monitor

- [Monitor Properties](#)
- [Health Check Parameters](#)

### Monitor Properties

Name \*

Description

Health Check Protocol \*

Monitoring Port

Monitoring Interval (sec) \*

Fall Count \*

Rise Count \*

Timeout Period (sec) \*

CANCEL NEXT

**Add New Active Health Monitor**

**SSL and HTTP Health Check Parameters**

Configure the SSL Connection sent before the HTTP Request

**SSL Protocols**

| Available(2)                   | Selected(1)                                 |
|--------------------------------|---------------------------------------------|
| <input type="checkbox"/> SSLv3 | <input checked="" type="checkbox"/> TLSv1_2 |
| <input type="checkbox"/> TLSv1 |                                             |

**SSL Ciphers**

High Security ⓘ

**CANCEL** **BACK** **FINISH**

**Add New Active Health Monitor**

**SSL and HTTP Health Check Parameters**

**HTTP Request Configuration**

HTTP Method: GET

HTTP Request URL: /healthz

HTTP Request Version: HTTP\_VERSION\_1\_1

HTTP Request Headers:

| + ADD       | DELETE       |
|-------------|--------------|
| Header Name | Header Value |
|             |              |

HTTP Request Body:

**HTTP Response Configuration**

HTTP Response Code: 200

Specify response codes separated by comma (support up to 64 codes)

**CANCEL** **BACK** **FINISH**

**Add New Server Pool**

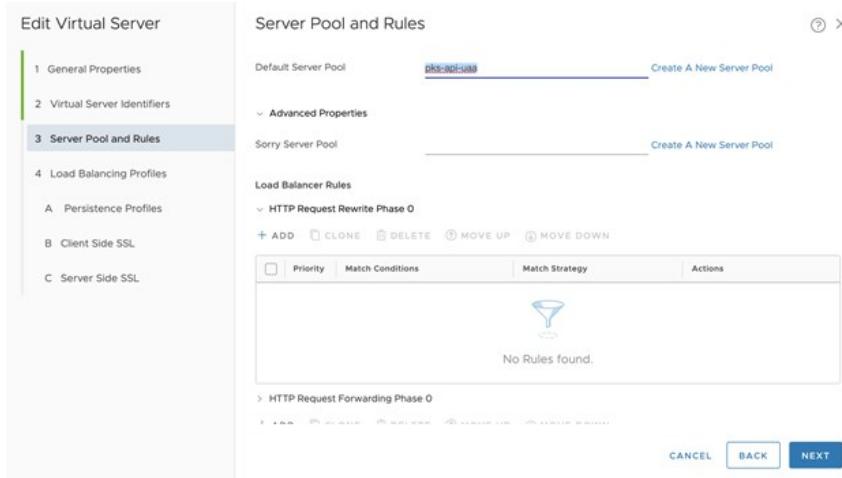
**Health Monitors**

Minimum Active Members: 1

Active Health Monitor: pks-api-uaa [Create A New Active Monitor](#)

Passive Health Monitor: [Create A New Passive Monitor](#)

**CANCEL** **BACK** **FINISH**



## Step 3: Create Load Balancer

1. In NSX-T Manager, select Networking > Load Balancing > Load Balancers.

2. Click **Add**.
3. Set the **Name**. For example `tkgi-api`.
4. Choose the Load Balancer Size. The default **SMALL** should be sufficient for most TKGI deployments. For large-scale deployments, use a larger size load balancer.

## Add Load Balancer



Name \*

pks-api

Description

Load Balancer Size \*

Select from one of the three available choices of size for the Load Balancer

| <input checked="" type="radio"/> SMALL | <input type="radio"/> MEDIUM | <input type="radio"/> LARGE |
|----------------------------------------|------------------------------|-----------------------------|
| Virtual Servers<br>20                  | Virtual Servers<br>100       | Virtual Servers<br>1000     |
| Pool Members<br>300                    | Pool Members<br>2000         | Pool Members<br>7500        |
| CPU 2                                  | CPU 4                        | CPU 12                      |
| Memory 4GB                             | Memory 8GB                   | Memory 16GB                 |

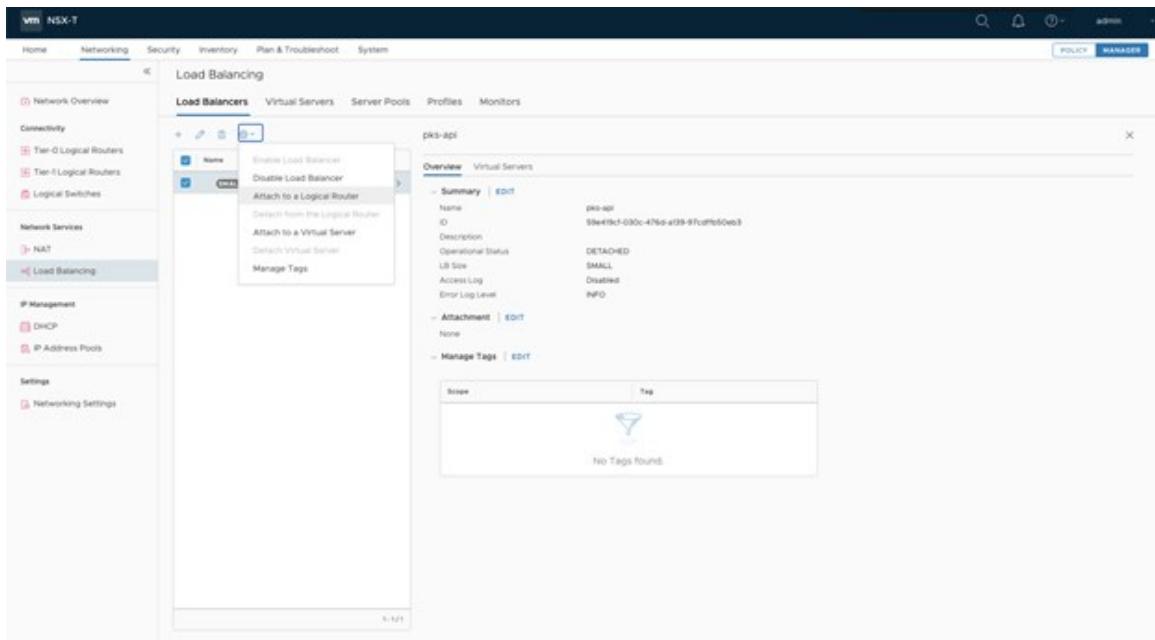
Error Log Level \*

INFO

5. Click OK.

## Step 4: Attach the Load Balancer to a Logical Router

1. In NSX-T Manager, select Networking > Load Balancing > Load Balancers.



2. Choose the `tkgs-api` load balancer you just created.
3. Click the gear icon and select **Attach to a Logical Router**.
4. Choose a Tier-1 logical router that is attached to TKGI API VMs.

## Attach to a Logical Router

×

Select the Router to which the Load Balancer `pks-api` is to be attached. Only Tier-1 Routers in Active Standby are currently supported. Note: The Load Balancer can only be Enabled if it had a Virtual Server associated with it.

**Tier-1 Logical Router \***

`deployment-router`

`deployment-router`

ID:`b0a6...37ca`

**CANCEL**

**OK**

5. Click **OK**.

## Troubleshooting LB-Router Attachment

If your logical router does not have an associated edge cluster, you will see an error similar to the following:

X

## Attach to a Logical Router

**!** The logical router LogicalRouter/b0a65a53-8004-4d2b-8551-6bc29abc37ca does not have associated edge cluster to deploy a load balancer service LoadBalancerService/c0c478b2-67f2-404d-b109-d72e5e3a5672.

Select the Router to which the Load Balancer pks-api is to be attached. Only Tier-1 Routers in Active Standby are currently supported. Note: The Load Balancer can only be Enabled if it had a Virtual Server associated with it.

|                                                                                                                                                                                                                               |                                                |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------|
| <b>Tier-1 Logical Router *</b>                                                                                                                                                                                                | <input type="text" value="deployment-router"/> |
| <span style="border: 1px solid #ccc; padding: 2px 10px; margin-right: 10px;">CANCEL</span> <span style="background-color: #0072bc; color: white; border: 1px solid #0072bc; padding: 2px 10px; border-radius: 5px;">OK</span> |                                                |

When this occurs, you must create a Logical Router that is associated with the Edge Cluster.

To create and configure a new Tier-1 router:

1. Select **Networking > Tier-1 Logical Routers**.

The screenshot shows the VMware NSX-T Management interface under the Networking tab. On the left, there's a navigation sidebar with options like Home, Networking (which is selected), Security, Inventory, Plan & Troubleshoot, and System. Under Networking, there are sections for Connectivity (Tier-0 Logical Routers, Tier-1 Logical Routers, Logical Switches), Network Services (NAT, Load Balancing), IP Management (DHCP, IP Address Pools), and Settings (Networking Settings). The main content area is titled 'Tier-1 Logical Routers'. It has a table with columns: Logical Router, ID, Connected Tier-0 Router, High Availability Mode, Transport Zone, and Edge Cluster. One row is selected, showing 'deployment-router' as the name, '59a2-2100' as the ID, 'ID-shared' as the Connected Tier-0 Router, 'Active-Standby' as the High Availability Mode, 'T1-Overlay' as the Transport Zone, and 'Edge Cluster' as the Edge Cluster. At the bottom of the table, there are buttons for ADD, EDIT, DELETE, and ACTIONS. A search bar is also present at the top of the table area.

2. Click **Add**.
3. Configure Tier-1 Router:
  - Set the **Name**. For example; `tkgi-api`.

- Set **Tier-0 Router**.
- Set **Edge Cluster**.
- Set **Edge Cluster member**.
- Click **Add**.

## New Tier-1 Router



Tier-1 Router   Advanced

Name\* pks-api

Description

Tier-0 Router t0-shared



Edge Cluster edge-cluster-0



StandBy Relocation  Disable

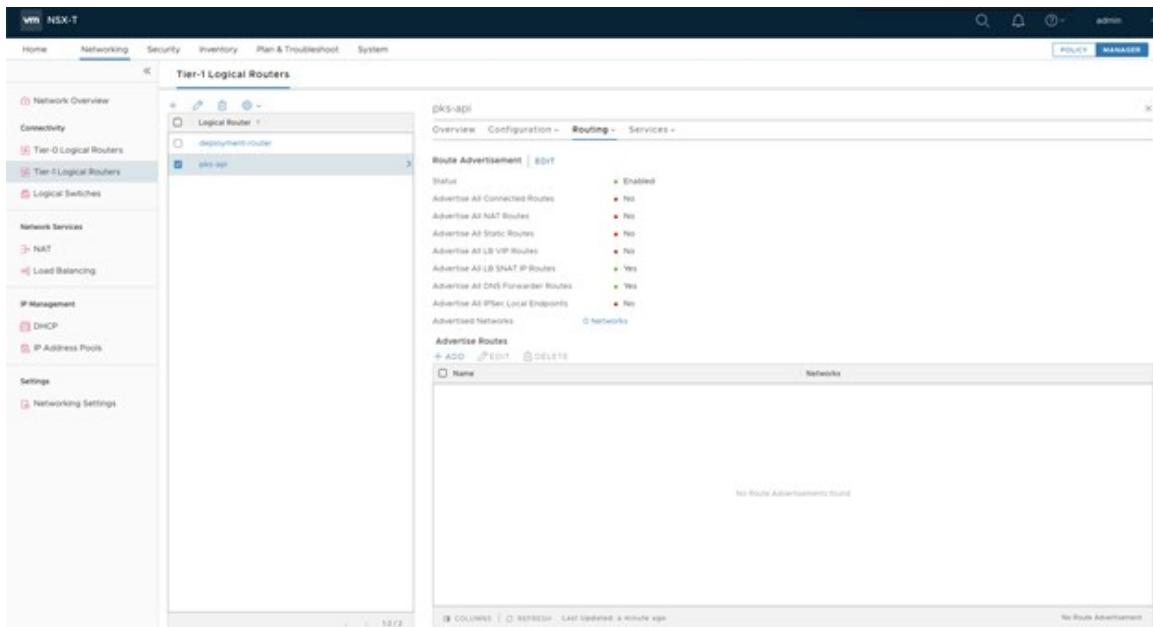
Failover Mode  Preemptive  Non-Preemptive

Edge Cluster Members [i](#) tn-cluster-0-edge-0 [x](#)

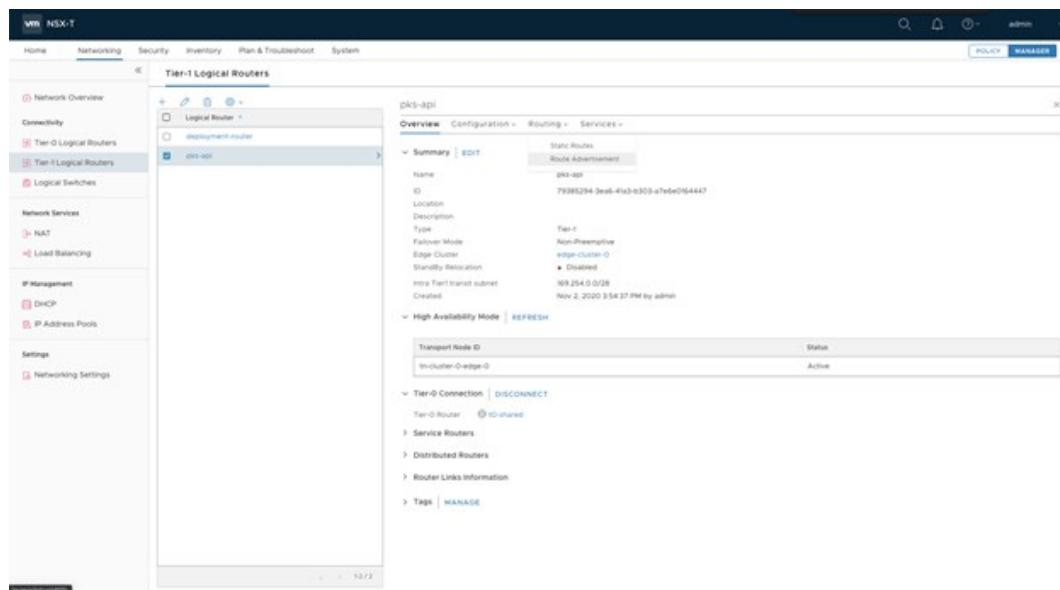
CANCEL

ADD

4. Configure **Route Advertisement** for the Tier-1 Router.



- ◊ Select the Tier-1 Router.
- ◊ Select the **Routing** tab.
- ◊ Select **Route Advertisement > Edit**.



- ◊ Enable Route Advertisement for all load balancer VIP routes for the Tier-1 Router:
  - **Status:** Enabled.
  - **Advertise all LB VIP routes:** Yes.
  - **Advertise all LB SNAT IP routes:** Yes.
  - Click **Save**.

Edit Route Advertisement Configuration X

|                                    |                                             |
|------------------------------------|---------------------------------------------|
| Status                             | <input checked="" type="checkbox"/> Enabled |
| Advertise All Connected Routes     | <input type="checkbox"/> No                 |
| Advertise All NAT Routes           | <input type="checkbox"/> No                 |
| Advertise All Static Routes        | <input type="checkbox"/> No                 |
| Advertise All LB VIP Routes        | <input checked="" type="checkbox"/> Yes     |
| Advertise All LB SNAT IP Routes    | <input checked="" type="checkbox"/> Yes     |
| Advertise All DNS Forwarder Routes | <input type="checkbox"/> No                 |

CANCEL SAVE

5. Attach the Logical Router to Load Balancer.

6. Click **OK** to complete the operation.

## Attach to a Logical Router X

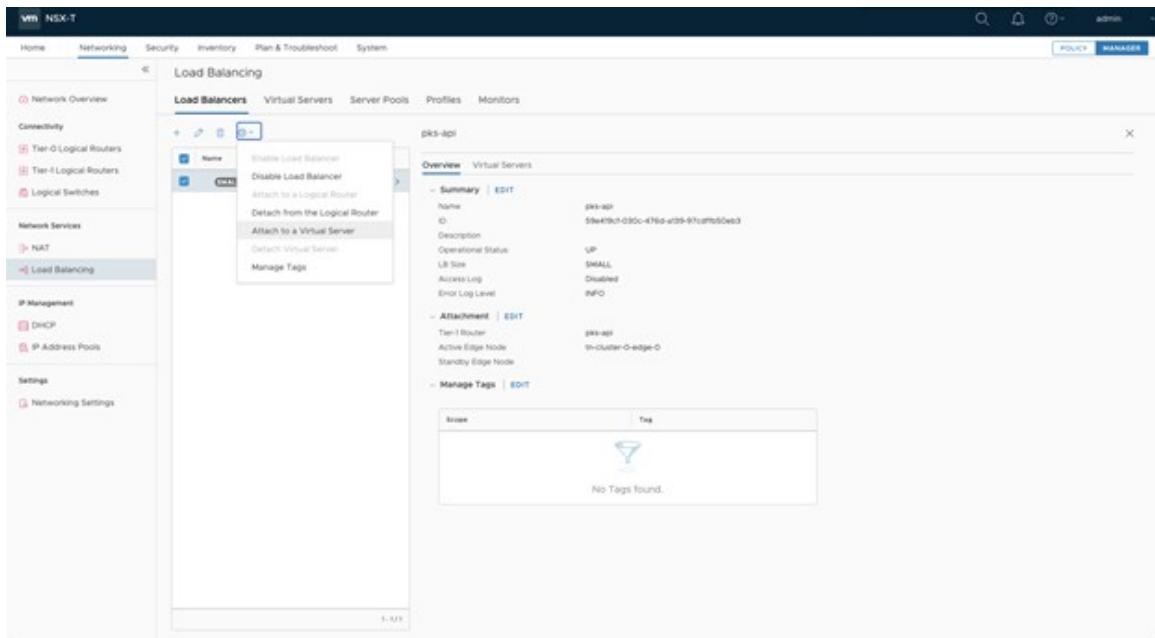
Select the Router to which the Load Balancer pks-api is to be attached. Only Tier-1 Routers in Active Standby are currently supported. Note: The Load Balancer can only be Enabled if it had a Virtual Server associated with it.

Tier-1 Logical Router \* pks-api

CANCEL OK

## Step 5: Attach the Load Balancer to the Virtual Servers

1. Select the Load Balancer.



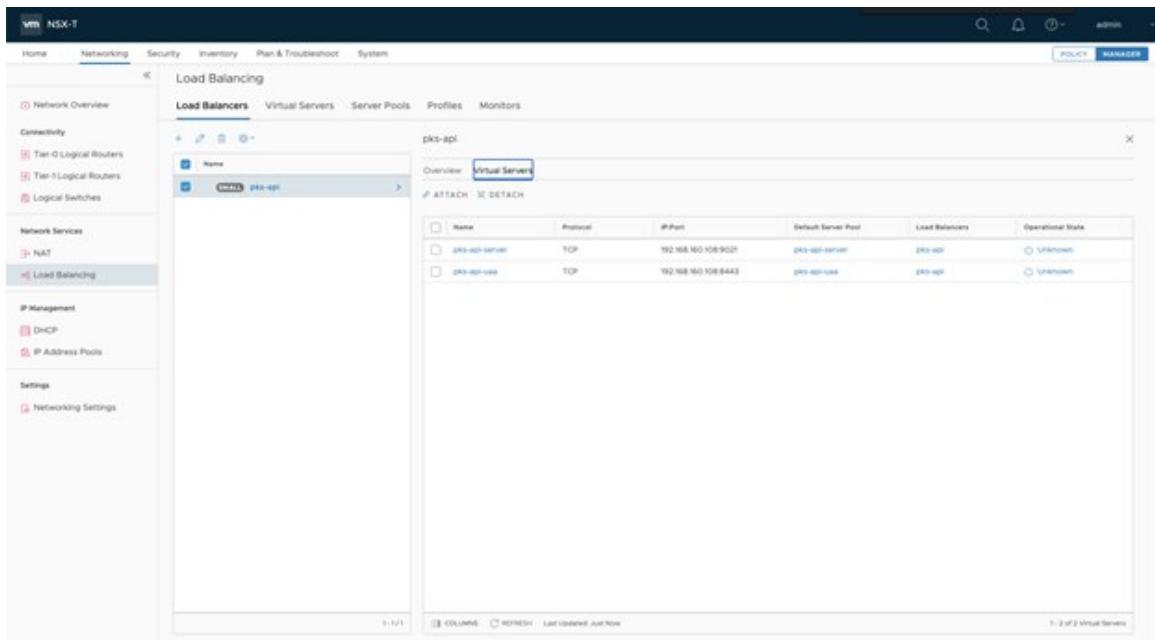
2. Click the **Settings** icon.
3. Select **Attach to a Virtual Server**.



4. Attach the load balancer to the `tkgi-api-server` virtual server. You should see it inside the **Virtual Servers** tab.
5. Click **Ok**.
6. Click **Settings**.
7. Select **Attach to a Virtual Server**.



8. Attach to the `tkgi-api-uua` virtual server. You should see it inside the **Virtual Servers** tab.
9. Click **Ok**.



## Step 6: Configure TKGI to Use the Load Balancer

Skip this if you deployed as [No-NAT with Virtual Switch \(VSS/VDS\) Topology](#)

Now that the load balancer for the TKGI API control plane is configured, update the TKGI tile to point to the load balancer.

1. Log in to Ops Manager.
2. Go to [Tanzu Kubernetes Grid Integrated Edition Tile Resource Config](#).
3. Click **TKGI API**. You will see a drop down for **TKGI API config**.
4. Change the **TKGI API Instances Number** to **2** or **3**. We recommend **3** for quorum.
5. Set the **NSGroup** if you configured **Dynamic Server Pool**. Otherwise leave it empty.
6. Set **VIF Type** to **PARENT** or leave it empty.
7. Set the **Logical Load Balancer** as follows:

```
{
 "server_pools": [
 {
 "name": "tkgi-api-server",
 "port": 9021
 },
 {
 "name": "tkgi-api-uaa",
 "port": 8443
 }
]
}
```

8. Click **Save**.
9. Click **apply-changes** and wait for Ops Manager to finish saving the settings.
  - For static server pools, this operation will add the VM as server pool member.

- For dynamic server pools, this operation will add the VM to the corresponding NSGroup.

## Step 7: Test the Load Balancer

To validate your Load Balancer configuration:

- In **NSX-T Manager**, verify that the operational status of the load balancer is up.

- Make sure the Virtual Servers are up.

| Name                                                | Protocol | IP:Port              | Default Server Pool                                  | Load Balancers | Operational State |
|-----------------------------------------------------|----------|----------------------|------------------------------------------------------|----------------|-------------------|
| lb-pks-33988550-9f3e-456e-a75e-28658fe5108a-lb-pool | TCP      | 192.168.160.100:8443 | lb-pks-33988550-9f3e-456e-a75e-28658fe5108a-456...   | ↑ Up           |                   |
| pks-33988550-9f3e-456e-a75e-28658fe5108a-pks-pool   | TCP      | 192.168.160.107:80   | pks-33988550-9f3e-456e-a75e-28658fe5108a-9f3e-456... | ↑ Up           |                   |
| pks-33988550-9f3e-456e-a75e-28658fe5108a-pks-pool   | TCP      | 192.168.160.101:80   |                                                      | ↑ Up           |                   |
| pks-33988550-9f3e-456e-a75e-28658fe5108a-pks-pool   | TCP      | 192.168.160.101:443  | lb-pks-33988550-9f3e-456...                          | ↑ Up           |                   |
| pks-api-server                                      | TCP      | 192.168.160.108:9021 | pks-api-server                                       | pks-api        | ↑ Up              |
| pks-api-usa                                         | TCP      | 192.168.160.108:8443 | pks-api-usa                                          | pks-api        | ↑ Up              |

3. Make sure the Server Pools are up.

| Name                                                      | LB Algorithm | Membership Type | Members/N3Group                                    | Virtual Servers            | Operational State |
|-----------------------------------------------------------|--------------|-----------------|----------------------------------------------------|----------------------------|-------------------|
| lb-pks-33988550-9f3e-456e-a75e-28658fe5108a-lb-pool       | Round Robin  | IPv4 - Dynamic  | lb-pks-33988550-9f3e-456e-a75e-28658fe5108a-456... | ↑ Up                       |                   |
| pks-33988550-9f3e-456e-a75e-28658fe5108a-default-nginx-80 | Round Robin  | IPv4 - Static   | 3                                                  | pks-33988550-9f3e-456e-... | ↑ Up              |
| pks-api-server                                            | Round Robin  | IPv4 - Dynamic  | pks-api                                            | pks-api-server             | ↑ Up              |
| pks-api-usa                                               | Round Robin  | IPv4 - Dynamic  | pks-api                                            | pks-api-usa                | ↑ Up              |

4. To test the load balancer:

1. Using your TKGI client jump host, change the TKGI API hostname to resolve to the Load Balancer IP.

For example, you can use `192.168.160.108` as the IP address of the load balancer:

```
kubo@jumper:~$ cat /etc/hosts
127.0.0.1 localhost
127.0.1.1 jumper.localdomain jumper

The following lines are desirable for IPv6 capable hosts
::1 localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
192.168.111.93 vxlan-vm-111-93.vmware.com
```

```
192.168.111.109 nsxmanager.tkgi.vmware.local
192.168.160.108 tkgi.tkgi-api.cf-app.com
```

2. Log in to the TKGI API Server via the load balancer.

For example:

```
kubo@jumper:~$ tkgi login -a tkgi.tkgi-api.cf-app.com -u lana -p password
-k && tkgi clusters

API Endpoint: tkgi.tkgi-api.cf-app.com
User: lana
Login successful.

TKGI Version Name k8s Version Plan Name UUID
 Status Action
1.12.0-build.1 test_one 1.21.3 Plan 1 33988550-...-2865
8fe51d8a succeeded UPDATE
```

## Provisioning a Load Balancer for the VMware NSX Management Cluster

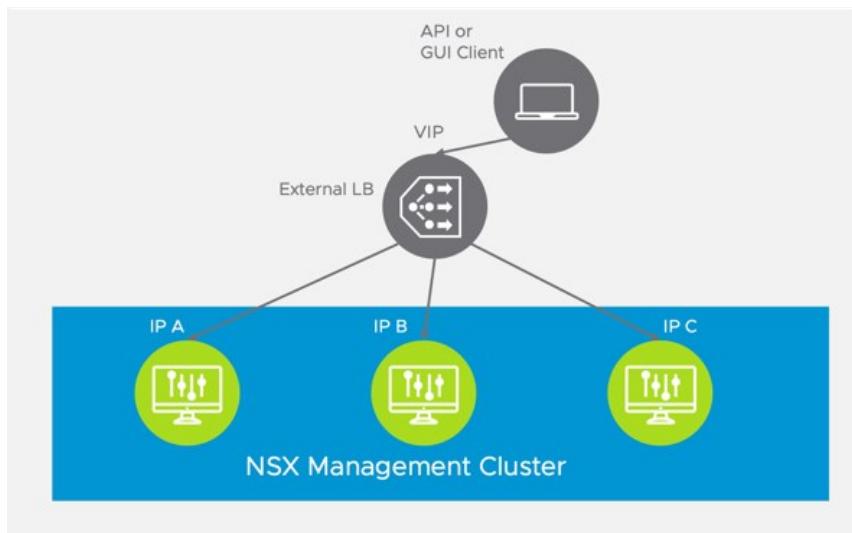
This topic describes how to deploy a load balancer for the NSX-T Management Cluster for Tanzu Kubernetes Grid Integrated Edition.

### About the NSX-T Management Cluster

NSX-T provides a converged management and control plane that is referred to as the **NSX-T Management Cluster**. The architecture delivers high availability of the NSX-T Manager node, reduces the likelihood of operation failures of NSX-T, and provides API and UI clients with multiple endpoints or a single VIP for high availability.

While using a VIP to access the NSX-T Management layer provides high-availability, it does not balance the workload. To avoid overloading a single NSX-T Manager, which might be the case when [HA VIP addressing](#) is used, an NSX-T load balancer can be provisioned to allow NCP and other components orchestrated by Tanzu Kubernetes Grid Integrated Edition to distribute load efficiently among NSX-T Manager nodes.

The diagram below shows an external load balancer fronting the NSX-T Manager nodes. The load balancer is deployed within the NSX-T environment and intercepts requests to the NSX-T Management Cluster. The load balancer selects one of the NSX-T Manager nodes to handle the request and rewrites the destination IP address to reflect the selection.



**Note:** The load balancer VIP load balances traffic to all NSX-T Manager instances in round robin fashion. A Cluster HA VIP, on the other hand, only sends traffic one of the NSX-T Manager instances that is mapped to the Cluster IP VIP; the other NSX-T Manager instances do not receive any traffic.

For scalability, deploy a load balancer in front of the NSX-T Manager nodes. When provisioning the load balancer, you configure a virtual server on the load balancer, and associate a virtual IP address with the virtual server. This load balancer VIP can be used as the entry-point for TKGI- and NCP-related API requests on the NSX-T Control Plane. The virtual server includes a member pool where all NSX-T Management Cluster nodes belong. Additionally, health monitoring is enabled for the member pool to quickly and efficiently address potential node failures detected among the NSX-T Management Cluster.

## Provision the NSX-T Load Balancer for the Management Cluster

To provision the load balancer for the NSX-T Management Cluster, complete the following steps.

### Step 1: Log in to an NSX-T Manager Node

1. Log in to an NSX-T Manager Node.



**Note:** You can connect to any NSX-T Manager Node in the management cluster to provision the load balancer.

2. Select the **Advanced Networking & Security** tab.



**Note:** You must use the **Advanced Networking and Security** tab in NSX-T Manager to create, read, update, and delete all NSX-T networking objects used for Tanzu Kubernetes Grid Integrated Edition.

### Step 2: Configure a Logical Switch

Add and configure a new logical switch for the load balancer.

- Select **Switching**.
- Click **Add**.
- Configure the logical switch:
  - **Name:** Enter a name for the logical switch, such as `LS-NSX-T-EXTERNAL-LB`.
  - **Transport Zone:** Select the overlay transport zone, such as `TZ-Overlay`.
- Click **Add**.

## Step 3: Configure a Tier-1 Logical Router

Configure a new Tier-1 Router. Create the Tier-1 Router on the same Edge Cluster where the Tier-0 Router that provides external connectivity to vCenter and NSX-T Manager is located.

- Select **Routers**.
- Click **Add > Tier-1 Router**.
- Configure the new Tier-1 Router and click **Add**.
  - **Name:** `T1-NSX-T-EXTERNAL-LB`, for example.
  - **Tier-0 Router:** Connect the Tier-1 Router to the Tier-0 Router, for example `Shared-T0`.
  - **Edge Cluster:** Select the same Edge Cluster where the Tier-0 Router is located, such as `edgecluster1`.
  - **Edge Cluster Members:** Select `nsx-edge-1-tn` and `nsx-edge-2-tn`, for example.

## Step 4: Advertise the Routes

Configure Route Advertisement for the Tier-1 Router.

- Select the Tier-1 Router.
- Select the **Routing** tab.
- Select **Route Advertisement > Edit**.
- Enable Route Advertisement for all load balancer VIP routes for the Tier-1 Router:
  - **Status:** `Enabled`.
  - **Advertise all LB VIP routes:** `Yes`.
  - **Advertise all LB SNAT IP routes:** `Yes`.
  - Click **Save**

## Step 5: Verify Router and Switch Configuration

Verify successful creation and configuration of the logical switch and router.

- Select the Tier-1 Router.
- Select the **Configuration** tab and the **Ports** option.

- Verify that the router has a single linked port connecting the Tier-1 Router to the Tier-0 Router.

## Step 6: Configure a Small Load Balancer

Create a new small-size Load Balancer and attach it to the Tier1 router previously created.



**Note:** The small-size load balancer is suitable for the NSX-T Management Cluster load balancer. Make sure you have enough Edge Cluster resources to provision a small load balancer.

- Select **Load Balancers**.
- Click **Add**.
- Enter a **Name** for the load balancer.
- Select the **SMALL** size load balancer.
- Click **OK**.

## Step 7: Attach the Load Balancer to the Router

Attach the load balancer to the Tier-1 Router previously created.

- Select the load balancer you just provisioned.
- Select the **Overview** tab.
- Select **Attachment > Edit**.
- Tier-1 Logical Router:** Enter the name of the Tier-1 Router you configured, for example `T1-NSX-T-EXTERNAL-LB`.
- Click **OK**.

## Step 8: Configure a Virtual Server

Add and configure a virtual server for the load balancer.

- Select **Load Balancers > Virtual Servers**.
- Click **Add**.

Configure **General Properties** for the virtual server:

- Name:** `VS-NSX-T-EXTERNAL-LB`.
- Application Types:** `Layer 4 TCP`.
- Application Profile:** `default-tcp-lb-app-profile`.
- Access Log:** `Deactivated`.
- Click **Next**

Configure **Virtual Server Identifiers** for the virtual server:

- IP Address:** Enter an IP address from the floating pool, such as `10.40.14.250`.

- **Port:** 443.
- Click **Next**.

Configure **Virtual Server Pool** for the virtual server:

- Click **Create a New Server Pool**.

Configure **General Properties** for the server pool:

- **Name:** For example `NSX-T-MGRS-SRV-POOL`.
- **Load Balancing Algorithm:** `ROUND_ROBIN`.
- Click **Next**

Configure **SNAT Translation** for the server pool:

- **Translation Mode:** IP List
- **IP address:** Enter the Virtual Server IP (VIP) address here, for example `10.40.14.250`.
- Click **Next**.

Configure **Pool Members** for the server pool:

- **Membership Type:** `Static`.
- **Static Membership:** Add all 3 NSX-T Managers as members by entering the node name, IP address, and port (443) for each node.
- Click **Next**.

Configure **Health Monitors**:

- We will create the Health Monitors separately.
- Click **Finish**.

Back at the Server Pool screen, click **Next**.

Configure **Load Balancing Profiles** for the load balancer:

- **Persistence Profile > Source IP:** Select `default-source-ip-lb-persistence-profile`.
- Click **Finish**.



**Note:** If a proxy is used between the NSX-T Management Cluster and the TKGI Management Plane, do not configure a persistence profile.

## Step 9: Attach the Virtual Server to the Load Balancer

Attach the virtual switch to the NSX-T load balancer.

- In the **Load Balancing** panel, select the Virtual Server you created.
- Select the **Load Balancers** tab.
- Click **Attach**.
- **Load Balancer:** Select the load balancer to attach, such as `NSX-T-EXTERNAL-LB`.
- Click **OK**.

## Step 10: Verify the Load Balancer.

Once the load balancer is configured, verify it by doing the following:

- Ping the NSX-T load balancer VIP address from your local machine.
- Access the NSX-T Manager interface using the load balancer VIP address, for example <https://10.40.14.250>.



**Note:** The URL redirects to the same NSX-T Manager. Persistence is done on the source IP based on the persistence profile you selected.

## Step 11: Create an Active Health Monitor (HM)

Create a new Active Health Monitor (HM) for NSX-T Management Cluster members using the NSX-T Health Check protocol.

- Select **Load Balancers > Server Pools**.
- Select the server pool you created. For example, [NSX-T-MGRS-SRV-POOL](#).
- Select the **Overview** tab.
- Click **Health Monitor > Edit**.
- Click **Create a new active monitor**.

Configure **Monitor Properties**:

- **Name:** [NSX-T-Mgr-Health-Monitor](#).
- **Health Check Protocol:** [LbHttpsMonitor](#).
- **Monitoring Port:** [443](#).

Configure **Health Check Parameters**.

Configure the new Active HM with specific HTTP request fields as follows:

- **SSL Protocols:** Select the **TLS\_v1** and **TLS\_v2** protocols.
- **SSL Ciphers:** Select **Balanced** (recommended).

Configure the **HTTP Request Configuration** settings for the health monitor:

- **HTTP Method:** [GET](#)
- **HTTP Request URL:** [/api/v1/reverse-proxy/node/health](#)
- **HTTP Request Version:** [HTTP\\_VERSION\\_1\\_1](#)

Configure the **HTTP Request Headers** for the health monitor:

- **Authorization:** [Basic YWRtaW46VkJ1YXJlMSE=](#), which is the base64-encoded value of the NSX-T administrator credentials.
- **Content-Type:** [application/json](#).
- **Accept:** [application/json](#).



**Note:** In the example, `YWRtaW46Vk13YXJIMSE=` is the base64-encoded value of the NSX-T administrator credentials, expressed in the form `admin-user:password`. You can use the free online service [www.base64encode.org](http://www.base64encode.org) to base64 encode your NSX-T administrator credentials.

Configure the **HTTP Response Configuration** for the health monitor:

- **HTTP Response Code:** `200`.
- Click **Finish**.

At the Health Monitors screen, specify the Active Health Monitor you just created:

- **Active Health Monitor:** Enter a name for the health monitor, such as `NSX-T-Mgr-Health-Monitor`.
- Click **Finish**.

## Step 12: Create SNAT Rule

If your Tanzu Kubernetes Grid Integrated Edition deployment uses NAT mode, make sure Health Monitoring traffic is correctly SNAT-translated when leaving the NSX-T topology. Add a specific SNAT rule that intercepts HM traffic generated by the load balancer and translates this to a globally-routable IP Address allocated using the same principle of the load balancer VIP. The following screenshot illustrates an example of SNAT rule added to the Tier0 Router to enable HM SNAT translation. In the example, `100.64.128.0/31` is the subnet for the Load Balancer Tier-1 uplink interface.

To do this you need to retrieve the IP of the T1 uplink (Tier-1 Router that connected the NSX-T LB instance). In the example below, the T1 uplink IP is `100.64.112.37/31`.

Create the following SNAT rule on the Tier-0 Router:

- **Priority:** `2000`.
- **Action:** `SNAT`.
- **Source IP:** `100.64.112.36/31`, for example.
- **Destination IP:** `10.40.206.0/25`, for example.
- **Translated IP:** `10.40.14.251`, for example.
- Click **Save**
- Verify configuration of the SNAT rule and server pool health:

## Step 13: Verify that NSX-T Manager Traffic Is Load Balanced

Verify the load balancer and that traffic is load balanced.

- Confirm that the status of the Logical Switch for the load balancer is Up.
- Confirm that the status of the Virtual Server for the load balancer is Up.
- Confirm that the status of the Server Pool is Up.
- Open an HTTPS session using multiple browser clients.

- Confirm that traffic is load-balanced across different NSX-T Managers:
  - You can use the NSX-T API to validate that secure HTTP requests against the new VIP address are associated with the load balancer's Virtual Server. Relying on the SuperUser Principal Identity created as part of TKGI provisioning steps, you can cURL the NSX-T Management Cluster using the standard HA-VIP address or the newly-provisioned virtual server VIP. For example:
    - Before load balancer provisioning is completed:

```
curl -k -X GET "https://192.168.6.210/api/v1/trust-management/principal-identities" --cert $(pwd)/pks-nsx-t-superuser.crt --key $(pwd)/pks-nsx-t-superuser.key
```

  - After load balancer provisioning is completed:

```
curl -k -X GET "https://91.0.0.1/api/v1/trust-management/principal-identities" --cert $(pwd)/pks-nsx-t-superuser.crt --key $(pwd)/pks-nsx-t-superuser.key
```
- The Key behavioral differences among the two API calls:
  - The call toward the Virtual Server VIP effectively balances load requests among the NSX-T Server Pool members.
  - The call made toward the HA VIP address ALWAYS selects the same member, the Active Member, of the NSX-T Management Cluster.
- Residual configuration steps:
  - Change TKGI Tile configuration for NSX-T Manager IP Address to use the newly-provisioned Virtual IP Address. This configuration enables any component internal to TKGI, for example NCP, NSX OSB Proxy, BOSH CPI, etc., to use the new Load Balancer functionality.

## Using Proxies with Tanzu Kubernetes Grid Integrated Edition on VMware NSX

This topic describes how HTTP/HTTPS proxies work in Tanzu Kubernetes Grid Integrated Edition (TKGI) with NSX-T, and how to set proxies globally.

To configure proxy settings specifically for individual TKGI clusters, see [Configure Cluster Proxies](#).

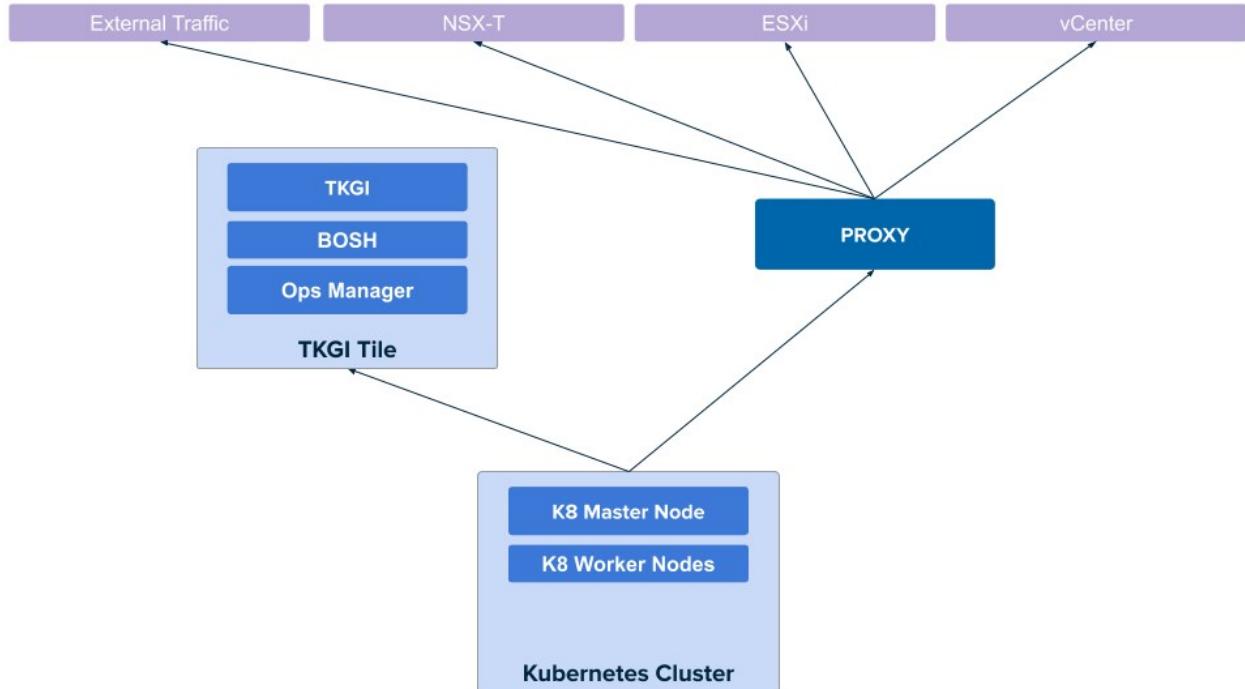
## Overview

If your environment includes HTTP proxies, you can configure Tanzu Kubernetes Grid Integrated Edition with NSX-T to use these proxies so that Tanzu Kubernetes Grid Integrated Edition-deployed Kubernetes control plane and worker nodes access public Internet services and other internal services through a proxy.

In addition, Tanzu Kubernetes Grid Integrated Edition proxy settings apply to the TKGI API instance. When an Tanzu Kubernetes Grid Integrated Edition operator creates a Kubernetes cluster, the TKGI API VM behind a proxy is able to manage NSX-T objects on the standard network.

You can also proxy outgoing HTTP/HTTPS traffic from Ops Manager and the BOSH Director so that all Tanzu Kubernetes Grid Integrated Edition components use the same proxy service.

The following diagram illustrates the network architecture:



## Enable TKGI API and Kubernetes Proxy

To configure a global HTTP proxy for all outgoing HTTP/HTTPS traffic from the Kubernetes cluster nodes and the TKGI API server, perform the following steps:

1. Navigate to Ops Manager and log in.
2. Click the **Tanzu Kubernetes Grid Integrated Edition** tile.
3. Click **Networking**.
4. Under **HTTP/HTTPS proxy**, select **Enabled**. When this option is enabled, you can proxy HTTP traffic, HTTPS traffic, or both.

**HTTP/HTTPS Proxy (for vSphere and AWS only)\***

Disabled  
 Enabled

**HTTP Proxy URL**

**HTTP Proxy Credentials**

Username  
 Password

**HTTPS Proxy URL**

**HTTPS Proxy Credentials**

Username  
 Password

**No Proxy**

5. (Optional) Configure Tanzu Kubernetes Grid Integrated Edition to use a proxy.

Production environments can deny direct access to public Internet services and between internal services by placing an HTTP or HTTPS proxy in the network path between Kubernetes nodes and those services.

Configure Tanzu Kubernetes Grid Integrated Edition to use your proxy and activate the following:

- TKGI API access to public Internet services and other internal services.
- Tanzu Kubernetes Grid Integrated Edition-deployed Kubernetes nodes access to public Internet services and other internal services.
- Tanzu Kubernetes Grid Integrated Edition Telemetry ability to forward Telemetry data to the CEIP and Telemetry program.



**Note:** This setting does not set the proxy for running Kubernetes workloads or pods.

6. To complete your global proxy configuration for all outgoing HTTP/HTTPS traffic from your

Kubernetes clusters, perform the following steps:

1. To proxy outgoing HTTP traffic, enter the URL of your HTTP proxy endpoint under **HTTP Proxy URL**. For example, `http\://myproxy.com:1234`.
2. (Optional) If your outgoing HTTP proxy uses basic authentication, enter the username and password in the **HTTP Proxy Credentials** fields.
3. To proxy outgoing HTTPS traffic, enter the URL of your HTTP proxy endpoint under **HTTPS Proxy URL**. For example, `http\://myproxy.com:1234`.



**Note:** Using an HTTPS connection to the proxy server is not supported. HTTP and HTTPS proxy options can only be configured with an HTTP connection to the proxy server. You cannot populate either of the proxy URL fields with an HTTPS URL. The proxy host and port can be different for HTTP and HTTPS traffic, but the proxy protocol must be HTTP.

4. (Optional) If your HTTPS proxy uses basic authentication, enter the username and password in the **HTTPS Proxy Credentials** fields.
5. Under **No Proxy**, enter the comma-separated list of IP addresses that must bypass the proxy to allow for internal Tanzu Kubernetes Grid Integrated Edition communication.

The **No Proxy** list should include `127.0.0.1` and `localhost`.

Also include the following in the **No Proxy** list:

- Your Tanzu Kubernetes Grid Integrated Edition environment's CIDRs, such as the service network CIDR where your Tanzu Kubernetes Grid Integrated Edition cluster is deployed, the deployment network CIDR, the node network IP block CIDR, and the pod network IP block CIDR.
- The FQDN of any registry, such as the Harbor API FQDN, or component communicating with Tanzu Kubernetes Grid Integrated Edition, using a hostname instead of an IP address.
- The IP addresses for your NSX Manager, vCenter Server, and all ESXi hosts, if you are upgrading and have an existing proxy configuration for reaching a Docker registry or other external services.
- Any additional IP addresses or domain names that should bypass the proxy.

The **No Proxy** property for vSphere accepts wildcard domains denoted by a prefixed `\*` or `..`

For example:

127.0.0.1, localhost, \*.example1.com, .example2.com, example3.com,  
198.51.100.0/24, 203.0.113.0/24, 192.0.2.0/24



**Note:** By default the `10.100.0.0/8` and `10.200.0.0/8` IP address ranges, `.internal`, `.svc`, `.cluster.local`, `.svc.cluster`, and your Tanzu Kubernetes Grid Integrated Edition FQDN are not proxied. This allows internal Tanzu Kubernetes Grid Integrated Edition communication.

Do not use the `_` character in the **No Proxy** field. Entering an underscore character in this field can cause upgrades to fail.

Because some jobs in the VMs accept `\*` as a wildcard, while others only accept `.`, we recommend that you define a wildcard domain using both of them. For example, to denote `example.com` as a wildcard domain, add both `\*.example.com` and `example.com` to the **No Proxy** property.

7. Save the changes to the Tanzu Kubernetes Grid Integrated Edition tile.
8. Proceed with any remaining Tanzu Kubernetes Grid Integrated Edition tile configurations and deploy Tanzu Kubernetes Grid Integrated Edition. See [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T](#).

## Enable Ops Manager and BOSH Proxy

To enable an HTTP proxy for outgoing HTTP/HTTPS traffic from Ops Manager and the BOSH Director, perform the following steps:

1. Log in to Ops Manager.
2. Select **User Name > Settings** in the upper right.
3. Click **Proxy Settings**.
4. Under **HTTP Proxy**, enter the FQDN or IP address of the HTTP proxy endpoint. For example, `http://myproxy.com:80`.
5. Under **HTTPS Proxy**, enter the FQDN or IP address of the HTTPS proxy endpoint. For example, `http://myproxy.com:80`.



**Note:** Using an HTTPS connection to the proxy server is not supported. Ops Manager and BOSH HTTP and HTTPS proxy options can be only configured with an HTTP connection to the proxy.

6. Under **No Proxy**, include the hosts that must bypass the proxy. This is required.

In addition to `127.0.0.1` and `localhost`, include the BOSH Director IP, Ops Manager IP, TKGI API VM IP, and the TKGI Database VM IP. If the TKGI Database is in HA mode (beta), enter all your database IPs in the **No Proxy** field.

127.0.0.1, localhost, BOSH-DIRECTOR-IP, TKGI-API-IP, OPS-MANAGER-IP, TKGI-DATABASE-IP



**Note:** Ops Manager does not allow the use of a CIDR range in the **No Proxy** field. You must specify each individual IP address to bypass the proxy.

The **No Proxy** field does not accept wildcard domain notation, such as `.docker.io` and `.docker.com`. You must specify the exact IP or FQDN to bypass the proxy, such as `registry-1.docker.io`.

7. Click **Save**.
8. Return to the Ops Manager Installation Dashboard and click **Review Pending Changes**.
9. Click **Apply Changes** to deploy Ops Manager and the BOSH Director with the updated proxy settings.

## Isolating Tenants

This topic describes how to isolate tenants in VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) multi-tenant environments.

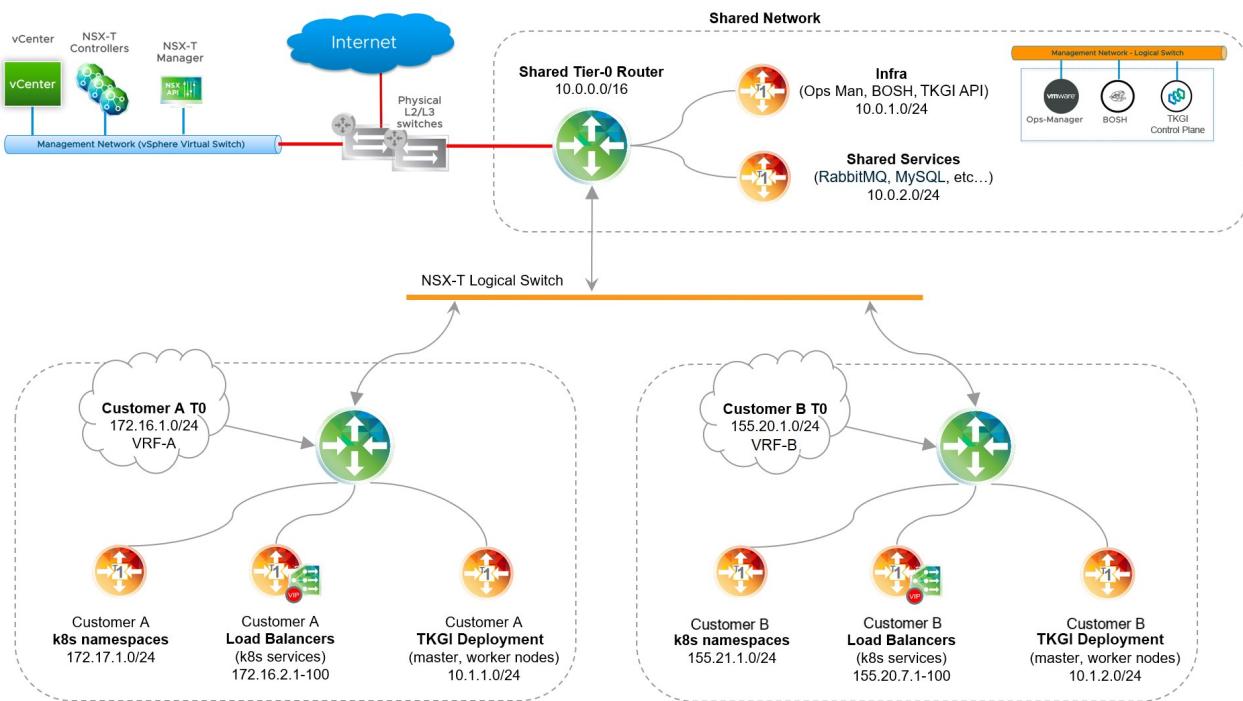
## About Tenant Isolation

You can isolate a cluster and its workloads using NSX-T Tier-0 (T0) logical routers or VRF Tier-0 gateways:

- [Using a Multi-T0 Router Configuration for Tenant Isolation](#)
- [Using a VRF Tier-0 Gateway Configuration for Tenant Isolation](#)

## Using a Multi-T0 Router Configuration for Tenant Isolation

Tanzu Kubernetes Grid Integrated Edition multi-T0 lets you provision, manage, and secure Kubernetes cluster deployments on isolated tenant networks. As shown in the diagram below, instead of having a single T0 router, there are multiple T0 routers. The Shared Tier-0 router handles traffic between the TKGI management network and the vSphere standard network where vCenter and NSX-T Manager are deployed. There are two Tenant Tier-0 routers that connect to the Shared Tier-0 over an NSX-T logical switch using a virtual LAN (VLAN) or Overlay transport zone. Using each dedicated T0, Kubernetes clusters are deployed in complete isolation on each tenant network.



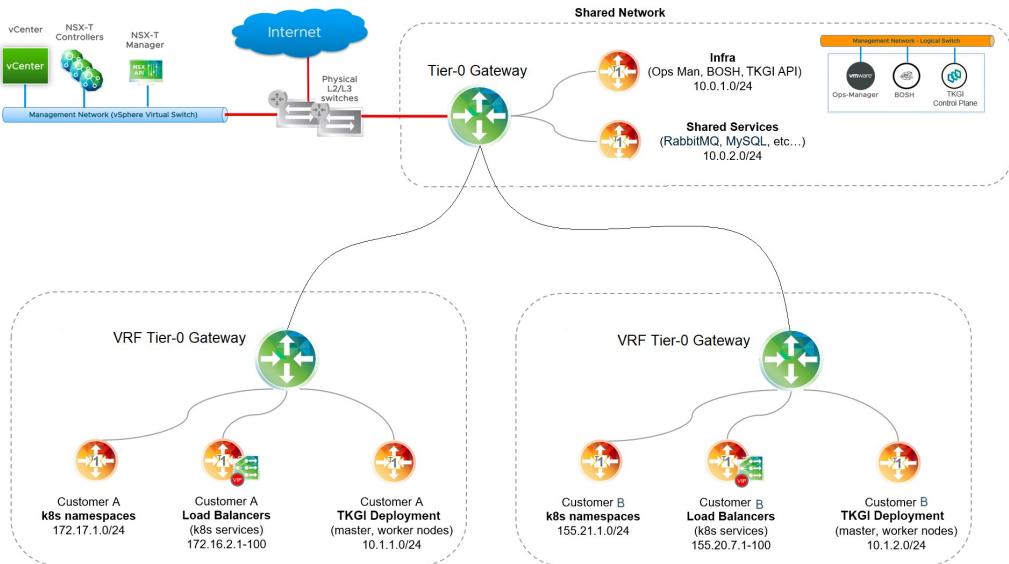
To isolate a cluster and its workloads behind TO routers:

- Prerequisites
- Configure Multi-TO Router-Based Tenant Isolation
- Configure Multi-TO Security

## Using a VRF Tier-0 Gateway Configuration for Tenant Isolation

Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T Policy API also supports provisioning, managing, and securing Kubernetes cluster deployments using a VRF gateway.

As shown in the diagram below, instead of using one or more TO routers, clusters are isolated behind a VRF gateway. The Shared Tier-0 router handles traffic between the TKGI management network and the vSphere standard network where vCenter and NSX-T Manager are deployed. Using Tenant VRF Tier-0 gateways to connect to the Shared Tier-0, Kubernetes clusters are deployed in complete isolation on tenant networks.



To isolate a cluster and its workloads behind a VRF gateway:

- Prerequisites
- Configure VRF Tier-0 Gateway-Based Tenant Isolation

## Prerequisites

The prerequisites for tenant isolation depend on the configuration used:

- Multi-TO-Based Tenant Isolation Prerequisites
- VRF Tier-0 Gateway-Based Tenant Isolation Prerequisites

## Multi-TO-Based Tenant Isolation Prerequisites

To implement Multi-TO-based tenant isolation, verify the following prerequisites:

- Supported version of vSphere IaaS is installed. See [vSphere with NSX-T Version Requirements](#).
- Supported version of VMware NSX-T Data Center is installed. See [vSphere with NSX-T Version Requirements](#).
- If you are using NAT mode for the Shared Tier-0 router, review [Considerations for NAT Topology on Shared Tier-0](#) and [Considerations for NAT Topology on Tenant Tier-0](#) before proceeding.

## VRF Tier-0 Gateway-Based Tenant Isolation Prerequisites

To implement VRF Tier-0 Gateway-based tenant isolation:

- TKGI on vSphere with NSX-T Policy API.
- Three VLANs for the VRF Tier-0 gateway.

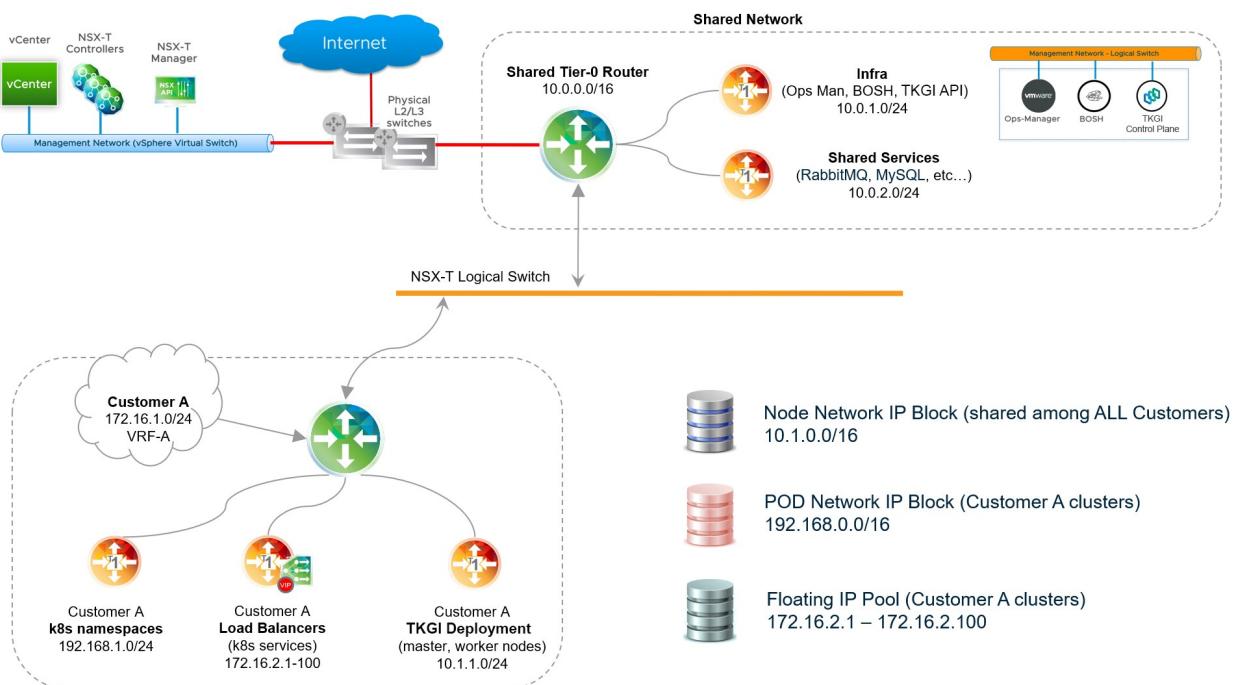
# Configure Multi-T0 Router-Based Tenant Isolation

To isolate tenants using a multi-T0 router-based configuration:

1. Plan and Provision Additional NSX-T Edge Nodes for Each Multi-T0 Router
2. Configure Inter-T0 Logical Switch
3. Configure a New Uplink Interface on the Shared Tier-0 Router
4. Provision Tier-0 Router for Each Tenant
5. Create Two Uplink Interfaces on Each Tenant Tier-0 Router
6. Verify the Status of the Shared and Tenant Tier-0 Routers
7. Configure Static Routes
8. Considerations for NAT Topology on Shared Tier-0
9. Considerations for NAT Topology on Tenant Tier-0
10. Configure BGP on Each Tenant Tier-0 Router
11. Configure BGP on the Shared Tier-0 Router
12. Test the Base Configuration

## Step 1: Plan and Provision Additional NSX-T Edge Nodes for Each Multi-T0 Router

Multi-T0 requires a minimum of four NSX-T Edge Nodes: Configure two nodes per T0. Use the T0 attached to the TKGI management plane as the Shared Tier-0 router that connects all T0 routers. In addition, deploy an additional T0 router for each tenant you want to isolate.



Each Tenant Tier-0 router requires a minimum of two NSX-T Edge Nodes. The formula for determining the minimum number of nodes for all tenants is as follows:

$$2 + (\text{TENANTS} \times 2)$$

Where **TENANTS** is the number of tenants you want to isolate.

For example, if you want to isolate three tenants, use the following calculation:

$$2 + (3 \times 2) = 8 \text{ NSX-T Edge Nodes}$$

To isolate ten tenants, use the following calculation:

$$2 + (10 \times 2) = 22 \text{ NSX-T Edge Nodes}$$

Using the NSX-T Manager interface, deploy at least the minimum number of Edge Nodes you need for each Tenant Tier-0 and join these Edge Nodes to an Edge Cluster. For more information, see [Installing and Configuring NSX-T Data Center v3.0 for TKGI](#).



**Note:** An Edge Cluster can have a maximum of 10 Edge Nodes. If the provisioning requires more Edge Nodes than what a single Edge Cluster can support, multiple Edge Clusters must be deployed.

## Step 2: Configure Inter-T0 Logical Switch

Connect all NSX-T Edge Nodes using an overlay logical switch. This overlay network is used to transport traffic between the T0 routers. Plan to allocate a network of sufficient size to accommodate all Tier-0 router interfaces that need to be connected to this network. You must allocate each T0 router one or more IP addresses from that range.

For example, if you plan to deploy two Tenant Tier-0 routers, a subnet with prefix size /28 might be sufficient, such as **50.0.0.0/28**.

Once you have physically connected the Edge Nodes, define a logical switch to connect the Shared Tier-0 router to the Tenant Tier-0 router or routers.

To define a logical switch based on an Overlay or VLAN transport zone, follow the steps below:

1. In NSX-T Manager, go to **Networking > Switching > Switches**.
2. Click **Add** and create a logical switch (LS).
3. Name the switch descriptively, such as `inter-t0-logical-switch`.
4. Connect the logical switch to the transport zone defined when deploying NSX-T. See [Installing and Configuring NSX-T Data Center v3.0 for TKGI](#).

## Step 3: Configure a New Uplink Interface on the Shared Tier-0 Router

The Shared Tier-0 router already has an uplink interface to the external (physical) network that was configured when it was created. For more information, see [Installing and Configuring NSX-T Data Center v3.0 for TKGI](#).

To enable Multi-T0, you must configure a second uplink interface on the Shared Tier-0 router that connects to the inter-T0 network (`inter-t0-logical-switch`, for example). To do this, complete the following steps:

1. In NSX-T Manager, go to **Networking > Routers**.
2. Select the Shared Tier-0 router.
3. Select **Configuration > Router Ports** and click **Add**.
4. Configure the router port as follows:
  1. For the logical switch, select the inter-T0 logical switch you created in the previous step (for example, `inter-t0-logical-switch`).
  2. Provide an IP address from the allocated range. For example, `50.0.0.1/24`.

## Step 4: Provision Tier-0 Router for Each Tenant

Create a Tier-0 logical router for each tenant you want to isolate. For more information, see [Create Tier-0 Router in \*Installing and Configuring NSX-T Data Center v3.0 for TKG\*](#).

When creating each Tenant Tier-0 router, make sure you set the router to be active/passive, and be sure to name the logical switch descriptively, such as `t0-router-customer-A`.

## Step 5: Create Two Uplink Interfaces on Each Tenant Tier-0 Router

Similar to the Shared Tier-0 router, each Tenant Tier-0 router requires at a minimum two uplink interfaces.

- The first uplink interface provides an uplink connection from the Tenant Tier-0 router to the tenant's corporate network.
- The second uplink interface provides an uplink connection to the Inter-T0 logical switch that you configured. For example, `inter-t0-logical-switch`.

For instructions, see [Create Tier-0 Router in \*Installing and Configuring NSX-T Data Center v3.0 for TKG\*](#). When creating the uplink interface that provides an uplink connection to the Inter-T0 logical switch, be sure to give this uplink interface an IP address from the allocated pool of IP addresses.

## Step 6: Verify the Status of the Shared and Tenant Tier-0 Routers

When you have completed the configuration of the Shared and Tenant Tier-0 routers as described above, verify your progress up to this point. On the Shared Tier-0 router, you should have two uplink interfaces, one to the external network and the other to the inter-T0 logical switch. On the Tenant Tier-0 router, you should have two uplink interfaces, one to the inter-T0 logical switch and the other to the external network. Each uplink interface is connected to a transport node.

The images below provide an example checkpoint for verifying the uplink interfaces for the Shared and Tenant Tier-0 routers. In this example, the Shared Tier-0 has one uplink interface at `10.40.206.10/25` on the transport Edge Node `edge-TN1`, and the second uplink interface at `10.40.206.9/25` on the transport Edge Node `edge-TN2`.

|                          |          |             |        |                 |                                     |          |  |
|--------------------------|----------|-------------|--------|-----------------|-------------------------------------|----------|--|
| <input type="checkbox"/> | Uplink-2 | 585e.....   | Uplink | 10.40.206.9/25  | uplink-LS1<br>(  8f0831de-01f1... ) | edge-TN2 |  |
| <input type="checkbox"/> | Uplink1  | e1f5...e... | Uplink | 10.40.206.10/25 | uplink-LS1<br>(  uplink1-port )     | edge-TN1 |  |

Similarly, the Tenant Tier-0 has one uplink interface at [10.40.206.13/25](#) on the transport Edge Node [edge-TN3](#), and the second uplink interface at [10.40.206.14/25](#) on the transport Edge Node [edge-TN4](#).

| <input type="checkbox"/> | Logical Router ID | Type        | IP Address/mask | Connected To    | Transport Node                     | Relay Service | Statistics |
|--------------------------|-------------------|-------------|-----------------|-----------------|------------------------------------|---------------|------------|
| <input type="checkbox"/> | TO-2-u...         | 4238.....   | Uplink          | 10.40.206.13/25 | uplink-LS1<br>(  311a54cb-48d... ) | edge-TN3      |            |
| <input type="checkbox"/> | TO-2-u...         | 8f15...f... | Uplink          | 10.40.206.14/24 | uplink-LS1<br>(  974cbf11-0b3... ) | edge-TN4      |            |

## Step 7: Configure Static Routes

For each T0 router, including the Shared Tier-0 and all Tenant Tier-0 routers, define a static route to the external network. For instructions, see [Create Tier-0 Router](#) in *Installing and Configuring NSX-T Data Center v3.0 for TKGI*.

For the Shared Tier-0 router, the default static route points to the external management components such as vCenter and NSX-T Manager and provides internet connectivity. As shown in the image below, the Shared Tier-0 defines a static route for vCenter and NSX-T Manager as [192.168.201.0/24](#), and the static route for internet connectivity as [0.0.0.0/0](#):

| tier0-shared             |                      |                        |
|--------------------------|----------------------|------------------------|
| Overview                 | Configuration        | <b>Routing</b>         |
| Static Routes            |                      |                        |
| <a href="#">+ ADD</a>    | <a href="#">EDIT</a> | <a href="#">DELETE</a> |
| <input type="checkbox"/> | Network              | ID                     |
| <input type="checkbox"/> | 0.0.0.0/0            | Oeaa...9e7c            |
| <input type="checkbox"/> | 192.168.201.0/24     | d495...b030            |
| <input type="checkbox"/> |                      | Next Hop               |
|                          |                      | 90.0.0.1               |
|                          |                      | 90.0.0.1               |

For each Tenant Tier-0 router, the default static route should point to the tenant's corporate network. As shown in the image below, the Tenant Tier-0 defines a static route to the corporate network as [0.0.0.0/0](#):

| <input type="checkbox"/> Network   | ID          | Next Hop |
|------------------------------------|-------------|----------|
| <input type="checkbox"/> 0.0.0.0/0 | 4ace...9e9d | 70.0.0.1 |

## Step 8: Considerations for NAT Topology on Shared Tier-0

The Multi-T0 configuration steps documented here apply to deployments where NAT mode is **not** used on the Shared Tier-0 router. For more information, see [NSX-T Deployment Topologies for Tanzu Kubernetes Grid Integrated Edition](#).

For deployments where NAT-mode is used on the Shared Tier-0 router, additional provisioning steps must be followed to preserve NAT functionality to external networks while bypassing NAT rules for traffic flowing from the Shared Tier-0 router to each Tenant Tier-0 router.

Existing Tanzu Kubernetes Grid Integrated Edition deployments where NAT mode is configured on the Shared Tier-0 router cannot be re-purposed to support a Multi-T0 deployment following this documentation.

## Step 9: Considerations for NAT Topology on Tenant Tier-0



**Note:** This step only applies to NAT topologies on the Tenant Tier-0 router. For more information on NAT mode, see [NSX-T Deployment Topologies for TKGI](#).



**Note:** NAT mode for Tenant Tier-0 routers is enabled by defining a non-routable custom Pods IP Block using a Network Profile. For more information, see [Defining Network Profiles](#).

In a Multi-T0 environment with NAT mode, traffic on the Tenant Tier-0 network going from Kubernetes cluster nodes to TKGI management components residing on the Shared Tier-0 router must bypass NAT rules. This is required because TKGI-managed components such as BOSH Director connect to Kubernetes nodes based on routable connectivity without NAT.

To avoid NAT rules being applied to this class of traffic, you need to create two high-priority **NO\_SNAT** rules on each Tenant Tier-0 router. These NO\_SNAT rules allow “selective” bypass of NAT for the relevant class of traffic, which in this case is connectivity from Kubernetes node networks to TKGI management components such as the TKGI API, Ops Manager, and BOSH Director, as well as to infrastructure components such as vCenter and NSX-T Manager.

For each Tenant Tier-0 router, define two NO\_SNAT rules to classify traffic. The source for both rules is the [Nodes IP Block](#) CIDR. The destination for one rule is the TKGI Management network where TKGI, Ops Manager, and BOSH Director are deployed. The destination for the other rule is the external network where NSX-T Manager and vCenter are deployed.

For example, the following image shows two NO\_SNAT rules created on a Tenant Tier-0 router. The first rule un-NATs traffic from Kubernetes nodes ([30.0.128.0/17](#)) to the TKGI management network ([30.0.0.0/24](#)). The second rule un-NATs traffic from Kubernetes nodes ([30.0.128.0/17](#)) to the external network ([192.168.201.0/24](#)).

## New NAT Rule

X

|                                      |                                                                                          |     |
|--------------------------------------|------------------------------------------------------------------------------------------|-----|
| Priority                             | 1024                                                                                     | ▼   |
| Action *                             | NO_SNAT                                                                                  | ▼   |
| Protocol                             | <input checked="" type="radio"/> Any Protocol<br><input type="radio"/> Specific Protocol |     |
| Source IP *                          | 30.0.128.0/17                                                                            |     |
| Destination IP                       | 30.0.0.0/24                                                                              |     |
| Applied To                           | t0-t0-cluster-1-vlan-uplink-1-Internet-vlan-1                                            | x ▼ |
| Status                               | <input checked="" type="checkbox"/> Enabled                                              |     |
| Logging                              | <input type="checkbox"/> Disabled                                                        |     |
| Firewall Bypass                      | <input checked="" type="checkbox"/> Enabled                                              |     |
| <span>CANCEL</span> <span>ADD</span> |                                                                                          |     |



## New NAT Rule

X

|                 |                                                                                          |          |
|-----------------|------------------------------------------------------------------------------------------|----------|
| Priority        | 1024                                                                                     | ▼        |
| Action*         | NO_SNAT                                                                                  | ▼        |
| Protocol        | <input checked="" type="radio"/> Any Protocol<br><input type="radio"/> Specific Protocol |          |
| Source IP*      | 30.0.128.0/17                                                                            |          |
| Destination IP  | 192.168.201.0/24                                                                         |          |
| Applied To      | t0-t0-cluster-1-vlan-uplink-1-Internet-vlan-1                                            |          |
| Status          | <input checked="" type="checkbox"/>                                                      | Enabled  |
| Logging         | <input type="checkbox"/>                                                                 | Disabled |
| Firewall Bypass | <input checked="" type="checkbox"/>                                                      | Enabled  |



The end result is two NO\_SNAT rules on each Tenant Tier-0 router that bypass the NAT rules for the specified traffic.

| tier0-customer-A                                                  |           |              |                |                   |                  |     |            |       |            |             |
|-------------------------------------------------------------------|-----------|--------------|----------------|-------------------|------------------|-----|------------|-------|------------|-------------|
| Overview Configuration ▾ Routing ▾ Services ▾                     |           |              |                |                   |                  |     |            |       |            |             |
| NAT   REFRESH                                                     |           |              |                |                   |                  |     |            |       |            |             |
| Total Rule Statistics   Last Updated: 11/12/2018, 3:51:22 PM      |           |              |                |                   |                  |     |            |       |            |             |
| 4 Active sessions 11177882 Packet count 14 GB Data                |           |              |                |                   |                  |     |            |       |            |             |
| <a href="#">+ ADD</a> <a href="#">EDIT</a> <a href="#">DELETE</a> |           |              |                |                   |                  |     |            |       |            |             |
| ID                                                                | Action    | Match        |                |                   |                  |     | Translated |       | Applied To |             |
| Protocol                                                          | Source IP | Source Ports | Destination IP | Destination Ports |                  |     | IP         | Ports |            |             |
| ▼ Priority: 1022                                                  |           |              |                |                   |                  |     |            |       |            |             |
| 3261                                                              | NO_SNAT   | Any          | 30.0.128.0/17  | Any               | 30.0.0.0/24      | Any | Any        | Any   | Any        | inter-tier0 |
| 3266                                                              | NO_SNAT   | Any          | 30.0.128.0/17  | Any               | 192.168.201.0/24 | Any | Any        | Any   | Any        | inter-tier0 |
| ▼ Priority: 1024                                                  |           |              |                |                   |                  |     |            |       |            |             |
| 3315                                                              | SNAT      | Any          | 30.0.128.0/24  | Any               | Any              | Any | 71.0.0.11  | Any   |            |             |
| 3318                                                              | SNAT      | Any          | 40.0.0.0/24    | Any               | Any              | Any | 71.0.0.13  | Any   |            |             |
| 3320                                                              | SNAT      | Any          | 40.0.1.0/24    | Any               | Any              | Any | 71.0.0.14  | Any   |            |             |
| 3322                                                              | SNAT      | Any          | 40.0.2.0/24    | Any               | Any              | Any | 71.0.0.15  | Any   |            |             |
| 3326                                                              | SNAT      | Any          | 40.0.3.0/24    | Any               | Any              | Any | 71.0.0.16  | Any   |            |             |

## Step 10: Configure BGP on Each Tenant Tier-0 Router

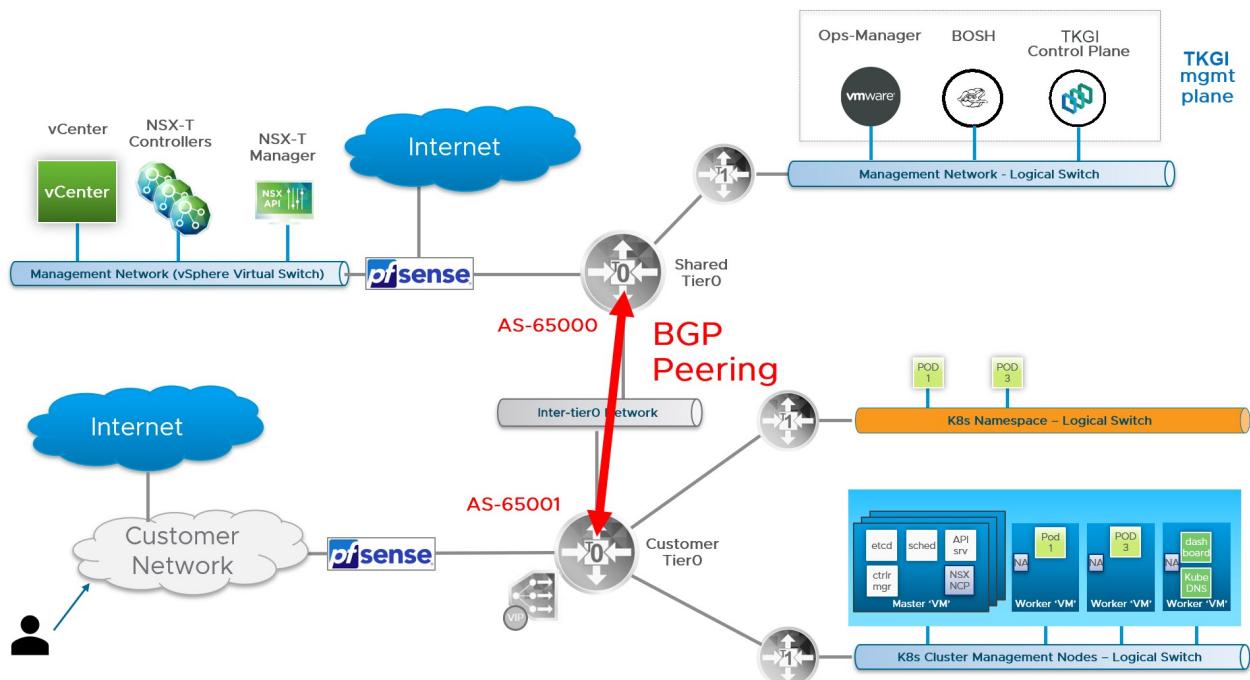
Use Border Gateway Protocol (BGP) to route redistribution and filtering across all Tier-0 routers. BGP allows the Shared Tier-0 router to dynamically discover the location of Kubernetes clusters (Node networks) deployed on each Tenant Tier-0 router.

To configure BGP on each tenant Tier-0 router:

- Considerations When Configuring BGP on Tenant Tier-0 Routers
- Configure BGP AS Number
- Configure BGP Route Distribution
- Configure IP Prefix Lists
- Configure BGP Peer

### Considerations When Configuring BGP on Tenant Tier-0 Routers

In a Multi-T0 deployment, special consideration must be given to the network design to preserve reliability and fault tolerance of the Shared and Tenant Tier-0 routers.

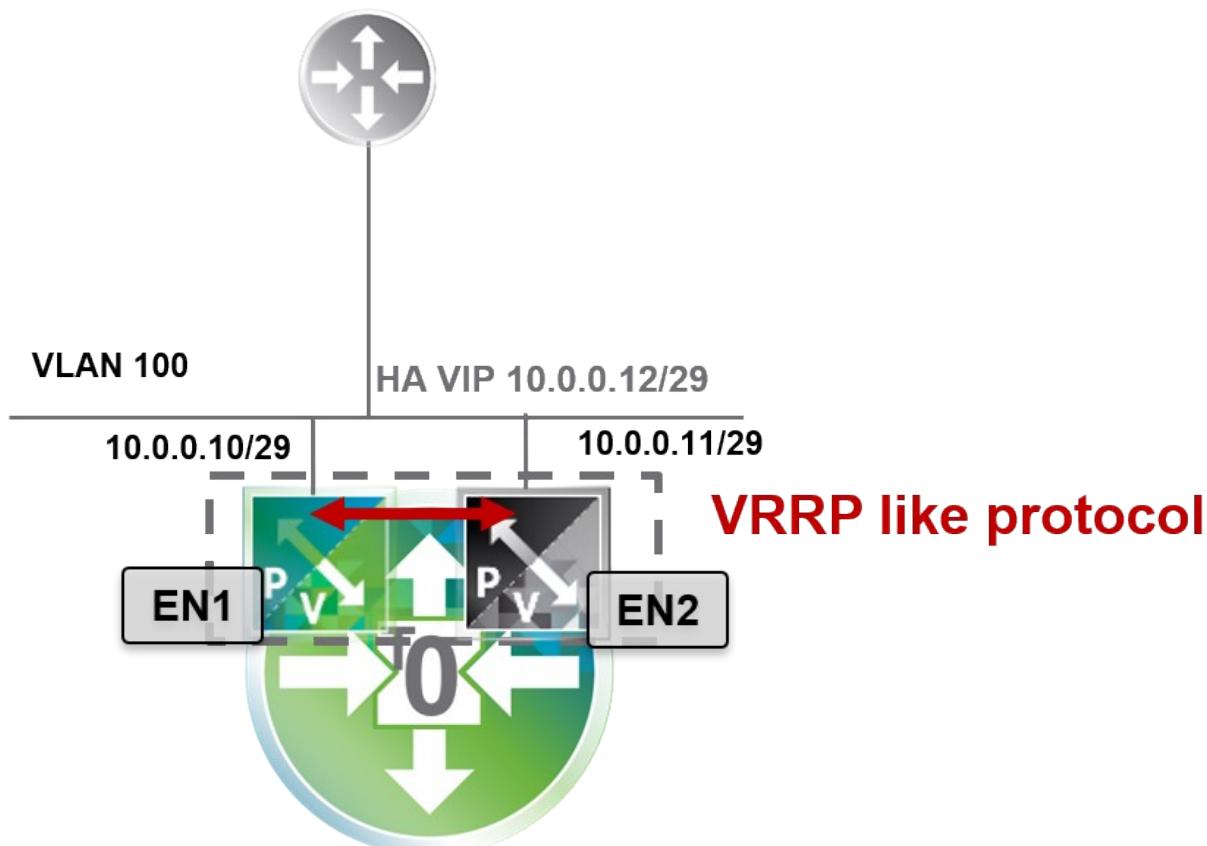


Failover of a logical router is triggered when the router is losing all of its BGP sessions. If multiple BGP sessions are established across different uplink interfaces of a Tier-0 router, failover will only occur if **all** such sessions are lost. Thus, to ensure high availability on the Shared and Tenant Tier-0 routers, BGP can only be configured on uplink interfaces facing the Inter-Tier-0 network. This configuration is shown in the diagram below.



**Note:** In a Multi-T0 deployment, BGP cannot be configured on external uplink interfaces. Uplink external connectivity must use VIP-HA with NSX-T to provide high availability for external interfaces. For more information, see [Deploy NSX-T Edge Nodes in Installing and Configuring NSX-T Data Center v3.0 for TKGI](#).

## Physical World



You must configure BGP routing on each Tier-0 router. The steps that follow are for each Tenant Tier-0 router. The instructions for the Shared Tier-0 are provided in subsequent steps. As a prerequisite, assign a unique Autonomous System Number to each Tier-0 router. Each AS number you assign must be private within the range [64512–65534](#). For more information, see [Configure BGP on a Tier-0 Logical Router](#) in the NSX-T documentation.



**Note:** To configure BGP for the Tenant Tier-0, you will need to use the Shared Tier-0 AS number. As such, identify the AS numbers you will use for the Tenant and Shared Tier-0 routers before proceeding.

### Configure BGP AS Number

Once you have chosen the AS number for the Tenant Tier-0 router, configure BGP with the chosen AS number as follows:

1. In NSX-T Manager, select **Networking > Routers**.
2. Select the Tenant Tier-0 router.
3. Select **Routing > BGP**, then click **ADD**.
4. Add the AS number to the BGP configuration in the `local_AS` field.
5. Click on the `enabled` slider to activate BGP.
6. Lastly, deactivate the ECMP slider.

## Configure BGP Route Distribution

To configure BGP route distribution for each Tenant Tier-0 router, follow the steps below:

1. In NSX-T Manager, select the Tenant Tier-0 router.
2. Select **Routing > Route Redistribution**.

Routing

| Logical Router Port | ID          | Type   |
|---------------------|-------------|--------|
| LinkedPort          | c152...e... | Uplink |
| external            | fc43...d... | Uplink |

3. Click **Add** and configure as follows:
  1. **Name:** NSX Static Route Redistribution
  2. **Sources:** Select **Static**, **NSX Static**, and **NSX Connected**

## Configure IP Prefix Lists

In this step you define an **IP Prefix List** for each Tenant Tier-0 router to advertise any Kubernetes node network of standard prefix size /24, as specified by the less-than-or-equal-to (le) and greater-than-or-equal-to (ge) modifiers in the configuration. The CIDR range to use for the definition of the list entry is represented by the **Nodes IP Block** network, for example `30.0.0.0/16`.

For more information about IP Prefix Lists, see [Create an IP Prefix List](#) in the NSX-T documentation.

To configure an IP Prefix List for each Tenant Tier-0 router, follow the steps below:

1. In NSX-T Manager, select the Tenant Tier-0 router.
2. Select **Routing > IP Prefix Lists**.
3. Click **Add** and configure as follows:
  1. **Name:** Enter a descriptive name.
  2. Click **Add** and create a **Permit** rule that allows redistribution of the exact /24 network, carved from the **Nodes IP Block**.
  3. Click **Add** and create a **Deny** rule that denies everything else on the network `0.0.0.0/0`.

## New IP Prefix List



Name\* tenant-to-IP-prefix-list

### Prefixes

| <input type="checkbox"/> Network*    | Action* | ge | le |
|--------------------------------------|---------|----|----|
| <input type="checkbox"/> 30.0.0.0/16 | Permit  | 24 | 24 |
| <input type="checkbox"/> 0.0.0.0/0   | Deny    |    |    |

CANCEL

ADD

## Configure BGP Peer

To configure BGP peering for each Tenant Tier-0 router, follow the steps below:

1. In NSX-T Manager, select the Tenant Tier-0 router.
2. Go to **Routing > BGP**.
3. Click **Add** and configure the BGP rule as follows:
  1. **Neighbor Address:** Enter the IP address of the Shared Tier-0 router.
  2. **Local Address:** Select the individual uplink interfaces facing the inter-tier0 logical switch.
  3. **Address Families:** Click **Add** and configure as follows:
    1. **Type:** `IPV4_UNICAST`.
    2. **State:** `Enabled`.
    3. **Out Filter:** Select the IP Prefix List created above.
    4. Click **Add**.
  4. Back at the **Routing > BGP** screen:
    1. Enter the Shared Tier-0 AS number.
    2. After creating the BGP neighbor, select **Edit** and click **Enable BGP**.

## Step 11: Configure BGP on the Shared Tier-0 Router

The configuration of BGP on the Shared Tier-0 is similar to the BGP configuration each Tenant Tier-0, with the exception of the IP Prefix list that permits traffic to the TKGI management network where TKGI, BOSH, and Ops Manager are located.

As with each Tenant Tier-0 router, you will need to assign a unique private AS number within the private range [64512–65534](#) to the Shared Tier-0 router. Once the AS number is assigned, use NSX-T Manager to configure the following BGP rules for the Shared Tier-0 router.

### Configure BGP AS Number

To configure BGP on the Shared Tier-0 with the AS number, complete the corresponding set of instructions in the tenant BGP section above.

### Configure BGP Route Distribution

To configure BGP route distribution for the Shared Tier-0 router, complete the corresponding set of instructions in the BGP tenant section above.

### Configure IP Prefix Lists

To configure IP prefix lists for each Tenant Tier-0 router, follow the steps below:

1. In NSX-T Manager, select the Tenant Tier-0 router.
2. Select **Routing > IP Prefix Lists**.
3. Click **Add** and configure as follows:
  1. **Name:** Enter a descriptive name.
  2. Click **Add** and create a **Permit** rule for the infrastructure components vCenter and NSX-T Manager.
  3. Click **Add** and create a **Permit** rule for the TKGI management components (TKGI, Ops Manager, and BOSH).
  4. Click **Add** and create a **Deny** rule that denies everything else on the network [0.0.0.0/0](#).

## Edit IP Prefix List - shared-prefix-list

Name\*

Prefixes

| <input type="checkbox"/> Network*         | Action* | ge | le |
|-------------------------------------------|---------|----|----|
| <input type="checkbox"/> 30.0.0.0/24      | Permit  |    |    |
| <input type="checkbox"/> 192.168.201.0/24 | Permit  |    |    |
| <input type="checkbox"/> 0.0.0.0/0        | Deny    |    |    |

**+ ADD** **DELETE** **UP** **DOWN**

**CANCEL** **SAVE**

### Configure BGP Peer

1. In NSX-T Manager, select the Tenant Tier-0 router.
2. Go to **Routing > BGP**.
3. Click **Add** and configure the BGP rule as follows:
  1. **Neighbor Address:** Enter the IP address of the Shared Tier-0 router.
  2. **Local Address:** Select **All Uplinks**.
  3. **Address Families:** Click **Add** and configure as follows:
    1. **Type:** IPV4\_UNICAST
    2. **State:** Enabled
    3. **Out Filter:** Select the IP Prefix List that includes the network where vCenter and NSX-T Manager are deployed, as well as the network where the TKGI management plane is deployed.
  4. Click **Add**.
4. Back at the **Routing > BGP** screen:
  1. Enter the Tenant Tier-0 AS number.
  2. After creating the BGP neighbor, select **Edit** and click **Enable BGP**.



**Note:** You must repeat this step for each Tenant Tier-0 router you want to peer with the Shared Tier-0 router.

## Step 12: Test the Base Configuration

Perform the following validation checks on all Tier-0 routers:

- [Shared Tier-0 Validation](#)
- [Tenant Tier-0 Validation](#)

Perform the validation checks on the Shared Tier-0 first followed by each Tenant Tier-0 router. For each Tier-0, the validation should alternate among checking for the BGP summary and the router Routing Table.

### Shared Tier-0 Validation

Verify that the Shared Tier-0 has an active peer connection to each Tenant Tier-0 router.

To verify BGP Peering:

1. In NSX-T Manager, select the Shared Tier-0 router and choose **Actions > Generate BGP Summary**.
2. Validate that the Shared Tier-0 router has one active peer connection to each Tenant Tier-0 router.

Verify that the Shared Tier-0 routing table includes all BGP routes to each Shared Tier-0:

1. In NSX-T Manager, select **Networking > Routers > Routing**.
2. Select the Shared Tier-0 router and choose **Actions > Download Routing Table**.
3. Download the routing table for the Shared Tier-0 and verify the routes.

### Tenant Tier-0 Validation

Verify that the Shared Tier-0 has an active peer connection to each Tenant Tier-0 router.

To verify BGP Peering:

1. In NSX-T Manager, select the Tenant Tier-0 router and choose **Actions > Generate BGP Summary**.
2. Validate that the Tenant Tier-0 router has one active peer connection to the Shared Tier-0 router.
3. Repeat for all other Tenant Tier-0 routers.

Verify that the TO routing table for each Tenant Tier-0 includes all BGP routes to reach vCenter, NSX-T Manager, and the TKGI management network:

1. In NSX-T Manager, select **Networking > Routers > Routing**.
2. Select the TO router and choose **Actions > Download Routing Table**.
3. Download the routing table for each of the Tenant Tier-0 routers.



**Note:** At this point, the Shared Tier-0 has no BGP routes because you have not deployed any Kubernetes clusters. The Shared Tier-0 will show BGP routes when you deploy Kubernetes clusters to the Tenant Tier-0 routers. Each Tenant Tier-0 router shows a BGP exported route that makes each Tenant Tier-0 router aware of the TKGI management network and other external networks where NSX-T and vCenter are deployed.

## Configure Multi-T0 Security

In a multi-T0 environment, you can secure two types of traffic:

- Traffic between tenants. See [Secure Inter-Tenant Communications](#).
- Traffic between clusters in the same tenant. See [Secure Intra-Tenant Communications](#).

### Secure Inter-Tenant Communications

Securing traffic between tenants isolates each tenant and ensures the traffic between the Tenant Tier-0 routers and the Shared Tier-0 router is restricted to the legitimate traffic path.

To secure traffic between tenants:

1. Define IP Sets
2. Create Edge Firewall
3. Add Firewall Rules
4. Create DFW Section

#### Step 1: Define IP Sets

In NSX-T an **IP Set** is a group of IP addresses that you can use as sources and destinations in firewall rules. For a Multi-T0 deployment you need to create several IP Sets as described below. For more information about creating IP Sets, see [Create an IP Set](#) in the NSX-T documentation.

The image below shows a summary of the three required IP Sets you will need to create for securing Multi-T0 deployments:

| Groups                                                    |                      |                        |                           |
|-----------------------------------------------------------|----------------------|------------------------|---------------------------|
| Groups                                                    | IP Sets              | IP Pools               | MAC Sets                  |
| <a href="#">+ ADD</a>                                     | <a href="#">EDIT</a> | <a href="#">DELETE</a> | <a href="#">ACTIONS</a> ▾ |
| <input type="checkbox"/> <a href="#">IP Set ↑</a>         |                      |                        | ID                        |
| <input type="checkbox"/> <a href="#">inter-tier0-CIDR</a> |                      |                        | 9c95...54fe               |
| <input type="checkbox"/> <a href="#">NSX/vCenter</a>      |                      |                        | d5c5...ac2b               |
| <input type="checkbox"/> <a href="#">pks-admin-CIDR</a>   |                      |                        | 4b88...b917               |

First, define an IP Set that includes the IP addresses for the NSX-T Manager and vCenter hosts. In the following IP Set example, `192.168.201.51` is the IP address for NSX and `192.168.201.20` is the IP address for vCenter.

The screenshot shows the 'NSX/vCenter' IP Set configuration. The 'Members' section is expanded, showing two IP addresses: `192.168.201.51` and `192.168.201.20`.

| Members                     |  |
|-----------------------------|--|
| <code>192.168.201.51</code> |  |
| <code>192.168.201.20</code> |  |

Next, define an IP Set that includes the network CIDR for TKGI management components. In the following IP Set example, `30.0.0.0/24` is the CIDR block for the TKGI Management network.

The screenshot shows the 'pks-admin-CIDR' IP Set configuration. The 'Members' section is expanded, showing one CIDR block: `30.0.0.0/24`.

| Members                  |  |
|--------------------------|--|
| <code>30.0.0.0/24</code> |  |

Lastly, define an IP Set for the Inter-Tier0 CIDR created during the base configuration.

The screenshot shows the 'inter-tier0-CIDR' IP Set configuration. The 'Members' section is expanded, showing one CIDR block: `50.0.0.1/24`.

| Members                  |  |
|--------------------------|--|
| <code>50.0.0.1/24</code> |  |



**Note:** These are the minimum IP Sets you need to create. You might want to define additional IP Sets for convenience.

## Step 2: Create Edge Firewall

NSX-T Data Center uses Edge Firewall sections and rules to specify traffic handling in and out of the network. A firewall section is a collection of firewall rules. For more information, see [About Firewall Rules](#) in the NSX-T documentation.

For each Tenant Tier-0 router, create an Edge Firewall and section as follows:

1. In NSX Manager, go to **Networking > Routers**.
2. Select the Tenant Tier-0 router and click **Services > Edge Firewall**.
3. Select the **Default LR Layer 3 Section**.
4. Click **Add Section > Add Section Above**.

The screenshot shows the NSX Manager interface for a 'tier0' router. The top navigation bar includes 'Overview', 'Configuration', 'Routing', and 'Services'. The 'Services' tab is selected, showing the 'Edge Firewall' section. Below the navigation, there are buttons for 'REFRESH' and 'ENABLE FIREWALL'. The main area has tabs for '+ ADD RULE', '+ ADD SECTION', and 'DELETE'. An 'ACTIONS' dropdown is open, with 'Add Section Above' highlighted. A table lists firewall sections, including one named '3990c366-d614-4173-83ff-43a03...' with the state 'Stateful'. At the bottom right of the interface, there are 'OBJECTS' and search/filter buttons.

5. Configure the section as follows:
  1. **Section Name:** Enter a unique name for the firewall section.
  2. **State:** **Stateful**

### Add Section

The screenshot shows the 'Add Section' dialog box. It has fields for 'Section Name\*' (Customer-A-Firewall), 'Description', and 'State' (with 'Stateful' selected). At the bottom are 'CANCEL' and 'ADD' buttons, with a cursor pointing at the 'ADD' button.

|                                      |                                                                           |
|--------------------------------------|---------------------------------------------------------------------------|
| Section Name*                        | Customer-A-Firewall                                                       |
| Description                          |                                                                           |
| State                                | <input checked="" type="radio"/> Stateful <input type="radio"/> Stateless |
| <span>CANCEL</span> <span>ADD</span> |                                                                           |

## Step 3: Add Firewall Rules

The last step is to define several firewall rules for the Edge Firewall. The firewall rules allow only legitimate control plane traffic to traverse the inter-Tier-0 logical switch, and deny all other traffic.

The following image shows a summary of the five firewall rules you will create:

**tier0-customer-A**

| # | Name                            | ID   | Direction | Sources        | Destinations | Services | Action | Applied To  |
|---|---------------------------------|------|-----------|----------------|--------------|----------|--------|-------------|
| 1 | BGP                             | 3159 | IN_OUT    | inter-tier0... | inter-ti...  | Any      | ALLOW  | inter-tier0 |
| 2 | ClusterA Masters to NSX/vCenter | 3157 | OUT       | lb-pks-d3...   | NSX/v...     | Any      | ALLOW  | inter-tier0 |
| 3 | k8s Nodes to BOSH               | 3158 | OUT       | all-pks-no...  | BOSH         | Any      | ALLOW  | inter-tier0 |
| 4 | PKS to k8s Nodes                | 3154 | IN        | pks-admi...    | all-pks...   | Any      | ALLOW  | inter-tier0 |
| 5 | Deny All                        | 3153 | IN_OUT    | Any            | Any          | Any      | DROP   | inter-tier0 |



**Note:** All firewall rules are applied to the Inter-TO-Uplink interface.

Select the Edge Firewall **Section** you just created, then select **Add Rule**. Add the following five firewall rules:

- [BGP Firewall Rule](#)
- [Clusters Masters Firewall Rule](#)
- [Node Network to Management Firewall Rule](#)
- [TKGI Firewall Rule](#)
- [Deny All Firewall Rule](#)

### BGP Firewall Rule

- **Name:** [BGP](#)
- **Direction:** in and out
- **Source:** IP Set defined for the Inter-TO CIDR
- **Destination:** IP Set for Inter-TO CIDR
- **Service:** Any
- **Action:** Allow
- Apply the rule to the Inter-TO-Uplink interface.
- Save the firewall rule.

### Clusters Masters Firewall Rule

The source for this firewall rule is a Namespace Group (NSGroup) you define in NSX Manager. The NSGroup is the Bootstrap Security Group specified in the Network Profile associated with this tenant. See [Bootstrap Security Group \(NSGroup\)](#).

Once you have defined the NSGroup, configure the firewall rule as follows.

- **Name:** Clusters-Masters-to-NSX-and-VC
- **Direction:** out
- **Source:** NSGroup for Kubernetes Control Plane Nodes
- **Destination:** IP Set for Inter-T0 CIDR
- **Service:** Any
- **Action:** Allow
- Apply the rule to the Inter-T0-Uplink interface.
- Save the firewall rule.

#### Node Network to Management Firewall Rule

This firewall rule allows Kubernetes node traffic to reach TKGI management VMs and the standard network.

- **Name:** Node-Network-to-Management
- **Direction:** out
- **Source:** IP Set defined for the Nodes IP Block network
- **Destination:** IP Sets defined for vCenter, NSX Manager, and TKGI management plane components
- **Service:** Any
- **Action:** Allow
- Apply the rule to the Inter-T0-Uplink interface.
- Save the firewall rule.

#### TKGI Firewall Rule

This firewall rule allows TKGI management plane components to talk to Kubernetes nodes.

- **Name:** TKGI-to-Node-Network
- **Direction:** ingress
- **Source:** IP Set defined for the TKGI management network
- **Destination:** IP Set defined for the Nodes IP Block network
- **Service:** Any
- **Action:** Allow
- Apply the rule to the Inter-T0-Uplink interface.
- Save the firewall rule.

## Deny All Firewall Rule

- **Name:** `Deny All`. This setting drops all other traffic that does not meet the criteria of the first three rules.
- **Direction:** in and out
- **Source:** Any
- **Destination:** Any
- **Service:** Any
- **Action:** Drop
- Apply the rule to the Inter-T0-Uplink interface.
- Save the firewall rule.

## (Optional) Step 4: Create DFW Section

To use distributed firewall (DFW) rules, you must create a DFW section for the DFW rule set. The DFW section must exist before you create a Kubernetes cluster.

This optional step is recommended for inter-tenant security. It is required for intra-tenant security as described in [Secure Intra-Tenant Communications](#). Because you need to create the DFW section only once, you can use the DFW section you configure in this step when defining DFW rules for intra-tenant communications.

Even if you do not currently plan to use DFW rules, you can create the DFW section and use it later if you decide to define DFW rules. Those rules will apply to any cluster created after you define the DFW section for the tenant Tier-0 router.



**Note:** You must perform this procedure before you deploy a Kubernetes cluster to the target tenant Tier-0 router.

1. In NSX Manager, navigate to **Security > DFW**, select the top-most rule, and click **Add Section Above**.
2. Configure the section as follows:
  1. In the **Section Name** field, enter a name for your DFW section. For example, `tkgi-dfw`.
  2. Use the defaults for all other settings on the **New Section** page.
  3. Navigate to the **Manage Tags** page and add a new tag.
    1. In the **Tag** field, enter `top`.
    2. In the **Scope** field, enter `ncp/fw_sect_marker`.

## Secure Intra-Tenant Communications

To secure communication between clusters in the same tenancy, you must disallow any form of communication between Kubernetes clusters created by TKGI. Securing inter-cluster

communications is achieved by provisioning security groups and DFW rules.



**Note:** You must perform the global procedures, the first three steps described below, before you deploy a Kubernetes cluster to the target tenant Tier-0 router.

To secure communication between clusters in the same tenancy:

1. Create NSGroup for All Tanzu Kubernetes Grid Integrated Edition Clusters
2. Create DFW Section
3. Create NSGroups
4. Create DFW Rules

### Step 1: Create NSGroup for All Tanzu Kubernetes Grid Integrated Edition Clusters

1. In NSX Manager, navigate to **Inventory > Groups > Groups** and **Add new group**.
2. Configure the new NSGroup as follows:
  1. In the **Name** field, enter `All-TKGI-Clusters`.
  2. In the **Membership Criteria** tab, add the following two criteria:
    1. For the first criterion, select **Logical switch**.
    2. For **Scope > Equals**, enter `pks/clusters`.
    3. For **Scope > Equals**, enter `pks/floating_ip`.
    4. For the second criterion, select **Logical switch**.
    5. For **Scope > Equals**, enter `ncp/cluster`.

#### Edit NSGroup - All-PKS-Clusters

| General             |  | Membership Criteria |        | Members |        |
|---------------------|--|---------------------|--------|---------|--------|
| Maximum Criteria: 5 |  |                     |        |         |        |
| Logical Switch      |  | Tag                 | Equals | Scope   | Equals |
| AND                 |  | Tag                 | Equals | Scope   | Equals |
| Logical Switch      |  | Tag                 | Equals | Scope   | Equals |
|                     |  |                     |        |         |        |
|                     |  |                     |        |         |        |
|                     |  |                     |        |         |        |



**Note:** The `pks/clusters`, `pks/floating_ip`, or `ncp/cluster` values are the exact values you must enter when configuring **Scope > Equals**. They map to NSX-T objects.

After you configure the `All-TKGI-Clusters` NSGroup, the **Membership Criteria** tab looks as follows:

## All-PKS-Clusters

Overview    **Membership Criteria**    Members    Applications    Related ▾

Membership Criteria | [EDIT](#)

- |                   |                                                           |
|-------------------|-----------------------------------------------------------|
| 1. Logical Switch | Scope Equals pks/clusters<br>Scope Equals pks/floating_ip |
| 2. Logical Switch | Scope Equals ncp/cluster                                  |

### Step 2: Create DFW Section

Before you create distributed firewall rules, you must create a DFW section for the DFW rule set you define later.

To create a DFW section, follow the instructions in [Create DFW Section](#).

### Step 3: Create NSGroups

Before creating NSGroups, retrieve the UUID of the cluster that you want to secure. To retrieve the cluster UUID, run the `tkgi cluster YOUR-CLUSTER-NAME` command. For more information about the TKGI CLI, see [TKGI CLI](#).

#### Create NSGroup for Cluster Nodes

1. In NSX Manager, navigate to **Inventory > Groups > Groups** and click **Add new group**.
2. Configure the new NSGroup as follows:
  1. In the **Name** field, enter the cluster UUID or cluster name and append `-nodes` to the end of the name to distinguish it. The cluster name must be unique.
  2. In the **Membership Criteria** tab, add the following criterion:
    1. Select **Logical Switch**.
    2. For **Tag > Equals**, enter `tkgi-cluster-YOUR-CLUSTER-UUID`.
    3. For **Scope > Equals**, enter `pks/cluster`.
    4. For **Scope > Equals**, enter `pks/floating_ip`. For this scope, leave the **Tag** field empty as shown in the image below.

#### Edit NSGroup - ClusterA-nodes

General    **Membership Criteria**    Members

Maximum Criteria: 5

| Logical Switch | Tag | Equals | scope                               | Equals | scope                        |
|----------------|-----|--------|-------------------------------------|--------|------------------------------|
| AND            | Tag | Equals | <code>pks-cluster-8de00ff-a6</code> | Scope  | Equals                       |
|                |     |        |                                     | Scope  | Equals                       |
|                |     |        |                                     |        | <code>pks/floating_ip</code> |

After you configure the NSGroup for cluster nodes, the **Membership Criteria** tab looks as follows:

**ClusterA-nodes**

**Overview    Membership Criteria    Members    Applications    Related**

**Membership Criteria** | [EDIT](#)

1. Logical Switch      Tag Equals pks-cluster-8de000ff-a87a-4930-81ba-106d42c2471e  
Scope Equals pks/cluster  
Scope Equals pks/floating\_ip

#### Create NSGroup for Cluster Pods

1. In NSX Manager, navigate to **Inventory > Groups > Groups** and click **Add new group**.
2. Configure the new NSGroup as follows:
  1. In the **Name** field, enter the cluster UUID or cluster name and append `-pods` to the end of the name to distinguish it. The cluster name must be unique.
  2. In the **Membership Criteria** tab, add the following criterion:
    1. Select **Logical Port**.
    2. For **Tag > Equals**, enter `tkgi-cluster-YOUR-CLUSTER-UUID`.
    3. For **Scope > Equals**, enter `ncp/cluster`.

Edit NSGroup - ClusterA-pods

[?](#) [X](#)

**General    Membership Criteria    Members**

Maximum Criteria: 5

|              |   |     |   |        |   |                                                  |       |        |   |             |   |
|--------------|---|-----|---|--------|---|--------------------------------------------------|-------|--------|---|-------------|---|
| Logical Port | ▼ | Tag | ▼ | Equals | ▼ | pks-cluster-8de000ff-a87a-4930-81ba-106d42c2471e | Scope | Equals | ▼ | ncp/cluster | + |
|--------------|---|-----|---|--------|---|--------------------------------------------------|-------|--------|---|-------------|---|

[+ CRITERIA](#) [CLEAR ALL](#)

[CANCEL](#) [SAVE](#)

After you configure the NSGroup for cluster pods, the **Membership Criteria** tab looks as follows:

**ClusterA-pods**

**Overview    Membership Criteria    Members    Applications    Related**

**Membership Criteria** | [EDIT](#)

1. Logical Port      Tag Equals pks-cluster-8de000ff-a87a-4930-81ba-106d42c2471e  
Scope Equals ncp/cluster

#### Create NSGroup for Cluster Nodes and Pods

1. In NSX Manager, navigate to **Inventory > Groups > Groups** and click **Add new group**.

2. Configure the new NSGroup as follows:

1. In the **Name** field, enter the cluster UUID or cluster name and append `-nodes-pods` to the end of the name to distinguish it. The cluster name must be unique.
2. In the **Membership Criteria** tab, add the following two criteria:
  1. For the first criterion, select **Logical Port**.
  2. For **Tag > Equals**, enter `tkgi-cluster-YOUR-CLUSTER-UUID`.
  3. For **Scope > Equals**, enter `ncp/cluster`.
  4. For the second criterion, select **Logical Switch**.
  5. For **Tag > Equals**, enter `tkgi-cluster-YOUR-CLUSTER-UUID`.
  6. For **Scope > Equals**, enter `pks/cluster`.

Edit NSGroup - ClusterA-nodes-pods

| Logical Port   | Tag | Equals | pks-cluster-8de000ff-a8 | Scope | Equals | ncp/cluster |
|----------------|-----|--------|-------------------------|-------|--------|-------------|
| Logical Switch | Tag | Equals | pks-cluster-8de000ff-a8 | Scope | Equals | pks/cluster |

[+ CRITERIA](#)

After you configure the NSGroup for cluster nodes and pods, the **Membership Criteria** tab looks as follows:

|                   |                                                                                         |
|-------------------|-----------------------------------------------------------------------------------------|
| 1. Logical Port   | Tag Equals pks-cluster-8de000ff-a87a-4930-81ba-106d42c2471e<br>Scope Equals ncp/cluster |
| 2. Logical Switch | Tag Equals pks-cluster-8de000ff-a87a-4930-81ba-106d42c2471e<br>Scope Equals pks/cluster |

## Step 4: Create DFW Rules

Select the DFW section you created above and configure the following three DFW rules:

- Deny Everything Else
- Prevent Pod to Node Communication
- Allow Node to Node and Nodes to Pods Communications

### DFW Rule 1: Deny Everything Else

This is a global deny rule. Configure the rule as follows:

1. Click **Add Rule**.
2. In the **Name** field, enter a name for your DFW rule.
3. For **Source**, select the `All-TKGI-Clusters` NSGroup.
4. For **Destination**, select the `All-TKGI-Clusters` NSGroup.
5. For **Service**, select **Any**.
6. For **Apply To**, select the `YOUR-CLUSTER-UUID-nodes-pods` NSGroup.
7. For **Action**, select **Drop**.

#### DFW Rule 2: Prevent Pod to Node Communication

Configure this rule as follows:

1. Click **Add Rule**.
2. In the **Name** field, enter a name for your DFW rule.
3. For **Source**, select the `YOUR-CLUSTER-UUID-pods` NSGroup.
4. For **Destination**, select `YOUR-CLUSTER-UUID-nodes` NSGroup.
5. For **Service**, select **Any**.
6. For **Apply To**, select the `YOUR-CLUSTER-UUID-nodes-pods` NSGroup.
7. For **Action**, select **Drop**.

#### DFW Rule 3: Allow Node to Node and Nodes to Pods Communications

Configure this rule as follows:

1. Click **Add Rule**.
2. In the **Name** field, enter a name for your DFW rule.
3. For **Source**, select the `YOUR-CLUSTER-UUID-nodes-pods` NSGroup.
4. For **Destination**, select `YOUR-CLUSTER-UUID-nodes-pods` NSGroup.
5. For **Service**, select **Any**.
6. For **Apply To**, select `YOUR-CLUSTER-UUID-nodes-pods` NSGroup.
7. For **Action**, select **Allow**.

For example, see the three configured DFW rules below:

| # | Name                                       | Source                         | Destination           | Service | Applied To            | Log   | Action    |
|---|--------------------------------------------|--------------------------------|-----------------------|---------|-----------------------|-------|-----------|
| ⋮ | ⊖ □ hc-lb-nks-d3eebd0a-37e0-483e-80a2-7... |                                |                       |         | ⓘ Applied To: 1       |       |           |
| ⋮ | ⊖ □ hc-ip-pks-d3eebd0a-37e0-483e-80a2-7... |                                |                       |         | ⓘ Applied To: 1       |       |           |
| ⋮ | ✓ HiPrio Section                           |                                |                       |         | ⓘ Applied To: All     |       |           |
| ⋮ | ✓ 3 ClusterA_pods_to_no...                 | ⓘ clusterA-pods<br>ID: 3161    | ⓘ clusterA-nodes      | Any     | ⓘ clusterA-nodes-p... | Off ⏪ | 🔴 Drop ⌂  |
| ⋮ | ✓ 4 intra-Cluster traffic                  | ⓘ clusterA-nodes-p...          | ⓘ clusterA-nodes-p... | Any     | ⓘ clusterA-nodes-p... | Off ⏪ | 🟢 Allow ⌂ |
| ⋮ | ✓ 5 inter-Cluster traffic                  | ⓘ all-PKS-clusters<br>ID: 3163 | ⓘ all-PKS-clusters    | Any     | ⓘ clusterA-nodes-p... | Off ⏪ | 🔴 Drop ⌂  |

## Configure VRF Tier-0 Gateway-Based Tenant Isolation

To isolate a cluster and its workloads behind a VRF gateway:

- [Review Your Network Configuration](#)
- [Create VRF Gateway Segments](#)
- [Create VRF Gateways](#)
- [Create a Network Profile](#)
- [Configure a Cluster with a VRF Gateway](#)

### Step 1: Review Your Network Configuration

To review the network configuration of your three VLANs:

1. To determine the VLAN IDs of your three VLANs, run either of the following for each VLAN:
  - ◊ Method one:
 

```
sudo cat /proc/net/vlan/VLAN-NAME |grep VID
```

 Where VLAN-NAME is the name of a single VLAN.
  - ◊ Method two:
 

```
ip -d link show dev VLAN-NAME |grep id
```

 Where VLAN-NAME is the name of a single VLAN.
2. Confirm that a t0-shared gateway uses the VLAN IDs and that its segment matches the segments returned by the commands above.

### Step 2: Create VRF Gateway Segments

You must create two VLAN-backed segments for your VRF gateways. For information on creating a VLAN-backed segment, see [Add a Segment](#) in the VMware NSX-T Data Center documentation.

To create two gateway segments:

1. Create a VLAN-backed segment for one of your VRF gateway VLANs with the following configuration:
  - ◊ Configure **Segment Name**. For example, `internet-vlan-vrf-0-seg`.
  - ◊ Configure **Transport Zone**. For example, `internet-tz-vlan-0`.
  - ◊ Configure **VLAN**. Specify one of the VLAN IDs determined above.
2. Create a VLAN-backed segment for your remaining VRF gateway VLAN with the following configuration:

- Configure **Segment Name** with a new unique name. For example, `internet-vlan-vrf-1-seg`.
- Configure **Transport Zone** with the same zone used by the first segment.
- Configure **VLAN** with the VLAN ID for the second VLAN.

## Step 3: Create VRF Gateways

You must create two VRF gateways to isolate your tenants. For information on creating a VRF gateway, see [Add a VRF Gateway](#) in the VMware NSX-T Data Center documentation.

1. Create a VRF gateway with the following configuration:
  1. Provide a **Name** for the VRF gateway. For example, `t0-vrf-0`.
  2. Connect the VRF gateway to your Tier-0 Gateway.
  3. Save your configuration.
  4. Set the interface for the gateway. For example, `t0-vrf-0-uplink-0` and `t0-vrf-0-uplink-1`.
  5. Assign a high availability VIP. For example, `192.168.116.2`.
2. Create a second VRF gateway with the following configuration:
  - Configure **Name** with a unique name. For example, `t0-vrf-1`.
  - Configure **Interface** with unique settings. For example, `t0-vrf-1-uplink-0` and `t0-vrf-1-unlink-1`.
  - Configure **HA VIP** with a unique IP Address. For example, `192.168.117.2`.
3. To test your configuration, Ping each gateway uplink VIP.

For example:

```
$ ping 192.168.116.2

PING 192.168.116.2 (192.168.116.2) 56(84) bytes of data.

64 bytes from 192.168.116.2: icmp_seq=1 ttl=64 time=0.478 ms
64 bytes from 192.168.116.2: icmp_seq=2 ttl=64 time=0.520 ms
^C

- 192.168.116.2 ping statistics -
2 packets transmitted, 2 received, 0% packet loss, time 999ms
rtt min/avg/max/mdev = 0.478/0.499/0.520/0.021 ms

$ ping 192.168.117.2
```

```
PING 192.168.117.2 (192.168.117.2) 56(84) bytes of data.

64 bytes from 192.168.117.2: icmp_seq=1 ttl=64 time=0.531 ms

64 bytes from 192.168.117.2: icmp_seq=2 ttl=64 time=0.504 ms

^C

-- 192.168.117.2 ping statistics --

2 packets transmitted, 2 received, 0% packet loss, time 999ms

rtt min/avg/max/mdev = 0.504/0.517/0.531/0.026 ms
```

4. (Optional) To allow communication to an external data path, add a default router for each VRF gateway. For each router, add a default route and configure **Network** and **Next Hop**.

## Step 4: Create a Network Profile

You must use a Network Profile to isolate a cluster behind a VRF gateway.

To configure a Network Profile for connecting to a VRF gateway:

1. Create a network profile configuration JSON file that defines the gateway as the **"t0\_router\_id"** value:

```
{
 "name": "PROFILE-NAME",
 "description": "PROFILE-DESCRIP",
 "parameters": {
 "t0_router_id": "VRF-GATEWAY-NAME",
 "infrastructure_networks": [NETWORK-RANGES],
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "extensions": {
 "ncp": {
 "nsx_v3": {},
 "coe": {},
 "ha": {},
 "k8s": {}
 }
 },
 "nsx-node-agent": {}
 }
 }
 }
}
```

```
}
```

Where:

- ◊ `VRF-GATEWAY-NAME` is the name of the VRF gateway the cluster should use.
- ◊ `NETWORK-RANGES` is an array of IP ranges the cluster can access.
- ◊ `PROFILE-NAME` is the internal name for your network profile.
- ◊ `PROFILE-DESCRIP` is an internal description for your network profile.

For example:

```
{
 "name": "np-1",
 "description": "",
 "parameters": {

 "t0_router_id": "vrf-103",
 "infrastructure_networks": ["88.0.0.0/24", "192.168.111.98", "192.168.111.46"],
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {

 "extensions": {
 "ncp": {
 "nsx_v3": {},
 "coe": {},
 "ha": {},
 "k8s": {}
 }
 },
 "nsx-node-agent": {}
 }
 }
 }
}
```

For more information on creating a Network Profile, see [Creating and Managing Network Profiles](#).

## Step 5: Configure a Cluster with a VRF Gateway

To configure a cluster to use a VRF gateway, assign the Network Profile to the cluster:

- Create a new cluster using the VRF gateway Network Profile.

For more information on creating clusters using a Network Profile, see [Create a Cluster with a Network Profile](#) in *Using Network Profiles*.

- Update an existing cluster using the VRF gateway Network Profile.

For more information on updating existing clusters with a Network Profile, see [Assign a Network Profile to an Existing Cluster](#) in *Using Network Profiles*.

## Implementing a Multi-Foundation Tanzu Kubernetes Grid Integrated Edition Deployment

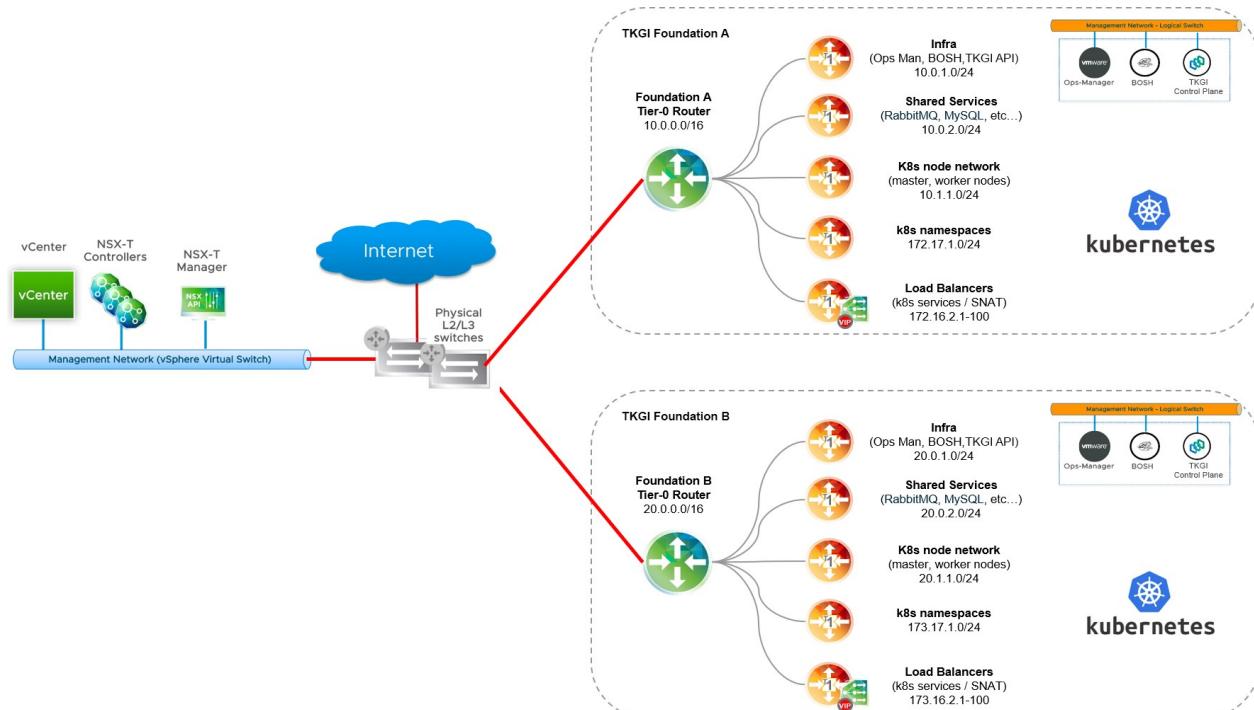
This topic describes how to deploy multiple instances of Tanzu Kubernetes Grid Integrated Edition (TKGI) on vSphere with NSX-T infrastructure.

## About Multi-Foundation Tanzu Kubernetes Grid Integrated Edition

A multi-foundation deployment of Tanzu Kubernetes Grid Integrated Edition lets you install and run multiple instances of Tanzu Kubernetes Grid Integrated Edition. The purpose of a multi-foundation deployment of Tanzu Kubernetes Grid Integrated Edition is to share a common vSphere and NSX-T infrastructure across multiple foundations, while providing complete networking isolation across foundations.

As shown in the diagram, with a multi-foundation Tanzu Kubernetes Grid Integrated Edition topology, each TKGI instance is deployed to a dedicated NSX-T Tier-0 router. Foundation A TO router with Management CIDR 10.0.0.0/16 connects to the vSphere and NSX-T infrastructure. Similarly, Foundation B TO router with Management CIDR 20.0.0.0/16 connects to the same vSphere and NSX-T components.

As with a single instance deployment, TKGI management components are deployed to a dedicated network, for example, 10.0.0.0/24 for TKGI Foundation A; 20.0.0.0/24 for TKGI Foundation B. When Tanzu Kubernetes Grid Integrated Edition is deployed, networks are defined for nodes, pods, and load balancers. Because of the dedicated Tier-0 router, there is complete networking isolation between each Tanzu Kubernetes Grid Integrated Edition instance.



# Requirements

To implement a multi-foundation Tanzu Kubernetes Grid Integrated Edition topology, adhere to the following requirements:

- One Tier-0 router for each Tanzu Kubernetes Grid Integrated Edition instance. For more information, see [Isolating Tenants](#).
- The Floating IP pool must not overlap. The CIDR range for each Floating IP Pool must be unique and not overlapping across foundations. For more information, see [Create Floating IP Pool](#).
- Tanzu Kubernetes Grid Integrated Edition instances can be deployed in NAT and no-NAT mode. If more than one Tanzu Kubernetes Grid Integrated Edition instance is deployed in no-NAT mode, the Nodes IP Block networks cannot overlap.
- For any Pods IP Block used to deploy Kubernetes clusters in no-NAT (routable) mode, the Pods IP Block cannot overlap across foundations.
- The [NSX-T Super User Principal Identity Certificate](#) should be unique per TKGI instance.

The image below shows three Tanzu Kubernetes Grid Integrated Edition installations across three Tier-0 foundations. Key considerations to keep in mind with a multi-foundation Tanzu Kubernetes Grid Integrated Edition topology include the following:

- Each foundation must rely on a dedicated Tier-0 router
- You can mix-and-match NAT and no-NAT mode across foundations for Node and Pod networks
- If you are using non-routable Pods IP Block networks, the Pods IP Block addresses can overlap across foundations
- Because Kubernetes nodes are behind a dedicated Tier-0 router, if clusters are deployed in NAT mode the Nodes IP Block addresses can also overlap across foundations
- For each foundation you must define a unique Floating ID Pool with non-overlapping IPs

| TKGI Foundation A                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | TKGI Foundation B                                                                                                                                    | TKGI Foundation C                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <input checked="" type="checkbox"/> NAT mode<br>Pods IP Block ID *<br><input type="text" value="927a2eff-fd86-4df6-bb21-c45b5314f547"/><br><br>Nodes IP Block ID *<br><input type="text" value="3d577e5c-acaf-4921-945b-d12b0e1318e6"/><br><br>T0 Router ID *<br><input type="text" value="40445803-8c3c-417e-bb24-a84cf90330b5"/><br><br>Floating IP Pool ID *<br><input type="text" value="86213c33-9b7a-4a91-b470-7145941bccb3"/><br><br>Nodes DNS *<br><input type="text" value="10.40.53.1"/><br><br>vSphere Cluster Names *<br><input type="text" value="Cluster-A"/> | Can mix modes<br><br>Must be unique if routable<br><br>Can overlap<br><br>Must be unique<br><br>Must be unique<br><br>Can overlap<br><br>Can overlap | <input checked="" type="checkbox"/> NAT mode<br>Pods IP Block ID *<br><input type="text" value="927a2eff-fd86-4df6-bb21-c45b5314f547"/><br><br>Nodes IP Block ID *<br><input type="text" value="3d577e5c-acaf-4921-945b-d12b0e1318e6"/><br><br>T0 Router ID *<br><input type="text" value="5c579e37-5318-4255-9650-1e2e99ea1d1e9"/><br><br>Floating IP Pool ID *<br><input type="text" value="31a0f4e1-19e7-4122-b300-438d465a486f"/><br><br>Nodes DNS *<br><input type="text" value="10.40.53.1"/><br><br>vSphere Cluster Names *<br><input type="text" value="Cluster-B"/> |
|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |                                                                                                                                                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

# Install Tanzu Kubernetes Grid Integrated Edition on vSphere Using Ops Manager (Antrea and Flannel Networking)

This topic lists the procedures to follow to manually install Tanzu Kubernetes Grid Integrated Edition on vSphere with Antrea or Flannel networking, using Ops Manager.



**Note:** The recommended method for installing Tanzu Kubernetes Grid Integrated Edition on vSphere is to use the Tanzu Kubernetes Grid Integrated Edition Management Console. For information, see [Install on vSphere with the Management Console](#).

To install Tanzu Kubernetes Grid Integrated Edition on vSphere with Flannel networking follow the instructions below:

- [Prerequisites and Resource Requirements](#)
- [VMware Ports and Protocols](#) on the VMware site
- [Creating Dedicated Users and Roles for vSphere \(Optional\)](#)
- [Installing and Configuring Ops Manager on vSphere](#)
- [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#)
- [Configuring a TKGI API Load Balancer](#)
- [Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on vSphere](#)
- [\(Optional\) Integrating VMware Harbor with Tanzu Kubernetes Grid Integrated Edition](#)



**Note:** VMware Harbor is an enterprise-class registry server for container images. For more information, see [VMware Harbor Registry](#) in the *VMware Partner documentation*.

## Install the TKGI and Kubernetes CLIs

The TKGI CLI and Kubernetes CLI help you interact with your Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes clusters and Kubernetes workloads.

To install the CLIs, follow the instructions below:

- [Installing the TKGI CLI](#)
- [Installing the Kubernetes CLI](#)

## vSphere Prerequisites and Resource Requirements

This topic describes the prerequisites and resource requirements for installing VMware Tanzu Kubernetes Grid Integrated Edition on vSphere.

For prerequisites and resource requirements for installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T integration, see [vSphere with NSX-T Version Requirements](#) and [Hardware Requirements for Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T](#).

## Prerequisites

Before installing Tanzu Kubernetes Grid Integrated Edition:

1. Review the sections below and the instructions in [Creating Dedicated Users and Roles for vSphere \(Optional\)](#).
2. Install and configure Ops Manager. To install Ops Manager, follow the instructions in [Installing and Configuring Ops Manager on vSphere](#).

## vSphere Version Requirements

For Tanzu Kubernetes Grid Integrated Edition on vSphere version requirements, refer to the [VMware Product Interoperability Matrices](#).

## Resource Requirements

Installing Ops Manager and Tanzu Kubernetes Grid Integrated Edition requires the following virtual machines (VMs):

| VM            | CPU | Memory (GB) | Ephemeral Disk (GB) |
|---------------|-----|-------------|---------------------|
| BOSH Director | 2   | 8           | 16                  |
| Ops Manager   | 1   | 8           | 160                 |
| TKGI API      | 2   | 8           | 64                  |
| TKGI Database | 2   | 8           | 64                  |



**NOTE:** VMware recommends deploying TKGI on its own dedicated Ops Manager instance, rather than on a shared Ops Manager that also hosts other runtimes such as Tanzu Application Service.

## Storage Requirements for Large Numbers of Pods

If you expect the cluster workload to run a large number of pods continuously, then increase the size of persistent disk storage allocated to the TKGI Database VM as follows:

| Number of Pods | Persistent Disk Requirements (GB) |
|----------------|-----------------------------------|
| 1,000 pods     | 20                                |
| 5,000 pods     | 100                               |
| 10,000 pods    | 200                               |
| 50,000 pods    | 1,000                             |

## Ephemeral VM Resources

Each Tanzu Kubernetes Grid Integrated Edition deployment requires ephemeral VMs during installation and upgrades of Tanzu Kubernetes Grid Integrated Edition. After you deploy Tanzu

Kubernetes Grid Integrated Edition, BOSH automatically deletes these VMs.

To enable Tanzu Kubernetes Grid Integrated Edition to dynamically create the ephemeral VMs when needed, ensure that the following resources are available in your vSphere infrastructure before deploying Tanzu Kubernetes Grid Integrated Edition:

| Ephemeral VM         | VM Count | CPU Cores | Memory (GB) | Ephemeral Disk (GB) |
|----------------------|----------|-----------|-------------|---------------------|
| BOSH Compilation VMs | 4        | 4         | 4           | 32                  |

## Kubernetes Cluster Resources

Each Kubernetes cluster provisioned through Tanzu Kubernetes Grid Integrated Edition deploys the VMs listed below. If you deploy more than one Kubernetes cluster, you must scale your allocated resources appropriately.

| VM                 | VM Count  | CPU Cores | Memory (GB) | Ephemeral Disk (GB) | Persistent Disk (GB) |
|--------------------|-----------|-----------|-------------|---------------------|----------------------|
| Control Plane      | 1 or 3    | 2         | 4           | 8                   | 5                    |
| Worker             | 1 or more | 2         | 4           | 8                   | 50                   |
| Errand (ephemeral) | 1         | 1         | 1           | 8                   | none                 |

## Network Communication Requirements

For a complete list of network communication requirements for vSphere without NSX-T, see [VMware Ports and Protocols](#) on the VMware site.

## Firewall Ports and Protocols Requirements for vSphere (Antrea and Flannel Networking)

This topic describes the firewall ports and protocols requirements for using VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) on vSphere with Antrea and Flannel container networking.

If you are not using TKGI on vSphere Antrea or Flannel container networking, see one of the follow topics instead:

- [Firewall Ports and Protocols Requirements for vSphere with NSX-T](#)
- [Firewall Ports and Protocols Requirements \(Antrea and Flannel Networking\)](#)

## Overview

Apps frequently require the ability to pass internal communication between system components on different networks.

Firewalls and Kubernetes Pod Security Policy are used to filter traffic and limit access in environments with strict inter-network access control policies and your apps require one or more conduits through a secured environment's firewalls.

VMware recommends that rather than using a Kubernetes Pod Security Policy to filter traffic between networks and TKGI system components and clusters that you instead enable access to apps through standard Kubernetes load-balancers and ingress controller types. This enables you to designate specific ports and protocols as a firewall conduit.

Consult the following tables when configuring port settings to install or upgrade TKGI or configure a Kubernetes cluster:

- [TKGI Users Ports and Protocols](#)
- [TKGI Core Ports and Protocols](#)
- [VMware Virtual Infrastructure Ports and Protocols](#)
- [VMware Optional Integration Ports and Protocols](#)



**Note:** To control which groups access deploying and scaling your organization's Tanzu Kubernetes Grid Integrated Edition-deployed Kubernetes clusters, configure your firewall settings as described on the Operator → TKGI API server lines below.

## TKGI Ports and Protocols

The following tables list ports and protocols required for network communications between Tanzu Kubernetes Grid Integrated Edition v1.5.0 and later, and vSphere 6.7 and later.

### TKGI Users Ports and Protocols

The following table lists ports and protocols used for network communication between TKGI user interface components.

| Source Component       | Destination Component | Destination Protocol | Destination Port | Service                |
|------------------------|-----------------------|----------------------|------------------|------------------------|
| Admin/Operator Console | All System Components | TCP                  | 22               | ssh                    |
| Admin/Operator Console | All System Components | TCP                  | 80               | http                   |
| Admin/Operator Console | All System Components | TCP                  | 443              | https                  |
| Admin/Operator Console | BOSH Director         | TCP                  | 25555            | bosh director rest api |
| Admin/Operator Console | Ops Manager           | TCP                  | 22               | ssh                    |
| Admin/Operator Console | Ops Manager           | TCP                  | 443              | https                  |
| Admin/Operator Console | TKGI Controller       | TCP                  | 9021             | tkgi api server        |
| Admin/Operator Console | vCenter Server        | TCP                  | 443              | https                  |
| Admin/Operator Console | vCenter Server        | TCP                  | 5480             | vami                   |

| Source Component                                  | Destination Component                 | Destination Protocol | Destination Port | Service             |
|---------------------------------------------------|---------------------------------------|----------------------|------------------|---------------------|
| Admin/Operator Console                            | vSphere ESXI Hosts Mgmt.<br>vmknic    | TCP                  | 902              | ideafarm-door       |
| Admin/Operator and Developer Consoles             | Harbor Private Image Registry         | TCP                  | 80               | http                |
| Admin/Operator and Developer Consoles             | Harbor Private Image Registry         | TCP                  | 443              | https               |
| Admin/Operator and Developer Consoles             | Harbor Private Image Registry         | TCP                  | 4443             | notary              |
| Admin/Operator and Developer Consoles             | Kubernetes App Load-Balancer Svc      | TCP/UDP              | Varies           | varies with apps    |
| Admin/Operator and Developer Consoles             | Kubernetes Cluster API Server -LB VIP | TCP                  | 8443             | httpsca             |
| Admin/Operator and Developer Consoles             | Kubernetes Cluster Ingress Controller | TCP                  | 80               | http                |
| Admin/Operator and Developer Consoles             | Kubernetes Cluster Ingress Controller | TCP                  | 443              | https               |
| Admin/Operator and Developer Consoles             | Kubernetes Cluster Worker Node        | TCP/UDP              | 30000-32767      | kubernetes nodeport |
| Admin/Operator and Developer Consoles             | TKGI Controller                       | TCP                  | 8443             | httpsca             |
| All User Consoles (Operator, Developer, Consumer) | Kubernetes App Load-Balancer Svc      | TCP/UDP              | Varies           | varies with apps    |
| All User Consoles (Operator, Developer, Consumer) | Kubernetes Cluster Ingress Controller | TCP                  | 80               | http                |
| All User Consoles (Operator, Developer, Consumer) | Kubernetes Cluster Ingress Controller | TCP                  | 443              | https               |
| All User Consoles (Operator, Developer, Consumer) | Kubernetes Cluster Worker Node        | TCP/UDP              | 30000-32767      | kubernetes nodeport |

## TKGI Core Ports and Protocols

The following table lists ports and protocols used for network communication between core TKGI components.

| Source Component      | Destination Component        | Destination Protocol | Destination Port | Service           |
|-----------------------|------------------------------|----------------------|------------------|-------------------|
| All System Components | Corporate Domain Name Server | TCP/UDP              | 53               | dns               |
| All System Components | Network Time Server          | UDP                  | 123              | ntp               |
| All System Components | vRealize LogInsight          | TCP/UDP              | 514/1514         | syslog/tls syslog |

| Source Component                    | Destination Component                      | Destination Protocol | Destination Port | Service                |
|-------------------------------------|--------------------------------------------|----------------------|------------------|------------------------|
| All System Control Plane Components | AD/LDAP Directory Server                   | TCP/UDP              | 389/636          | ldap/ldaps             |
| Ops Manager                         | Admin/Operator Console                     | TCP                  | 22               | ssh                    |
| Ops Manager                         | BOSH Director                              | TCP                  | 6868             | bosh agent http        |
| Ops Manager                         | BOSH Director                              | TCP                  | 8443             | httpsca                |
| Ops Manager                         | BOSH Director                              | TCP                  | 8844             | credhub                |
| Ops Manager                         | BOSH Director                              | TCP                  | 25555            | bosh director rest api |
| Ops Manager                         | Harbor Private Image Registry              | TCP                  | 22               | ssh                    |
| Ops Manager                         | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 22               | ssh                    |
| Ops Manager                         | Kubernetes Cluster Worker Node             | TCP                  | 22               | ssh                    |
| Ops Manager                         | TKGI Controller                            | TCP                  | 22               | ssh                    |
| Ops Manager                         | TKGI Controller                            | TCP                  | 8443             | httpsca                |
| Ops Manager                         | vCenter Server                             | TCP                  | 443              | https                  |
| Ops Manager                         | vSphere ESXI Hosts Mgmt. vmknic            | TCP                  | 443              | https                  |
| BOSH Director                       | vCenter Server                             | TCP                  | 443              | https                  |
| BOSH Director                       | vSphere ESXI Hosts Mgmt. vmknic            | TCP                  | 443              | https                  |
| BOSH Compilation Job VM             | BOSH Director                              | TCP                  | 4222             | bosh nats server       |
| BOSH Compilation Job VM             | BOSH Director                              | TCP                  | 25250            | bosh blobstore         |
| BOSH Compilation Job VM             | BOSH Director                              | TCP                  | 25923            | health monitor daemon  |
| BOSH Compilation Job VM             | Harbor Private Image Registry              | TCP                  | 443              | https                  |
| BOSH Compilation Job VM             | Harbor Private Image Registry              | TCP                  | 8853             | bosh dns health        |
| TKGI Controller                     | BOSH Director                              | TCP                  | 4222             | bosh nats server       |
| TKGI Controller                     | BOSH Director                              | TCP                  | 8443             | httpsca                |
| TKGI Controller                     | BOSH Director                              | TCP                  | 25250            | bosh blobstore         |
| TKGI Controller                     | BOSH Director                              | TCP                  | 25555            | bosh director rest api |
| TKGI Controller                     | BOSH Director                              | TCP                  | 25923            | health monitor daemon  |

| Source Component                           | Destination Component                      | Destination Protocol | Destination Port | Service               |
|--------------------------------------------|--------------------------------------------|----------------------|------------------|-----------------------|
| TKGI Controller                            | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 8443             | httpsca               |
| TKGI Controller                            | TKGI Database VM                           | TCP                  | 3306             | tkgi db proxy         |
| TKGI Controller                            | vCenter Server                             | TCP                  | 443              | https                 |
| Harbor Private Image Registry              | BOSH Director                              | TCP                  | 4222             | bosh nats server      |
| Harbor Private Image Registry              | BOSH Director                              | TCP                  | 25250            | bosh blobstore        |
| Harbor Private Image Registry              | BOSH Director                              | TCP                  | 25923            | health monitor daemon |
| Harbor Private Image Registry              | IP NAS Storage Array                       | TCP                  | 111              | nfs rpc portmapper    |
| Harbor Private Image Registry              | IP NAS Storage Array                       | TCP                  | 2049             | nfs                   |
| Harbor Private Image Registry              | Public CVE Source Database                 | TCP                  | 443              | https                 |
| kube-system pod/telemetry-agent            | TKGI Controller                            | TCP                  | 24224            | fluentd out_forward   |
| Kubernetes Cluster Control Plane/Etcd Node | BOSH Director                              | TCP                  | 4222             | bosh nats server      |
| Kubernetes Cluster Control Plane/Etcd Node | BOSH Director                              | TCP                  | 25250            | bosh blobstore        |
| Kubernetes Cluster Control Plane/Etcd Node | BOSH Director                              | TCP                  | 25923            | health monitor daemon |
| Kubernetes Cluster Control Plane/Etcd Node | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 2379             | etcd client           |
| Kubernetes Cluster Control Plane/Etcd Node | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 2380             | etcd server           |
| Kubernetes Cluster Control Plane/Etcd Node | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 8443             | httpsca               |
| Kubernetes Cluster Control Plane/Etcd Node | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 8853             | bosh dns health       |
| Kubernetes Cluster Control Plane/Etcd Node | Kubernetes Cluster Worker Node             | TCP                  | 4194             | cadvisor              |
| Kubernetes Cluster Control Plane/Etcd Node | Kubernetes Cluster Worker Node             | TCP                  | 10250            | kubelet api           |
| Kubernetes Cluster Control Plane/Etcd Node | Kubernetes Cluster Worker Node             | TCP                  | 31194            | cadvisor              |
| Kubernetes Cluster Control Plane/Etcd Node | TKGI Controller                            | TCP                  | 8443             | httpsca               |
| Kubernetes Cluster Control Plane/Etcd Node | TKGI Controller                            | TCP                  | 8853             | bosh dns health       |

| Source Component                           | Destination Component                      | Destination Protocol | Destination Port | Service               |
|--------------------------------------------|--------------------------------------------|----------------------|------------------|-----------------------|
| Kubernetes Cluster Control Plane/Etcd Node | vCenter Server                             | TCP                  | 443              | https                 |
| Kubernetes Cluster Worker Node             | BOSH Director                              | TCP                  | 4222             | bosh nats server      |
| Kubernetes Cluster Worker Node             | BOSH Director                              | TCP                  | 25250            | bosh blobstore        |
| Kubernetes Cluster Worker Node             | BOSH Director                              | TCP                  | 25923            | health monitor daemon |
| Kubernetes Cluster Worker Node             | Harbor Private Image Registry              | TCP                  | 443              | https                 |
| Kubernetes Cluster Worker Node             | Harbor Private Image Registry              | TCP                  | 8853             | bosh dns health       |
| Kubernetes Cluster Worker Node             | IP NAS Storage Array                       | TCP                  | 111              | nfs rpc portmapper    |
| Kubernetes Cluster Worker Node             | IP NAS Storage Array                       | TCP                  | 2049             | nfs                   |
| Kubernetes Cluster Worker Node             | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 8443             | httpsca               |
| Kubernetes Cluster Worker Node             | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 8853             | bosh dns health       |
| Kubernetes Cluster Worker Node             | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 10250            | kubelet api           |
| pks-system pod/cert-generator              | TKGI Controller                            | TCP                  | 24224            | fluentd out_forward   |
| pks-system pod/fluent-bit                  | TKGI Controller                            | TCP                  | 24224            | fluentd out_forward   |

## VMware Ports and Protocols

The following tables list ports and protocols required for network communication between VMware components. For additional information, see [VMware Ports and Protocols](#).

## VMware Virtual Infrastructure Ports and Protocols

The following table lists ports and protocols used for network communication between VMware virtual infrastructure components.

| Source Component | Destination Component | Destination Protocol | Destination Port | Service |
|------------------|-----------------------|----------------------|------------------|---------|
|------------------|-----------------------|----------------------|------------------|---------|

|                                         |                                      |     |       |                       |
|-----------------------------------------|--------------------------------------|-----|-------|-----------------------|
| vCenter Server                          | vSphere ESXI Hosts Mgmt.<br>vmknic   | TCP | 443   | https                 |
| vCenter Server                          | vSphere ESXI Hosts Mgmt.<br>vmknic   | TCP | 8080  | http alt              |
| vCenter Server                          | vSphere ESXI Hosts Mgmt.<br>vmknic   | TCP | 9080  | io filter storage     |
| vSphere ESXI Hosts Mgmt.<br>vmknic      | vCenter Server                       | UDP | 902   | ideafarm-door         |
| vSphere ESXI Hosts Mgmt.<br>vmknic      | vCenter Server                       | TCP | 9084  | update manager        |
| vSphere ESXI Hosts Mgmt.<br>vmknic      | vSphere ESXI Hosts Mgmt.<br>vmknic   | TCP | 8182  | vsphere ha            |
| vSphere ESXI Hosts Mgmt.<br>vmknic      | vSphere ESXI Hosts Mgmt.<br>vmknic   | UDP | 8182  | vsphere ha            |
| vSphere ESXI Hosts vMotion<br>vmknic    | vSphere ESXI Hosts vMotion<br>vmknic | TCP | 8000  | vmotion               |
| vSphere ESXI Hosts IP Storage<br>vmknic | IP NAS Storage Array                 | TCP | 111   | nfs rpc<br>portmapper |
| vSphere ESXI Hosts IP Storage<br>vmknic | IP NAS Storage Array                 | TCP | 2049  | nfs                   |
| vSphere ESXI Hosts IP Storage<br>vmknic | IP NAS Storage Array                 | TCP | 3260  | iscsi                 |
| vSphere ESXI Hosts vSAN<br>vmknic       | vSphere ESXI Hosts vSAN<br>vmknic    | TCP | 2233  | vsan transport        |
| vSphere ESXI Hosts vSAN<br>vmknic       | vSphere ESXI Hosts vSAN<br>vmknic    | UDP | 12321 | unicast agent         |
| vSphere ESXI Hosts vSAN<br>vmknic       | vSphere ESXI Hosts vSAN<br>vmknic    | UDP | 12345 | vsan cluster svc      |
| vSphere ESXI Hosts vSAN<br>vmknic       | vSphere ESXI Hosts vSAN<br>vmknic    | UDP | 23451 | vsan cluster svc      |
| vSphere ESXI Hosts TEP<br>vmknic        | vSphere ESXI Hosts TEP<br>vmknic     | UDP | 3784  | bfd                   |
| vSphere ESXI Hosts TEP<br>vmknic        | vSphere ESXI Hosts TEP<br>vmknic     | UDP | 3785  | bfd                   |
| vSphere ESXI Hosts TEP<br>vmknic        | vSphere ESXI Hosts TEP<br>vmknic     | UDP | 6081  | geneve                |

## VMware Optional Integration Ports and Protocols

The following table lists ports and protocols used for network communication between optional VMware integrations.

| Source Component                           | Destination Component                 | Destination Protocol | Destination Port | Service             |
|--------------------------------------------|---------------------------------------|----------------------|------------------|---------------------|
| Admin/Operator Console                     | vRealize Operations Manager           | TCP                  | 443              | https               |
| vRealize Operations Manager                | Kubernetes Cluster API Server -LB VIP | TCP                  | 8443             | httpsca             |
| vRealize Operations Manager                | TKGI Controller                       | TCP                  | 8443             | httpsca             |
| vRealize Operations Manager                | Kubernetes Cluster API Server -LB VIP | TCP                  | 8443             | httpsca             |
| Admin/Operator Console                     | vRealize LogInsight                   | TCP                  | 443              | https               |
| Kubernetes Cluster Ingress Controller      | vRealize LogInsight                   | TCP                  | 9000             | ingestion api       |
| Kubernetes Cluster Control Plane/Etcd Node | vRealize LogInsight                   | TCP                  | 9000             | ingestion api       |
| Kubernetes Cluster Control Plane/Etcd Node | vRealize LogInsight                   | TCP                  | 9543             | ingestion api - tls |
| Kubernetes Cluster Worker Node             | vRealize LogInsight                   | TCP                  | 9000             | ingestion api       |
| Kubernetes Cluster Worker Node             | vRealize LogInsight                   | TCP                  | 9543             | ingestion api - tls |
| TKGI Controller                            | vRealize LogInsight                   | TCP                  | 9000             | ingestion api       |
| Admin/Operator and Developer Consoles      | Wavefront SaaS APM                    | TCP                  | 443              | https               |
| kube-system pod/wavefront-proxy            | Wavefront SaaS APM                    | TCP                  | 443              | https               |
| kube-system pod/wavefront-proxy            | Wavefront SaaS APM                    | TCP                  | 8443             | httpsca             |
| pks-system pod/wavefront-collector         | TKGI Controller                       | TCP                  | 24224            | fluentd out_forward |
| Admin/Operator Console                     | vRealize Network Insight Platform     | TCP                  | 443              | https               |
| Admin/Operator Console                     | vRealize Network Insight Proxy        | TCP                  | 22               | ssh                 |
| vRealize Network Insight Proxy             | Kubernetes Cluster API Server -LB VIP | TCP                  | 8443             | httpsca             |
| vRealize Network Insight Proxy             | TKGI Controller                       | TCP                  | 8443             | httpsca             |
| vRealize Network Insight Proxy             | TKGI Controller                       | TCP                  | 9021             | tkgi api server     |

## Creating Dedicated Users and Roles for vSphere (Optional)

This topic describes how to create dedicated users and roles for your vSphere environment before deploying VMware Tanzu Kubernetes Grid Integrated Edition.



**Note:** This topic provides security considerations for defining dedicated vSphere user accounts for use with Kubernetes cluster VMs provisioned by Tanzu Kubernetes Grid Integrated Edition. The information in this topic is only relevant if you **do not** want to use the vSphere administrator account for the Tanzu Kubernetes Grid Integrated Edition and Kubernetes cluster VMs. If you are comfortable using the vSphere administrator account for the TKGI and Kubernetes cluster VMs, skip this topic.

## Overview

Before you install Tanzu Kubernetes Grid Integrated Edition on vSphere, you can prepare your vSphere environment by creating the required user accounts and configuring DNS for the TKGI API endpoint.

You can create the following service accounts in vSphere:

- **Master Node User Account** for the Kubernetes control plane node VMs.
- **BOSH/Ops Manager User Account** for BOSH Director operations.



**WARNING:** The TKGI **Master Node User Account** and BOSH/Ops Manager service accounts must be two separate accounts.

After creating the Master Node and BOSH/Ops Manager service accounts you must grant the accounts privileges in vSphere:

- **Master Node User Account:** Kubernetes control plane node VMs require storage permissions to create load balancers and attach persistent disks to pods. Creating a custom role for this service account allows vSphere to apply the same privileges to all Kubernetes control plane node VMs in your Tanzu Kubernetes Grid Integrated Edition installation.
- **BOSH/Ops Manager User Account:** BOSH Director requires permissions to create VMs. You can apply privileges directly to this service account without creating a role. You can also apply the default [VMware Administrator System Role](#) to this user account to achieve the appropriate permission level.

VMware recommends configuring each service account with the least permissive privileges and unique credentials.



**Note:** If your Kubernetes clusters span multiple vCenters, you must set the user account privileges correctly in each vCenter.

To prepare your vSphere environment, do the following:

1. [Create the Master Node Service Account](#)
2. [Grant Storage Permissions](#)
3. [Create the BOSH/Ops Manager Service Account](#)
4. [Grant Permissions to the BOSH/Ops Manager Service Account](#)
5. [Configure DNS for the TKGI API](#)

## Prerequisites

Before you prepare your vSphere environment, fulfill the prerequisites in [vSphere Prerequisites and Resource Requirements](#).

## Create the Master Node User Account

**Virtual Machine Configuration** privileges control the ability to configure virtual machine options and devices.

1. From the vCenter console, create a user account for Kubernetes cluster control plane VMs.
2. Grant the following **Virtual Machine Object** privileges to the user account:

| Privilege (UI)                           | Privilege (API)                      |
|------------------------------------------|--------------------------------------|
| Virtual Machine > Advanced configuration | VirtualMachine.Config.AdvancedConfig |
| Virtual Machine > Change Settings        | VirtualMachine.Config.Settings       |

## Grant Storage Permissions

Kubernetes control plane node VM user accounts require the following:

- Read access to the folder, host, and datacenter of the cluster node VMs
- Permission to create and delete VMs within the resource pool where Tanzu Kubernetes Grid Integrated Edition is deployed

Grant these permissions to the control plane node user account based on your storage configuration using one of the procedures below:

- [Static Only Persistent Volume Provisioning](#)
- [Dynamic Persistent Volume Provisioning \(with Storage Policy-Based Volume Placement\)](#)
- [Dynamic Persistent Volume Provisioning \(without Storage Policy-Based Volume Placement\)](#)

The procedures in this topic use the following vCenter permissions objects:

- **Virtual Machine Configuration** privileges control the ability to configure virtual machine options and devices. For information about **Virtual Machine Configuration**, see [Virtual Machine Configuration Privileges](#) in the VMware vSphere documentation.
- **Datastore** privileges control the ability to browse, manage, and allocate space on datastores. For information about **Datastore**, see [Datastore Privileges](#) in the VMware vSphere documentation.
- **Resource** privileges control the creation and management of resource pools, as well as the migration of virtual machines. For information about **Resource**, see [Resource Privileges](#) in the VMware vSphere documentation.
- **Storage Views** privileges control privileges for Storage Monitoring Service APIs. **Starting with vSphere 6.0, storage views are deprecated and these privileges no longer apply to them.** For information about **Storage Views**, see [Storage Views Privileges](#) in the VMware vSphere documentation. For more information about vSphere storage configurations, see

[vSphere Storage for Kubernetes](#) in the VMware vSphere documentation.

For information about the vSphere virtual machine permissions API, see [ReconfigVM\\_Task\(reconfigure\)](#) in the *vSphere Web Services API* documentation.

## Static Only Persistent Volume Provisioning

To configure your Kubernetes control plane node user account using static only Persistent Volume (PV) provisioning, do the following:

1. Create a custom role that allows the service account to manage Kubernetes node VMs. For more information about custom roles in vCenter, see [Create a Custom Role](#) in the VMware vSphere documentation.

1. Give this role a name. For example, `manage-k8s-node-vms`.
2. Grant the following privileges at the **VM Folder** level using either the vCenter UI or API:

| Privilege (UI)                         | Privilege (API)                       |
|----------------------------------------|---------------------------------------|
| Virtual Machine > Add existing disk    | VirtualMachine.Config.AddExistingDisk |
| Virtual Machine > Add new disk         | VirtualMachine.Config.AddNewDisk      |
| Virtual Machine > Add or remove device | VirtualMachine.Config.AddRemoveDevice |
| Virtual Machine > Remove disk          | VirtualMachine.Config.RemoveDisk      |

3. Select the **Propagate to Child Objects** checkbox.
2. (Optional) Create a custom role that allows the user account to manage Kubernetes volumes.



**Note:** This role is required if you create a Persistent Volume Claim (PVC) to bind with a statically provisioned PV, and the reclaim policy is set to delete. When the PVC is deleted, the statically provisioned PV is also deleted.

1. Give this role a name. For example, `manage-k8s-volumes`.
2. Grant the following privilege at the **Datastore** level using either the vCenter UI or API:

| Privilege (UI)                        | Privilege (API)          |
|---------------------------------------|--------------------------|
| Datastore > Low level file operations | Datastore.FileManagement |

3. Clear the **Propagate to Child Objects** checkbox.
3. Grant the service account the existing **Read-only** role. This role includes the following privileges at the **vCenter**, **Datacenter**, **Datastore Cluster**, and **Datastore Storage Folder** levels: This role includes the following privileges at the **vCenter**, **Datacenter**, **Datastore Cluster**, and **Datastore Storage Folder** levels:

| Privilege (UI) | Privilege (API)  |
|----------------|------------------|
| Read-only      | System.Anonymous |
|                | System.Read      |

## System.View

4. Continue to [Create the BOSH/Ops Manager User Account](#).

## Dynamic Persistent Volume Provisioning (with Storage Policy-Based Volume Placement)

To configure your Kubernetes control plane node user account using dynamic PV provisioning **with** storage policy-based placement, do the following:

1. Create a custom role that allows the user account to manage Kubernetes node VMs. For more information about custom roles in vCenter, see [Create a Custom Role](#) in the VMware vSphere documentation.

1. Give this role a name. For example, `manage-k8s-node-vms`.
2. Grant the following privileges at the **Cluster**, **Hosts**, and **VM Folder** levels using either the vCenter UI or API:

| Privilege (UI)                                     | Privilege (API)                       |
|----------------------------------------------------|---------------------------------------|
| Resource > Assign virtual machine to resource pool | Resource.AssignVMToPool               |
| Virtual Machine > Add existing disk                | VirtualMachine.Config.AddExistingDisk |
| Virtual Machine > Add new disk                     | VirtualMachine.Config.AddNewDisk      |
| Virtual Machine > Add or remove device             | VirtualMachine.Config.AddRemoveDevice |
| Virtual Machine > Remove disk                      | VirtualMachine.Config.RemoveDisk      |
| Virtual Machine > Create new                       | VirtualMachine.Inventory.Create       |
| Virtual Machine > Remove                           | VirtualMachine.Inventory.Remove       |

3. Select the **Propagate to Child Objects** checkbox.
2. Create a custom role that allows the user account to manage Kubernetes volumes.
  1. Give this role a name. For example, `manage-k8s-volumes`.
  2. Grant the following privileges using either the vCenter UI or API:

| Privilege (UI)                        | Privilege (API)          |
|---------------------------------------|--------------------------|
| Datastore > Allocate space            | Datastore.AllocateSpace  |
| Datastore > Low level file operations | Datastore.FileManagement |

3. Clear the **Propagate to Child Objects** checkbox.
3. Create a custom role that allows the user account to read the Kubernetes storage profile.

1. Give this role a name. For example, `k8s-system-read-and-spbm-profile-view`.
2. Grant the following privilege at the **vCenter** level using either the vCenter UI or API:

| Privilege (UI)              | Privilege (API)     |
|-----------------------------|---------------------|
| Profile-driven storage view | StorageProfile.View |

3. Clear the **Propagate to Child Objects** checkbox.

- Grant the user account the existing **Read-only** role. This role includes the following privileges at the **vCenter**, **Datacenter**, **Datastore Cluster**, and **Datastore Storage Folder** levels:

| Privilege (UI) | Privilege (API)  |
|----------------|------------------|
| Read-only      | System.Anonymous |
|                | System.Read      |
|                | System.View      |

- Continue to [Create the BOSH/Ops Manager Service Account](#).

## Dynamic Volume Provisioning (without Storage Policy-Based Volume Placement)

To configure your Kubernetes control plane node user account using dynamic PV provisioning **without** storage policy-based placement, do the following:

- Create a custom role that allows the user account to manage Kubernetes node VMs. For more information about custom roles in vCenter, see [Create a Custom Role](#) in the VMware vSphere documentation.
  - Give this role a name. For example, `manage-k8s-node-vms`.
  - Grant the following privileges at the **Cluster**, **Hosts**, and **VM Folder** levels using either the vCenter UI or API:
 

| Privilege (UI)                         | Privilege (API)                       |
|----------------------------------------|---------------------------------------|
| Virtual Machine > Add existing disk    | VirtualMachine.Config.AddExistingDisk |
| Virtual Machine > Add new disk         | VirtualMachine.Config.AddNewDisk      |
| Virtual Machine > Add or remove device | VirtualMachine.Config.AddRemoveDevice |
| Virtual Machine > Remove disk          | VirtualMachine.Config.RemoveDisk      |

- Select the **Propagate to Child Objects** checkbox.
- Create a custom role that allows the user account to manage Kubernetes volumes.
  - Give this role a name. For example, `manage-k8s-volumes`.
  - Grant the following privileges using either the vCenter UI or API:
 

| Privilege (UI)                        | Privilege (API)          |
|---------------------------------------|--------------------------|
| Datastore > Allocate space            | Datastore.AllocateSpace  |
| Datastore > Low level file operations | Datastore.FileManagement |
- Clear the **Propagate to Child Objects** checkbox.
- Grant the user account the existing **Read-only** role. This role includes the following privileges at the **vCenter**, **Datacenter**, **Datastore Cluster**, and **Datastore Storage Folder** levels:

| Privilege (UI) | Privilege (API)  |
|----------------|------------------|
| Read-only      | System.Anonymous |
|                | System.Read      |

## Create the BOSH/Ops Manager User Account

1. From the vCenter console, create the BOSH/Ops Manager User Account.
2. If you are deploying both Tanzu Application Service (TAS) and TKGI within the same vSphere environment, create an additional BOSH/Ops Manager Service Account so that you have one account for TAS and a separate account for TKGI.

## Grant Permissions to the BOSH/Ops Manager User Account

There are two options for granting permissions to the BOSH/Ops Manager Service Account(s):

- Grant minimal permissions. Grant each BOSH/Ops Manager User Account the minimum required permissions as described in [vSphere Service Account Requirements](#).
- Grant Administrator Role permissions. Apply the default VMware Administrator Role to each BOSH/Ops Manager Service Account as described in [vCenter Server System Roles](#).



**Warning:** Applying the VMware Administrator Role to the BOSH/Ops Manager Service Account grants the account more privileges than are required. For optimal security always use the least privileged account.

## Configure DNS for the TKGI API

Navigate to your DNS provider and create an entry for a fully qualified domain name (FQDN) within your system domain. For example, `api.tkgi.example.com`.

When you configure the Tanzu Kubernetes Grid Integrated Edition tile, enter this FQDN in the **TKGI API** pane.

After you deploy Tanzu Kubernetes Grid Integrated Edition, you map the IP address of the TKGI API to this FQDN. You can then use this FQDN to access the TKGI API from your local system.

## Next Installation Step

To install and configure Ops Manager, follow the instructions in [Installing and Configuring Ops Manager on vSphere](#).

## Installing and Configuring Ops Manager on vSphere

This topic describes how to install and configure Ops Manager before deploying VMware Tanzu Kubernetes Grid Integrated Edition on vSphere.

## Prerequisites

You use Ops Manager to install and configure Tanzu Kubernetes Grid Integrated Edition. Before you install Ops Manager, review the following prerequisites:

- [vSphere Prerequisites and Resource Requirements](#)
- [VMware Ports and Protocols](#) on the VMware site.
- [Creating Dedicated Users and Roles for vSphere \(Optional\)](#)

## Install and Configure Ops Manager

Each version of Tanzu Kubernetes Grid Integrated Edition is compatible with multiple versions of Ops Manager. To determine Ops Manager compatibility, see [VMware Tanzu Network](#).

To install and configure Ops Manager, follow the instructions in the table below:

| Version           | Instructions                                                                                                                                                           |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Ops Manager v2.9  | <ol style="list-style-type: none"> <li>1. <a href="#">Deploying Ops Manager on vSphere</a></li> <li>2. <a href="#">Configuring BOSH Director on vSphere</a></li> </ol> |
| Ops Manager v2.10 | <ol style="list-style-type: none"> <li>1. <a href="#">Deploying Ops Manager on vSphere</a></li> <li>2. <a href="#">Configuring BOSH Director on vSphere</a></li> </ol> |

## Next Installation Step

To install and configure Tanzu Kubernetes Grid Integrated Edition, follow the instructions in [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#).

## Installing Tanzu Kubernetes Grid Integrated Edition on vSphere (Antrea and Flannel Networking)

This topic describes how to install and configure VMware Tanzu Kubernetes Grid Integrated Edition on vSphere with Antrea or Flannel Networking.

## Prerequisites

Before performing the procedures in this topic, you must have deployed and configured Ops Manager. For more information, see [vSphere Prerequisites and Resource Requirements](#).

If you use an instance of Ops Manager that you configured previously to install other runtimes, perform the following steps before you install Tanzu Kubernetes Grid Integrated Edition:

1. Navigate to Ops Manager.
2. Open the **Director Config** pane.
3. Select the **Enable Post Deploy Scripts** checkbox.
4. Click the **Installation Dashboard** link to return to the Installation Dashboard.
5. Click **Review Pending Changes**. Select all products you intend to deploy and review the changes. For more information, see [Reviewing Pending Product Changes](#).
6. Click **Apply Changes**.

## Step 1: Install Tanzu Kubernetes Grid Integrated Edition

To install Tanzu Kubernetes Grid Integrated Edition, do the following:

1. Download the product file from [VMware Tanzu Network](#).
2. Navigate to <https://YOUR-OPS-MANAGER-FQDN/> in a browser to log in to the Ops Manager Installation Dashboard.
3. Click **Import a Product** to upload the product file.
4. Under **Tanzu Kubernetes Grid Integrated Edition** in the left column, click the plus sign to add this product to your staging area.

## Step 2: Configure Tanzu Kubernetes Grid Integrated Edition

Click the orange **Tanzu Kubernetes Grid Integrated Edition** tile to start the configuration process.

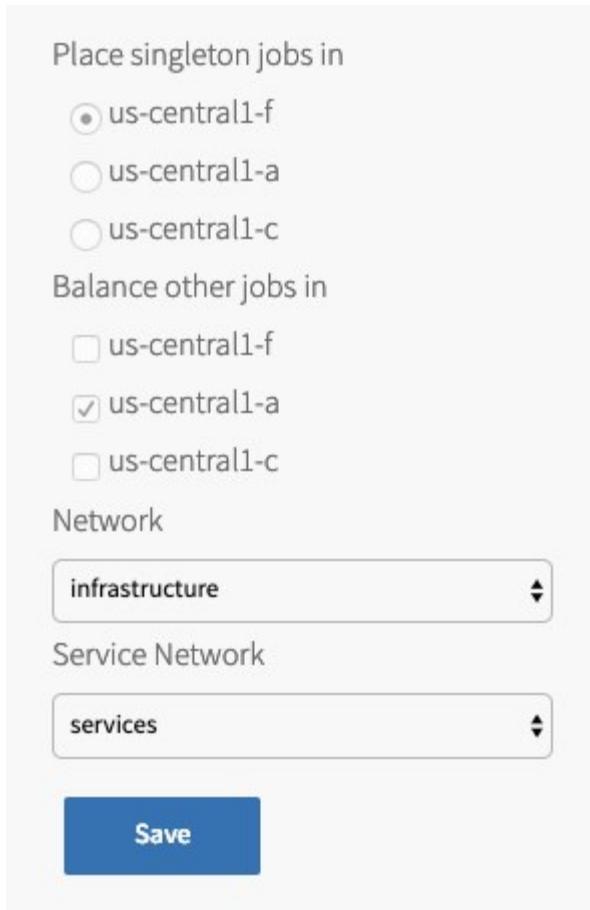


**WARNING:** When you configure the Tanzu Kubernetes Grid Integrated Edition tile, do not use spaces in any field entries. This includes spaces between characters as well as leading and trailing spaces. If you use a space in any field entry, the deployment of Tanzu Kubernetes Grid Integrated Edition fails.

## Assign AZs and Networks

To configure the availability zones (AZs) and networks used by the Tanzu Kubernetes Grid Integrated Edition control plane:

1. Click **Assign AZs and Networks**.
2. Under **Place singleton jobs in**, select the AZ where you want to deploy the TKGI API and TKGI Database.



- Under **Balance other jobs in**, select the AZ for balancing other Tanzu Kubernetes Grid Integrated Edition control plane jobs.



**Note:** You must specify the **Balance other jobs in** AZ, but the selection has no effect in the current version of Tanzu Kubernetes Grid Integrated Edition.

- Under **Network**, select the infrastructure subnet that you created for Tanzu Kubernetes Grid Integrated Edition component VMs, such as the TKGI API and TKGI Database VMs.
- Under **Service Network**, select the services subnet that you created for Kubernetes cluster VMs.
- Click **Save**.

## TKGI API

Perform the following steps:

- Click **TKGI API**.
- Under **Certificate to secure the TKGI API**, provide a certificate and private key pair.

## TKGI API Service

Certificate to secure the TKGI API \*

Certificate PEM

Private Key PEM

[Generate RSA Certificate](#)

API Hostname (FQDN) \*

tkgi.api.example.com

Worker VM Max in Flight \*

4

**Save**

The certificate that you supply must cover the specific subdomain that routes to the TKGI API VM with TLS termination on the ingress. If you use UAA as your OIDC provider, this certificate must be a proper certificate chain and have a SAN field.

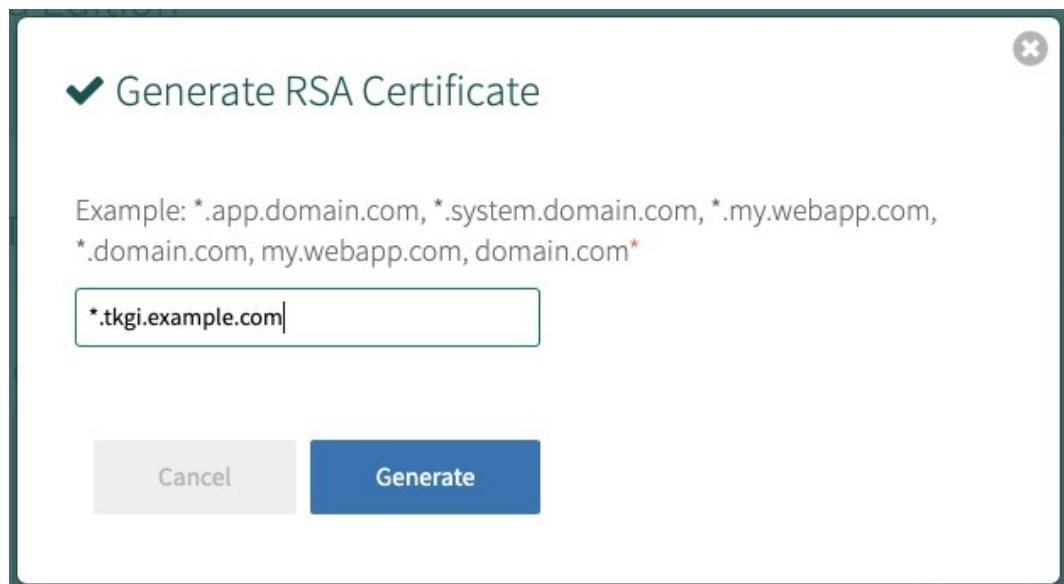


**Warning:** TLS certificates generated for wildcard DNS records only work for a single domain level. For example, a certificate generated for `*.tkgi.EXAMPLE.com` does not permit communication to `*.api.tkgi.EXAMPLE.com`. If the certificate does not contain the correct FQDN for the TKGI API, calls to the API will fail.

You can enter your own certificate and private key pair, or have Ops Manager generate one for you.

To generate a certificate using Ops Manager:

1. Click **Generate RSA Certificate** for a new install or **Change** to update a previously-generated certificate.
2. Enter the domain for your API hostname. This must match the domain you configure under **TKGI API > API Hostname (FQDN)** below, in the same pane. It can be a standard FQDN or a wildcard domain.
3. Click **Generate**.



3. Under **API Hostname (FQDN)**, enter the FQDN that you registered to point to the TKGI API load balancer, such as `api.tkgi.example.com`. To retrieve the public IP address or FQDN of the TKGI API load balancer, log in to your IaaS console.



**Note:** The FQDN for the TKGI API must not contain uppercase letters or trailing whitespace.

4. Under **Worker VM Max in Flight**, enter the maximum number of non-canary worker instances to create or update in parallel within an availability zone.

This field sets the `max_in_flight` variable value. The `max_in_flight` setting limits the number of component instances the TKGI CLI creates or starts simultaneously when running `tkgi create-cluster` or `tkgi update-cluster`. By default, `max_in_flight` is set to `4`, limiting the TKGI CLI to creating or starting a maximum of four component instances in parallel.

5. Click **Save**.

## Plans

A plan defines a set of resource types used for deploying a cluster.

### Activate a Plan

You must first activate and configure **Plan 1**, and afterwards you can activate up to twelve additional, optional, plans.

To activate and configure a plan, perform the following steps:

1. Click the plan that you want to activate.



**Note:** Plans 11, 12, and 13 support Windows worker-based Kubernetes clusters on vSphere with NSX-T, and are a beta feature on vSphere with Flannel. To configure a Windows worker plan see [Plans](#) in *Configuring Windows Worker-Based Kubernetes Clusters* for more information.

2. Select **Active** to activate the plan and make it available to developers deploying clusters.

## Configuration for Plan 1

Select 'Active' to allow users of the PKS CLI to create a cluster using this template plan.

### Plan\*

Active

### Name \*

small

### Description \*

Example: This plan will configure a lightweight kubernetes cluster. Not recommended for production workloads.

### Master/ETCD Node Instances ( min: 1, max: 5 ) \*

1

### Master/ETCD VM Type\*

Automatic: medium.disk (cpu: 2, ram: 4 GB, disk: 32 GB)



### Master Persistent Disk Type\*

Automatic: 10 GB



### Master/ETCD Availability Zones \*

us-central1-f

us-central1-a

us-central1-c

3. Under **Name**, provide a unique name for the plan.
4. Under **Description**, edit the description as needed. The plan description appears in the Services Marketplace, which developers can access by using the TKGI CLI.
5. Under **Master/ETCD Node Instances**, select the default number of Kubernetes control

plane/etcdb nodes to provision for each cluster. You can enter 1, 3, or 5.



**Note:** If you deploy a cluster with multiple control plane/etcdb node VMs, confirm that you have sufficient hardware to handle the increased load on disk write and network traffic. For more information, see [Hardware recommendations](#) in the etcd documentation.

In addition to meeting the hardware requirements for a multi-control plane node cluster, we recommend configuring monitoring for etcd to monitor disk latency, network latency, and other indicators for the health of the cluster. For more information, see [Configuring Telegraf in TKGI](#).



**WARNING:** To change the number of control plane/etcdb nodes for a plan, you must ensure that no existing clusters use the plan. Tanzu Kubernetes Grid Integrated Edition does not support changing the number of control plane/etcdb nodes for plans with existing clusters.

6. Under **Master/ETCD VM Type**, select the type of VM to use for Kubernetes control plane/etcdb nodes. For more information, including control plane node VM customization options, see the [Control Plane Node VM Size](#) section of *VM Sizing for Tanzu Kubernetes Grid Integrated Edition Clusters*.
7. Under **Master Persistent Disk Type**, select the size of the persistent disk for the Kubernetes control plane node VM.
8. Under **Master/ETCD Availability Zones**, select one or more AZs for the Kubernetes clusters deployed by Tanzu Kubernetes Grid Integrated Edition. If you select more than one AZ, Tanzu Kubernetes Grid Integrated Edition deploys the control plane VM in the first AZ and the worker VMs across the remaining AZs. If you are using multiple control plane nodes, Tanzu Kubernetes Grid Integrated Edition deploys the control plane and worker VMs across the AZs in round-robin fashion.



**Note:** Tanzu Kubernetes Grid Integrated Edition does not support changing the AZs of existing control plane nodes.

9. Under **Maximum number of workers on a cluster**, set the maximum number of Kubernetes worker node VMs that Tanzu Kubernetes Grid Integrated Edition can deploy for each cluster. Enter any whole number in this field.

Maximum number of workers on a cluster (min: 1) \*

Worker Node Instances (min: 1) \*

Worker VM Type\*

Automatic: medium.disk (cpu: 2, ram: 4 GB, disk: 32 GB)

Worker Persistent Disk Type\*

Automatic: 50 GB

Worker Availability Zones \*

- us-central1-f
- us-central1-a
- us-central1-c

10. Under **Worker Node Instances**, specify the default number of Kubernetes worker nodes the TKGI CLI provisions for each cluster. The **Worker Node Instances** setting must be less than, or equal to, the **Maximum number of workers on a cluster** setting.

For high availability, create clusters with a minimum of three worker nodes, or two per AZ if you intend to use PersistentVolumes (PVs). For example, if you deploy across three AZs, you should have six worker nodes. For more information about PVs, see [PersistentVolumes in Maintaining Workload Uptime](#). Provisioning a minimum of three worker nodes, or two nodes per AZ is also recommended for stateless workloads.

For more information about creating clusters, see [Creating Clusters](#).



**Note:** Changing a plan's **Worker Node Instances** setting does not alter the number of worker nodes on existing clusters. For information about scaling an existing cluster, see [Scale Horizontally by Changing the Number of Worker Nodes Using the TKGI CLI in Scaling Existing Clusters](#).

11. Under **Worker VM Type**, select the type of VM to use for Kubernetes worker node VMs. For more information, including worker node VM customization options, see [Worker Node VM Number and Size in VM Sizing for Tanzu Kubernetes Grid Integrated Edition Clusters](#).



**Note:** Tanzu Kubernetes Grid Integrated Edition requires a **Worker VM Type** with an ephemeral disk size of 32 GB or more.

12. Under **Worker Persistent Disk Type**, select the size of the persistent disk for the Kubernetes worker node VMs.
13. Under **Worker Availability Zones**, select one or more AZs for the Kubernetes worker nodes. Tanzu Kubernetes Grid Integrated Edition deploys worker nodes equally across the AZs you select.
14. Under **Kubelet customization - system-reserved**, enter resource values that Kubelet can use to reserve resources for system daemons. For example, `memory=250Mi, cpu=150m`. For more information about system-reserved values, see the [Kubernetes documentation](#).

#### Kubelet customization - system-reserved

#### Kubelet customization - eviction-hard

#### Errand VM Type\*

Automatic: t3.micro (cpu: 1, ram: 1 GB, disk: 8 GB) ▾

15. Under **Kubelet customization - eviction-hard**, enter threshold limits that Kubelet can use to evict pods when they exceed the limit. Enter limits in the format `EVICTION-SIGNAL=QUANTITY`. For example, `memory.available=100Mi, nodefs.available=10%, nodefs.inodesFree=5%`. For more information about eviction thresholds, see the [Kubernetes documentation](#).



**WARNING:** Use the Kubelet customization fields with caution. If you enter values that are invalid or that exceed the limits the system supports, Kubelet might fail to start. If Kubelet fails to start, you cannot create clusters.

16. Under **Errand VM Type**, select the size of the VM that contains the errand. The smallest instance possible is sufficient, as the only errand running on this VM is the one that applies the **Default Cluster App** YAML configuration.
17. (Optional) Under **(Optional) Add-ons - Use with caution**, enter additional YAML configuration to add custom workloads to each cluster in this plan. You can specify multiple files using `---` as a separator. For more information, see [Adding Custom Linux Workloads](#).

## (Optional) Add-ons - Use with caution

Allow Privileged

## Admission Plugins

- PodSecurityPolicy
- SecurityContextDeny

18. (Optional) Select the **Allow Privileged** option to allow users to either create Pods with privileged containers or create clusters with resizable persistent volumes using a manually installed vSphere CSI driver. For more information about privileged mode, see [Pods](#) in the Kubernetes documentation.

**Allow Privileged** is not required if clusters use the automatically installed vSphere CSI driver for vSphere CNS. For information about using the automatically installed vSphere CSI driver, see [Storage](#) below and [Deploying Cloud Native Storage \(CNS\) on vSphere](#).



**Note:** Enabling the [Allow Privileged](#) option means that all containers in the cluster will run in privileged mode. [Pod Security Policy](#) provides a [privileged](#) parameter that can be used to activate or deactivate Pods running in privileged mode. As a best practice, if you activate [Allow Privileged](#), define PSP to limit which Pods run in privileged mode. If you are implementing PSP for privileged pods, you must activate [Allow Privileged](#) mode.

19. (Optional) Activate or deactivate one or more admission controller plugins: [PodSecurityPolicy](#) and [SecurityContextDeny](#). For more information see [Using Admission Control Plugins for Tanzu Kubernetes Grid Integrated Edition Clusters](#).



**Note:** To use PodSecurityPolicy features, you must use Ops Manager v2.10.17 or later.

20. (Optional) Under **Node Drain Timeout(mins)**, enter the timeout in minutes for the node to drain pods. If you set this value to [0](#), the node drain does not terminate.

Node Drain Timeout(mins) ( min: 0, max: 1440 )  
0

Pod Shutdown Grace Period (seconds) ( min: -1, max: 86400 )  
10

Force node to drain even if it has running pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or StatefulSet.

Force node to drain even if it has running DaemonSet-managed pods.

Force node to drain even if it has running running pods using emptyDir.

Force node to drain even if pods are still running after timeout.

21. (Optional) Under **Pod Shutdown Grace Period (seconds)**, enter a timeout in seconds for the node to wait before it forces the pod to terminate. If you set this value to `-1`, the default timeout is set to the one specified by the pod.
22. (Optional) To configure when the node drains, activate the following:
  - Force node to drain even if it has running pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or StatefulSet.
  - Force node to drain even if it has running DaemonSet-managed pods.
  - Force node to drain even if it has running running pods using emptyDir.
  - Force node to drain even if pods are still running after timeout.



**Warning:** If you select **Force node to drain even if pods are still running after timeout**, the node halts all running workloads on pods. Before enabling this configuration, set **Node Drain Timeout** to a value greater than `0`.

For more information about configuring default node drain behavior, see [Worker Node Hangs Indefinitely](#) in [Troubleshooting](#).

23. Click **Save**.

## Deactivate a Plan

To deactivate a plan, perform the following steps:

1. Click the plan that you want to deactivate.

2. Select **Inactive**.
3. Click **Save**.

## Kubernetes Cloud Provider

In the procedure below, you use credentials for vCenter master VMs. You must have provisioned the service account with the correct permissions. For more information, see [Create the Master Node Service Account](#) in *Preparing vSphere Before Deploying Tanzu Kubernetes Grid Integrated Edition*.

To configure your Kubernetes cloud provider settings, follow the procedure below:

1. Click **Kubernetes Cloud Provider**.
2. Under **Choose your IaaS**, select **vSphere**.

Choose your IaaS\*

GCP  
 vSphere

vCenter Master Credentials \*

vCenter Host \*

Datacenter Name \*

Datastore Name \*

Stored VM Folder \*

3. Ensure the values in the following procedure match those in the **vCenter Config** section of the Ops Manager tile:
  1. Enter your **vCenter Master Credentials**. Enter the vCenter Server username using the format `user@domainname`, for example: “`user@example.com`”. For more information about the master node service account, see [Preparing vSphere Before](#)

## Deploying Tanzu Kubernetes Grid Integrated Edition.



**Warning:** The vSphere Container Storage Plug-in will not function if you do not specify the domain name for active directory users.

2. Enter your **vCenter Host**. For example, `vcneter-example.com`.



**Note:** The FQDN for the vCenter Server cannot contain uppercase letters.

3. Enter your **Datacenter Name**. For example, `example-dc`.
4. Enter your **Datastore Name**. For example, `example-ds`. Populate **Datastore Name** with the Persistent Datastore name configured in your **BOSH Director** tile under **vCenter Config > Persistent Datastore Names**.

The **Datastore Name** field should contain a single Persistent datastore.



**Note:** The vSphere datastore type must be Datastore. Tanzu Kubernetes Grid Integrated Edition does not support the use of vSphere Datastore Clusters with or without Storage DRS. For more information, see [Datastores and Datastore Clusters](#) in the vSphere documentation.



**Note:** The **Datastore Name** is the default datastore used if the Kubernetes cluster `StorageClass` does not define a `StoragePolicy`. Do not enter a datastore that is a list of BOSH Job/VMDK datastores. For more information, see [PersistentVolume Storage Options on vSphere](#).



**Note:** For multi-AZ and multi-cluster environments, your **Datastore Name** should be a shared Persistent datastore available to each vSphere cluster. Do not enter a datastore that is local to a single cluster. For more information, see [PersistentVolume Storage Options on vSphere](#).

5. Enter the **Stored VM Folder** so that the persistent stores know where to find the VMs. To retrieve the name of the folder, navigate to your BOSH Director tile, click **vCenter Config**, and locate the value for **VM Folder**. The default folder name is `pks_vms`.
4. Click **Save**.

## Networking

To configure networking, do the following:

1. Click **Networking**.

2. Under **Container Networking Interface**, select **Antrea**.

**Networking Configurations**

---

Container Networking Interface\*

Antrea (Switching from Flannel to Antrea is only supported on upgrade)

Kubernetes Pod Network CIDR Range \*

Kubernetes Service Network CIDR Range \*

Flannel (Will be deprecated in TKGI 1.11)

NSX-T

HTTP/HTTPS Proxy (for vSphere and AWS only)\*

Disabled

Enabled

Allow outbound internet access from Kubernetes cluster vms (IaaS-dependent)

Enable outbound internet access

**Save**

Antrea is selected as the default Container Networking Interface (CNI). VMware recommends that you use Antrea as your CNI.

 **Note:** Support for the Flannel Container Networking Interface (CNI) is deprecated. VMware recommends that you switch your Flannel CNI-configured clusters to the Antrea CNI. For more information about Flannel CNI deprecation, see [About Switching from the Flannel CNI to the Antrea CNI](#) in [About Tanzu Kubernetes Grid Integrated Edition Upgrades](#).

3. (Optional) Enter values for **Kubernetes Pod Network CIDR Range** and **Kubernetes Service Network CIDR Range**.
- ◊ Ensure that the CIDR ranges do not overlap and have sufficient space for your deployed services.
  - ◊ Ensure that the CIDR range for the **Kubernetes Pod Network CIDR Range** is large enough to accommodate the expected maximum number of pods.

HTTP/HTTPS Proxy (for vSphere and AWS only)\*

Disabled  
 Enabled

HTTP Proxy URL

HTTP Proxy Credentials  
 Username  
 Password

HTTPS Proxy URL

HTTPS Proxy Credentials  
 Username  
 Password

No Proxy

#### 4. (Optional) Configure Tanzu Kubernetes Grid Integrated Edition to use a proxy.

Production environments can deny direct access to public Internet services and between internal services by placing an HTTP or HTTPS proxy in the network path between Kubernetes nodes and those services.

Configure Tanzu Kubernetes Grid Integrated Edition to use your proxy and activate the following:

- TKGI API access to public Internet services and other internal services.
- Tanzu Kubernetes Grid Integrated Edition-deployed Kubernetes nodes access to public Internet services and other internal services.
- Tanzu Kubernetes Grid Integrated Edition Telemetry ability to forward Telemetry data to the CEIP and Telemetry program.



**Note:** This setting does not set the proxy for running Kubernetes workloads or pods.

#### 5. To complete your global proxy configuration for all outgoing HTTP/HTTPS traffic from your

Kubernetes clusters, perform the following steps:

1. To proxy outgoing HTTP traffic, enter the URL of your HTTP proxy endpoint under **HTTP Proxy URL**. For example, `http://myproxy.com:1234`.
2. (Optional) If your outgoing HTTP proxy uses basic authentication, enter the username and password in the **HTTP Proxy Credentials** fields.
3. To proxy outgoing HTTPS traffic, enter the URL of your HTTP proxy endpoint under **HTTPS Proxy URL**. For example, `http://myproxy.com:1234`.



**Note:** Using an HTTPS connection to the proxy server is not supported. HTTP and HTTPS proxy options can only be configured with an HTTP connection to the proxy server. You cannot populate either of the proxy URL fields with an HTTPS URL. The proxy host and port can be different for HTTP and HTTPS traffic, but the proxy protocol must be HTTP.

4. (Optional) If your HTTPS proxy uses basic authentication, enter the username and password in the **HTTPS Proxy Credentials** fields.
5. Under **No Proxy**, enter the comma-separated list of IP addresses that must bypass the proxy to allow for internal Tanzu Kubernetes Grid Integrated Edition communication.

The **No Proxy** list should include `127.0.0.1` and `localhost`.

Also include the following in the **No Proxy** list:

- Your Tanzu Kubernetes Grid Integrated Edition environment's CIDRs, such as the service network CIDR where your Tanzu Kubernetes Grid Integrated Edition cluster is deployed, the deployment network CIDR, the node network IP block CIDR, and the pod network IP block CIDR.
- The FQDN of any registry, such as the Harbor API FQDN, or component communicating with Tanzu Kubernetes Grid Integrated Edition, using a hostname instead of an IP address.
- The IP addresses for your NSX Manager, vCenter Server, and all ESXi hosts, if you are upgrading and have an existing proxy configuration for reaching a Docker registry or other external services.
- Any additional IP addresses or domain names that should bypass the proxy.

The **No Proxy** property for vSphere accepts wildcard domains denoted by a prefixed `\*` or `..`

For example:

127.0.0.1,localhost, \*.example1.com, .example2.com, example3.com,  
198.51.100.0/24, 203.0.113.0/24, 192.0.2.0/24



**Note:** By default the `10.100.0.0/8` and `10.200.0.0/8` IP address ranges, `.internal`, `.svc`, `.cluster.local`, `.svc.cluster`, and your Tanzu Kubernetes Grid Integrated Edition FQDN are not proxied. This allows internal Tanzu Kubernetes Grid Integrated Edition communication.

Do not use the `_` character in the **No Proxy** field. Entering an underscore character in this field can cause upgrades to fail.

Because some jobs in the VMs accept `\*` as a wildcard, while others only accept `.`, we recommend that you define a wildcard domain using both of them. For example, to denote `example.com` as a wildcard domain, add both `\*.example.com` and `example.com` to the **No Proxy** property.

6. Under **Allow outbound internet access from Kubernetes cluster vms (IaaS-dependent)**, ignore the **Enable outbound internet access** checkbox.
7. Click **Save**.

## UAA

To configure the UAA server:

1. Click **UAA**.
2. Under **TKGI API Access Token Lifetime**, enter a time in seconds for the TKGI API access token lifetime. This field defaults to `600`.

## UAA Configuration

PKS API Access Token Lifetime (in seconds) \*

600

PKS API Refresh Token Lifetime (in seconds) \*

21600

PKS Cluster Access Token Lifetime (in seconds) \*

600

PKS Cluster Refresh Token Lifetime (in seconds) \*

21600

3. Under **TKGI API Refresh Token Lifetime**, enter a time in seconds for the TKGI API refresh token lifetime. This field defaults to **21600**.
4. Under **TKGI Cluster Access Token Lifetime**, enter a time in seconds for the cluster access token lifetime. This field defaults to **600**.
5. Under **TKGI Cluster Refresh Token Lifetime**, enter a time in seconds for the cluster refresh token lifetime. This field defaults to **21600**.



**Note:** VMware recommends using the default UAA token timeout values. By default, access tokens expire after ten minutes and refresh tokens expire after six hours.

6. Under **Configure created clusters to use UAA as the OIDC provider**, select **Enabled** or **Disabled**. This is a global default setting for TKGI-provisioned clusters. For more information, see [OIDC Provider for Kubernetes Clusters](#).

To configure Tanzu Kubernetes Grid Integrated Edition to use UAA as the OIDC provider:

1. Under **Configure created clusters to use UAA as the OIDC provider**, select **Enabled**.

Configure created clusters to use UAA as the OIDC provider.\*

Disabled

Enabled

UAA OIDC Groups Claim \*

UAA OIDC Groups Prefix \*

UAA OIDC Username Claim \*

UAA OIDC Username Prefix \*

TKGI cluster client redirect URIs

Configure your UAA user account store with either internal or external authentication mechanisms \*

Internal UAA

LDAP Server

SAML Identity Provider

**Save**

2. For **UAA OIDC Groups Claim**, enter the name of your groups claim. This is used to set a user's group in the JSON Web Token (JWT) claim. The default value is `roles`.
3. For **UAA OIDC Groups Prefix**, enter a prefix for your groups claim. This prevents conflicts with existing names. For example, if you enter the prefix `oidc:`, UAA creates a group name like `oidc:developers`. The default value is `oidc:.`
4. For **UAA OIDC Username Claim**, enter the name of your username claim. This is used to set a user's username in the JWT claim. The default value is `user_name`. Depending on your provider, you can enter claims besides `user_name`, like `email` or `name`.
5. For **UAA OIDC Username Prefix**, enter a prefix for your username claim. This prevents conflicts with existing names. For example, if you enter the prefix `oidc:`, UAA creates a username like `oidc:admin`. The default value is `oidc:.`



**Warning:** VMware recommends adding OIDC prefixes to prevent users and groups from gaining unintended cluster privileges. If you change the above values for a pre-existing Tanzu Kubernetes Grid Integrated Edition installation, you must change any existing role bindings that bind to a username or group. If you do not change your role bindings, developers cannot access Kubernetes clusters. For instructions, see [Managing Cluster Access and Permissions](#).

7. (Optional) For **TKGI cluster client redirect URIs**, enter one or more comma-delimited UAA redirect URIs. Configure **TKGI cluster client redirect URIs** to assign persistent UAA `cluster_client_redirect_uri` URIs to your clusters. UAA redirect URIs configured in the **TKGI cluster client redirect URIs** field persist through cluster updates and TKGI upgrades.
8. Select one of the following options:
  - ◊ To use an internal user account store for UAA, select **Internal UAA**. Click **Save** and continue to [\(Optional\) Host Monitoring](#).
  - ◊ To use LDAP for UAA, select **LDAP Server** and continue to [Connecting Tanzu Kubernetes Grid Integrated Edition to an LDAP Server](#).
  - ◊ To use SAML for UAA, select **SAML Identity Provider** and continue to [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#).

## (Optional) Host Monitoring

In **Host Monitoring**, you can configure monitoring of nodes and VMs using Syslog, VMware vRealize Log Insight (vRLI) Integration, or Telegraf.

## Configure TKGI Monitoring Features on Host

Enable Syslog for TKGI?\*

No  
 Yes

Enable VMware vRealize Log Insight Integration?\*

No  
 Yes

Enable Telegraf Outputs?\*

No  
 Yes

**Save**

You can configure one or more of the following:

- **Syslog:** To configure Syslog, see [Syslog](#) below. Syslog forwards log messages from all BOSH-deployed VMs to a syslog endpoint.
- **VMware vRealize Log Insight (vRLI) Integration:** To configure VMware vRealize Log Insight (vRLI) Integration, see [VMware vRealize Log Insight Integration](#) below. The vRLI integration pulls logs from all BOSH jobs and containers running in the cluster, including node logs from core Kubernetes and BOSH processes, Kubernetes event logs, and pod `stdout` and `stderr`.
- **Telegraf:** To configure Telegraf, see [Configuring Telegraf in TKGI](#). The Telegraf agent sends metrics from TKGI API, control plane node, and worker node VMs to a monitoring service, such as Wavefront or Datadog.

For more information about these components, see [Monitoring TKGI and TKGI-Provisioned Clusters](#).

### Syslog

To configure Syslog for all BOSH-deployed VMs in Tanzu Kubernetes Grid Integrated Edition:

1. Click **Host Monitoring**.
2. Under **Enable Syslog for TKGI**, select **Yes**.
3. Under **Address**, enter the destination syslog endpoint.

4. Under **Port**, enter the destination syslog port.
5. Under **Transport Protocol**, select a transport protocol for log forwarding.
6. (Optional) To enable TLS encryption during log forwarding, complete the following steps:
  1. Ensure **Enable TLS** is selected.



**Note:** Logs might contain sensitive information, such as cloud provider credentials. VMware recommends that you enable TLS encryption for log forwarding.

2. Under **Permitted Peer**, provide the accepted fingerprint (SHA1) or name of remote peer. For example, `*.YOUR-LOGGING-SYSTEM.com`.
3. Under **TLS Certificate**, provide a TLS certificate for the destination syslog endpoint.



**Note:** You do not need to provide a new certificate if the TLS certificate for the destination syslog endpoint is signed by a Certificate Authority (CA) in your BOSH certificate store.

7. (Optional) Under **Max Message Size**, enter a maximum message size for logs that are forwarded to a syslog endpoint. By default, the **Max Message Size** field is 10,000 characters.
8. (Optional) Under **Custom Rsyslog Configuration**, enter your RSyslog rules configuration using RainerScript syntax. For more information, see [RainerScript](#) in the RSyslog documentation. For example RSyslog rule configurations, see [Example Custom Rules](#) in the Syslog BOSH GitHub repository.
9. Click **Save**.

## VMware vRealize Log Insight Integration



**Note:** Before you configure the vRLI integration, you must have a vRLI license and vRLI must be installed, running, and available in your environment. You need to provide the live instance address during configuration. For instructions and additional information, see the [vRealize Log Insight documentation](#).

By default, vRLI logging is deactivated. To configure vRLI logging:

1. Under **Enable VMware vRealize Log Insight Integration?**, select **Yes**.

Enable VMware vRealize Log Insight Integration?\*

No  
 Yes

Host \*

Enable SSL?

Disable SSL certificate validation

CA certificate

Rate limiting \*

2. Under **Host**, enter the IP address or FQDN of the vRLI host.
3. (Optional) Select the **Enable SSL?** checkbox to encrypt the logs being sent to vRLI using SSL.
4. Choose one of the following SSL certificate validation options:
  - To skip certificate validation for the vRLI host, select the **Disable SSL certificate validation** checkbox. Select this option if you are using a self-signed certificate in order to simplify setup for a development or test environment.
  - To enable certificate validation for the vRLI host, clear the **Disable SSL certificate validation** checkbox.
5. (Optional) If your vRLI certificate is not signed by a trusted CA root or other well known certificate, enter the certificate in the **CA certificate** field. Locate the PEM of the CA used to sign the vRLI certificate, copy the contents of the certificate file, and paste them into the field. Certificates must be in PEM-encoded format.
6. Under **Rate limiting**, enter a time in milliseconds to change the rate at which logs are sent to the vRLI host. The rate limit specifies the minimum time between messages before the



**Note:** Deactivating certificate validation is not recommended for production environments.

fluentd agent begins to drop messages. The default value `0` means that the rate is not limited, which suffices for many deployments.



**Note:** If your deployment is generating a high volume of logs, you can increase this value to limit network traffic. Consider starting with a lower value, such as `10`, then tuning to optimize for your deployment. A large number might result in dropping too many log entries.

7. Click **Save**. These settings apply to any clusters created after you have saved these configuration settings and clicked **Apply Changes**. If the **Upgrade all clusters errand** has been enabled, these settings are also applied to existing clusters.



**Note:** The Tanzu Kubernetes Grid Integrated Edition tile does not validate your vRLI configuration settings. To verify your setup, look for log entries in vRLI.

## (Optional) In-Cluster Monitoring

In **In-Cluster Monitoring**, you can configure one or more observability components and integrations that run in Kubernetes clusters and capture logs and metrics about your workloads. For more information, see [Monitoring Workers and Workloads](#).

## Configure PKS Monitoring Features Deployed in Cluster

Wavefront Integration\*

No

Yes

Deploy cAdvisor\*

No

Yes

Enable Metric Sink Resources

Enable Log Sink Resources

Enable node exporter on workers

**Save**

To configure in-cluster monitoring:

- To configure Wavefront, see [Wavefront](#).
- To configure cAdvisor, see [VMware vRealize Operations Management Pack for Container Monitoring](#).
- To configure sink resources, see:
  - [Metric Sink Resources](#)
  - [Log Sink Resources](#)

You can enable both log and metric sink resources or only one of them.

### Wavefront

You can monitor Kubernetes clusters and pods metrics externally using the integration with [Wavefront by VMware](#).



**Note:** Before you configure Wavefront integration, you must have an active Wavefront account and access to a Wavefront instance. You provide your Wavefront access token during configuration. For additional information, see the [Wavefront](#)

documentation.

To use Wavefront with Windows worker-based clusters, developers must install Wavefront to their clusters manually, using Helm.

To enable and configure Wavefront monitoring:

1. In the Tanzu Kubernetes Grid Integrated Edition tile, select **In-Cluster Monitoring**.
2. Under **Wavefront Integration**, select **Yes**.
3. Under **Wavefront URL**, enter the URL of your Wavefront subscription. For example:

```
https://try.wavefront.com/api
```

4. Under **Wavefront Access Token**, enter the API token for your Wavefront subscription.
5. (Optional) For installations that require a proxy server for outbound Internet access, enable access by entering values for **HTTP Proxy Host**, **HTTP Proxy Port**, **Proxy username**, and **Proxy password**.
6. Click **Save**.

The Tanzu Kubernetes Grid Integrated Edition tile does not validate your Wavefront configuration settings. To verify your setup, look for cluster and pod metrics in Wavefront.

## VMware vRealize Operations Management Pack for Container Monitoring

You can monitor Tanzu Kubernetes Grid Integrated Edition Kubernetes clusters with VMware vRealize Operations Management Pack for Container Monitoring.

To integrate Tanzu Kubernetes Grid Integrated Edition with VMware vRealize Operations Management Pack for Container Monitoring, you must deploy a container running [cAdvisor](#) in your TKGI deployment.

cAdvisor is an open source tool that provides monitoring and statistics for Kubernetes clusters.

To deploy a cAdvisor container:

1. Select **In-Cluster Monitoring**.
2. Under **Deploy cAdvisor**, select **Yes**.
3. Click **Save**.

For more information about integrating this type of monitoring with TKGI, see the [VMware vRealize Operations Management Pack for Container Monitoring User Guide](#) and [Release Notes](#) in the VMware documentation.

## Metric Sink Resources

You can configure TKGI-provisioned clusters to send Kubernetes node metrics and pod metrics to metric sinks. For more information about metric sink resources and what to do after you enable them in the tile, see [Sink Resources in Monitoring Workers and Workloads](#).

To enable clusters to send Kubernetes node metrics and pod metrics to metric sinks:

1. In **In-Cluster Monitoring**, select **Enable Metric Sink Resources**. If you enable this checkbox,

Tanzu Kubernetes Grid Integrated Edition deploys Telegraf as a `DaemonSet`, a pod that runs on each worker node in all your Kubernetes clusters.

2. (Optional) To enable Node Exporter to send worker node metrics to metric sinks of kind `ClusterMetricSink`, select **Enable node exporter on workers**. If you enable this checkbox, Tanzu Kubernetes Grid Integrated Edition deploys Node Exporter as a `DaemonSet`, a pod that runs on each worker node in all your Kubernetes clusters.

For instructions on how to create a metric sink of kind `ClusterMetricSink` for Node Exporter metrics, see [Create a ClusterMetricSink Resource for Node Exporter Metrics](#) in *Creating and Managing Sink Resources*.

3. Click **Save**.

## Log Sink Resources

You can configure TKGI-provisioned clusters to send Kubernetes API events and pod logs to log sinks. For more information about log sink resources and what to do after you enable them in the tile, see [Sink Resources](#) in *Monitoring Workers and Workloads*.

To enable clusters to send Kubernetes API events and pod logs to log sinks:

1. Select **Enable Log Sink Resources**. If you enable this checkbox, Tanzu Kubernetes Grid Integrated Edition deploys Fluent Bit as a `DaemonSet`, a pod that runs on each worker node in all your Kubernetes clusters.
2. Click **Save**.

## Tanzu Mission Control

Tanzu Mission Control integration lets you monitor and manage Tanzu Kubernetes Grid Integrated Edition clusters from the Tanzu Mission Control console, which makes the Tanzu Mission Control console a single point of control for all Kubernetes clusters. For more information about Tanzu Mission Control, see the [VMware Tanzu Mission Control](#) home page.

To integrate Tanzu Kubernetes Grid Integrated Edition with Tanzu Mission Control:

1. Confirm that the TKGI API VM has internet access and can connect to [cna.tmc.cloud.vmware.com](https://cna.tmc.cloud.vmware.com) and the other outbound URLs listed in the [What Happens When You Attach a Cluster](#) section of the Tanzu Mission Control Product documentation.
2. Navigate to the **Tanzu Kubernetes Grid Integrated Edition** tile > the **Tanzu Mission Control** pane and select **Yes** under **Tanzu Mission Control Integration**.

Tanzu Mission Control Integration\*

No  
 Yes

Tanzu Mission Control URL \*

VMware Cloud Services API Token \*

Tanzu Mission Control Cluster Group \*

Tanzu Mission Control Cluster Name Prefix \*

**Save**

3. Configure the fields below:

- **Tanzu Mission Control URL:** Enter the Org URL of your Tanzu Mission Control subscription, without a trailing slash (/). For example, `YOUR-ORG.tmc.cloud.vmware.com`.
- **VMware Cloud Services API token:** Enter your API token to authenticate with VMware Cloud Services APIs. You can retrieve this token by logging in to [VMware Cloud Services](#) and viewing your account information.
- **Tanzu Mission Control Cluster Group:** Enter the name of a Tanzu Mission Control cluster group.

The name can be `default` or another value, depending on your role and access policy:

- `Org Member` users in VMware cloud services have a `service.admin` role in Tanzu Mission Control. These users:
  - By default, can create and attach clusters only in the `default` cluster group.
  - Can create and attach clusters to other cluster groups after an `organization.admin` user grants them the `clustergroup.admin` or

`clustergroup.edit` role for those groups.

- `Org Owner` users in VMware cloud services have `organization.admin` permissions in Tanzu Mission Control. These users:
  - Can create cluster groups.
  - Can grant `clustergroup` roles to `service.admin` users through the Tanzu Mission Control Access Policy view.

For more information about role and access policy, see [Access Control](#) in the VMware Tanzu Mission Control Product documentation.

- ◊ **Tanzu Mission Control Cluster Name Prefix:** Enter a name prefix for identifying the Tanzu Kubernetes Grid Integrated Edition clusters in Tanzu Mission Control.

4. Click **Save**.



**Warning:** After the Tanzu Kubernetes Grid Integrated Edition tile is deployed with a configured cluster group, the cluster group cannot be updated.



**Note:** When you upgrade your Kubernetes clusters and have Tanzu Mission Control integration enabled, existing clusters will be attached to Tanzu Mission Control.

## VMware CEIP

Tanzu Kubernetes Grid Integrated Edition-provisioned clusters send usage data to the TKGI control plane for storage. The VMware Customer Experience Improvement Program (CEIP) provides the option to also send the cluster usage data to VMware to improve customer experience.

To configure Tanzu Kubernetes Grid Integrated Edition CEIP Program settings:

1. Click **CEIP**.
2. Review the information about the CEIP.

## About the CEIP Program

VMware's Customer Experience Improvement Program ("CEIP") provides VMware with information that enables VMware to improve its products and services, to fix problems, and to advise you on how best to deploy and use our products. As part of the CEIP, VMware collects technical information about your organization's use of VMware products and services on a regular basis in association with your organization's VMware license key(s). This information does not personally identify any individual.

Customers who participate in the CEIP receive proactive support benefits that include a weekly report based on telemetry data. Contact your Customer Success Manager to subscribe to this report. You can view a sample report at <https://via.vmware.com/TKGI>.

Additional information regarding the data collected through CEIP and the purposes for which it is used by VMware is set forth in the Trust & Assurance Center at <http://www.vmware.com/trustvmware/ceip.html>. If you prefer not to participate in VMware's CEIP for this product, you should uncheck the box below. You may join or leave VMware's CEIP for this product at any time.

Join the VMware Customer Experience Improvement Program\*

- No
- Yes

Please enter your Entitlement Account Number or your Tanzu Customer Number. We need this information to generate a report for you

Please enter a label for this TKGI Installation

Assign a label to this TKGI Installation for use in your reports

Please select how you will be using this TKGI Installation \*

- Demo or Proof-of-concept
- Development or Pre-production
- Production
- I do not wish to provide this information

**Save**

[View a larger version of this image.](#)

3. If you wish to participate in CEIP, select **Yes**. Otherwise, select **No**.
4. If you selected the **Yes**, complete the following:
  - ◊ (Optional) Enter your entitlement account number or Tanzu customer number. If you are a VMware customer, you can find your entitlement account number in your **Account Summary** on [my.vmware.com](https://my.vmware.com). If you are a Pivotal customer, you can find your Pivotal Customer Number in your Pivotal Order Confirmation email.
  - ◊ (Optional) Enter a descriptive name for your TKGI installation. The label you assign to this installation will be used in CEIP reports to identify the environment.
5. To provide information about the purpose for this installation, select an option.

Please select how you will be using this TKGI Installation \*

- Demo or Proof-of-concept
- Development or Pre-production
- Production
- I do not wish to provide this information

6. Click **Save**.

## Storage

In **Storage Configurations**, you can configure vSphere CNS settings.

## Storage Configurations

vSphere CSI Driver Integration (Enable automatic installation of vSphere CSI driver on all clusters)\*

No

Yes (Warning: Manually deployed vSphere CSI driver must be removed after enabling this feature!)

**Save**

To configure vSphere CNS:

1. Select **Storage**.
2. (Optional) To enable automatic installation of the vSphere CSI driver on all clusters, select **Yes**.



**Warning:** If you have existing clusters with a manually deployed vSphere CSI driver, you must remove the manually deployed driver after enabling this feature. For more information, see [Deploying Cloud Native Storage \(CNS\) on vSphere](#).

## Errands

Errands are scripts that run at designated points during an installation.

To configure which post-deploy and pre-delete errands run for Tanzu Kubernetes Grid Integrated Edition:

1. Make a selection in the dropdown next to each errand.



**Note:** We recommend that you use the default settings for all errands except for the **Run smoke tests** errand.

2. (Optional) Set the **Run smoke tests** errand to **On**.

This errand uses the TKGI CLI to create a Kubernetes cluster and then delete it. If the creation or deletion fails, the errand fails and the installation of the Tanzu Kubernetes Grid Integrated Edition tile is aborted.

3. (Optional) To ensure that all of your cluster VMs are patched, configure the **Upgrade all clusters errand** to **On**.

Updating the Tanzu Kubernetes Grid Integrated Edition tile with a new Linux stemcell and the **Upgrade all clusters errand** enabled triggers the rolling of every Linux VM in each Kubernetes cluster. Similarly, updating the Tanzu Kubernetes Grid Integrated Edition tile with a new Windows stemcell triggers the rolling of every Windows VM in your Kubernetes

clusters.



**Note:** VMware recommends that you review the Tanzu Network metadata and confirm stemcell version compatibility before using the Tanzu Network APIs to update the stemcells in your automated pipeline. For more information, see the [API reference](#).

## Resource Config

To modify the resource configuration of Tanzu Kubernetes Grid Integrated Edition, follow the steps below:

1. Select **Resource Config**.
2. For each job, review the **Automatic** values in the following fields:
  - ◊ **INSTANCES:** Tanzu Kubernetes Grid Integrated Edition defaults to the minimum configuration. If you want a highly available configuration (beta), scale the number of VM instances as follows:
    1. To configure your Tanzu Kubernetes Grid Integrated Edition database for high availability (beta), increase the **INSTANCES** value for **TKGI Database** to 3.
    2. To configure your Tanzu Kubernetes Grid Integrated Edition API and UAA for high availability (beta), increase the **INSTANCES** value for **TKGI API** to 2 or more.



**Warning:** High availability mode is a beta feature. Do not scale your **TKGI API** or **TKGI Database** to more than one instance in production environments.

- ◊ **VM TYPE:** By default, the **TKGI Database** and **TKGI API** jobs are set to the same **Automatic** VM type. If you want to adjust this value, we recommend that you select the same VM type for both jobs.



**Note:** The **Automatic VM TYPE** values match the recommended resource configuration for the **TKGI API** and **TKGI Database** jobs.

- ◊ **PERSISTENT DISK TYPE:** By default, the **TKGI Database** and **TKGI API** jobs are set to the same persistent disk type. If you want to adjust this value, you can change the persistent disk type for each of the jobs independently. Using the same persistent disk type for both jobs is not required.

3. Under each job, leave **NSX-T CONFIGURATION** and **NSX-V CONFIGURATION** blank.



**Warning:** To avoid workload downtime, use the resource configuration recommended in [About Tanzu Kubernetes Grid Integrated Edition Upgrades](#) and [Maintaining Workload Uptime](#).

## Step 3: Apply Changes

1. Return to the Ops Manager Installation Dashboard.
2. Click **Review Pending Changes**. Select the product that you intend to deploy and review the changes. For more information, see [Reviewing Pending Product Changes](#).
3. Click **Apply Changes**.

## Next Installation Step

To configure the TKGI API load balancer, follow the instructions in [Configure TKGI API Load Balancer](#).

## Configuring a TKGI API Load Balancer

This topic describes how to configure an external load balancer for the TKGI API.

### Overview

You must configure an external load balancer to make the TKGI API accessible from outside the network. This external load balancer forwards traffic to the TKGI API endpoint on ports 8443 and 9021. You can use any external load balancer for the TKGI API.

To set up an external load balancer for the TKGI API, do the following after you install the Tanzu Kubernetes Grid Integrated Edition tile:

1. [Retrieve the TKGI API Endpoint](#)
2. [Configure an External Load Balancer](#)

### Prerequisites

Before configuring an external load balancer for the TKGI API, you must have the following:

- The TKGI API certificate that you provided in the **Tanzu Kubernetes Grid Integrated Edition** tile > **TKGI API** > **Certificate to secure the TKGI API**.
- The TKGI API hostname that you entered in the **Tanzu Kubernetes Grid Integrated Edition** tile > **TKGI API** > **API Hostname (FQDN)**.

## Step 1: Retrieve the TKGI API Endpoint

You need to retrieve the TKGI API endpoint to allow your organization to use the API to create, update, and delete Kubernetes clusters.

To retrieve the TKGI API endpoint, do the following:

1. Navigate to the Ops Manager **Installation Dashboard**.
2. Click the **Tanzu Kubernetes Grid Integrated Edition** tile.
3. Click the **Status** tab and locate the **TKGI API** job. The IP address of the TKGI API job is the TKGI API endpoint.

## Step 2: Configure an External Load Balancer

To set up an external load balancer for the TKGI API, configure the external load balancer to resolve to the domain name you entered in the **Tanzu Kubernetes Grid Integrated Edition** tile > **TKGI API** > **API Hostname (FQDN)** using the following information:

- IP address from [Retrieve TKGI API Endpoint](#)
- Ports 8443 and 9021
- HTTPS or TCP protocol

## Next Installation Step

To set up Tanzu Kubernetes Grid Integrated Edition admin users who can create and manage Kubernetes clusters, follow the instructions in [Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on vSphere](#).

## Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on vSphere

This topic describes how to create admin users in VMware Tanzu Kubernetes Grid Integrated Edition with User Account and Authentication (UAA). Creating at least one admin user is a necessary step during the initial set up of Tanzu Kubernetes Grid Integrated Edition.

## Overview

UAA is the identity management service for Tanzu Kubernetes Grid Integrated Edition. Tanzu Kubernetes Grid Integrated Edition includes a UAA server, which is hosted on the TKGI API VM.

To interact with the UAA server, you can use the UAA Command Line Interface (UAAC). You can either run UAAC commands from the Ops Manager VM or install UAAC on your local workstation.

## Prerequisites

Before setting up admin users for Tanzu Kubernetes Grid Integrated Edition, you must have one of the following:

- SSH access to the Ops Manager VM
- A machine that can connect to your TKGI API VM

## Step 1: Connect to the TKGI API VM

You can connect to the TKGI API VM from the Ops Manager VM or from a different machine such as your local workstation.

### Option 1: Connect through the Ops Manager VM

You can connect to TKGI API VM by logging in to the Ops Manager VM through SSH.

To SSH into the Ops Manager VM on vSphere, do the following:

1. Locate the credentials that were used to import the Ops Manager `.ova` or `.ovf` file into your virtualization system. You configured these credentials when you installed Ops Manager and used them to complete the [Prepare vSphere](#) steps in *Deploying Ops Manager on vSphere*.
2. Change the permissions for your private SSH key by running the following command:

```
chmod 600 PRIVATE-KEY
```

Where `PRIVATE-KEY` is the name of your private SSH key.

3. SSH into the Ops Manager VM by running the following command:

```
ssh -i PRIVATE-KEY ubuntu@OPS-MANAGER-FQDN
```

Where `OPS-MANAGER-FQDN` is the fully qualified domain name (FQDN) of Ops Manager.

For example:

```
$ ssh -i id_rsa ubuntu@my-opsmanager-fqdn.example.com
```

4. Proceed to the [Log In as a UAA Admin](#) section to create admin users with UAAC.

## Option 2: Connect through a Non-Ops Manager Machine

To connect to the TKGI API VM and run UAA commands, do the following:

1. Install UAAC on your machine. For example:

```
gem install cf-uaac
```

2. Download a copy of your Ops Manager root CA certificate to the machine. To download the certificate, do the following:
  1. In a web browser, navigate to the FQDN of Ops Manager and log in.
  2. In Ops Manager, navigate to **Settings** in the drop-down menu under your username.
  3. Click **Advanced Options**.
  4. On the **Advanced Options** configuration page, click **Download Root CA Cert**.
  5. Move the certificate to a secure location on your machine and record the path.
3. Proceed to the [Log In as a UAA Admin](#) section to create admin users with UAAC.

## Step 2: Log In as a UAA Admin

Before creating TKGI users, you must log in to the UAA server as a UAA admin. To log in to the UAA server, do the following:

1. Retrieve the UAA management admin client secret:
  1. In a web browser, navigate to the Ops Manager **Installation Dashboard** and click the **Tanzu Kubernetes Grid Integrated Edition** tile.

2. Click the **Credentials** tab.
  3. Click **Link to Credential** next to **Pks Uaa Management Admin Client** and copy the value of **secret**.
2. Target your UAA server by running the following command:

```
uaac target https://TKGI-API:8443 --ca-cert CERTIFICATE-PATH
```

Where:

- ◊ **TKGI-API** is the domain name of your TKGI API server. You entered this domain name in the **Tanzu Kubernetes Grid Integrated Edition** tile > **TKGI API** > **API Hostname (FQDN)**.
- ◊ **CERTIFICATE-PATH** is the path to your Ops Manager root CA certificate. Provide this certificate to validate the TKGI API certificate with SSL.
  - If you are logged in to the Ops Manager VM, specify `/var/tempest/workspaces/default/root_ca_certificate` as the path. This is the default location of the root certificate on the Ops Manager VM.
  - If you downloaded the Ops Manager root CA certificate to your machine, specify the path where you stored the certificate.

For example:

```
$ uaac target api.tkgi.example.com:8443 -ca-cert /var/tempest/workspaces/default/root_ca_certificate
```



**Note:** If you receive an `Unknown key: Max-Age = 86400` warning message, you can ignore it because it has no impact.

3. Authenticate with UAA by running the following command:

```
uaac token client get admin -s ADMIN-CLIENT-SECRET
```

Where **ADMIN-CLIENT-SECRET** is your UAA management admin client secret that you retrieved in a previous step. The client username is **admin**.

## Step 3: Assign Tanzu Kubernetes Grid Integrated Edition Cluster Scopes

The `pks.clusters.manage` and `pks.clusters.admin` UAA scopes grant users the ability to create and manage Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition. For information about UAA scopes in Tanzu Kubernetes Grid Integrated Edition, see [UAA Scopes for Tanzu Kubernetes Grid Integrated Edition Users](#).

To create Tanzu Kubernetes Grid Integrated Edition users with the `pks.clusters.manage` or `pks.clusters.admin` UAA scope, perform one or more of the following procedures based on the

needs of your deployment:

- To assign TKGI cluster scopes to an individual user, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an Individual User](#). Follow this procedure if you selected **Internal UAA** when you configured **UAA** in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#).
- To assign TKGI cluster scopes to an LDAP group, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an External LDAP Group](#). Follow this procedure if you selected **LDAP Server** when you configured **UAA** in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Installing Tanzu Kubernetes Grid Integrated Edition TKGI on vSphere](#).
- To assign TKGI cluster scopes to a SAML group, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an External SAML Group](#). Follow this procedure if you selected **SAML Identity Provider** when you configured **UAA** in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Installing Tanzu Kubernetes Grid Integrated Edition TKGI on vSphere](#).
- To assign TKGI cluster scopes to a client, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to a Client](#).

## Next Step

After you create admin users in Tanzu Kubernetes Grid Integrated Edition, the admin users can create and manage Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition. For more information, see [Managing Kubernetes Clusters and Workloads](#).

## Install Tanzu Kubernetes Grid Integrated Edition on VMware Cloud Foundation

This topic explains how to install and operate Tanzu Kubernetes Grid Integrated Edition (TKGI) on the VMware Cloud Foundation (VCF) platform.

## About TKGI Integration with VCF

VMware Cloud Foundation (VCF) is a unified SDDC platform that brings together vSphere, vSAN, NSX, and vRealize components into an integrated stack to deliver enterprise-ready infrastructure for private and public clouds. For more information, see the [VCF Documentation](#).

You can install Tanzu Kubernetes Grid Integrated Edition (TKGI) v1.9 on VCF v4. You can use either the [TKGI Management Console](#) or [Ops Manager](#) to install TKGI on VCF. The installation procedure is generally the same that you would follow if you were not using the VCF platform, with the requirements and considerations documented here.

## Requirements for Installing TKGI on VCF

To install TKGI on VCF, you must adhere to the following requirements:

- Deploy only the supported versions of TKGI and VCF. See the [Release Notes](#) for precise

version compatibility.

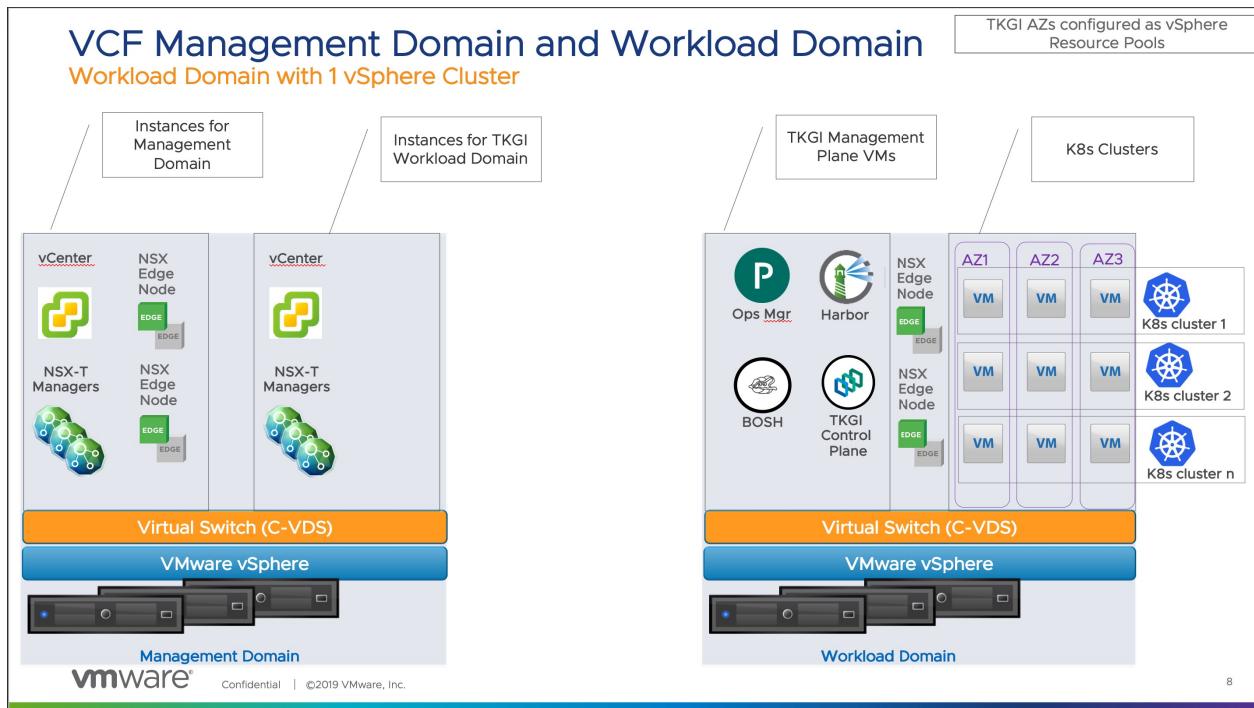
- Deploy vSphere 7.x with NSX-T 3.x using a converged VDS for vSphere and NSX-T traffic. You cannot use an N-VDS for NSX-T transport node traffic. Because of this requirement, a fresh installation is required. There is no way to migrate from N-VDS to VDS.
- Deploy TKGI in the Workload domain only. VCF creates domains, including a Management and Workload domain. TKGI components, including Ops Manager, BOSH Director, TKGI API and DB, and Harbor, must be installed into the Workload domain.
- Deploy TKGI using a single vSphere cluster, if you are using the TKGI Management Console. The reason is that currently the TKGI Management Console does not support using multiple converged VDSes. If you are using Ops Manager, you can use multiple vSphere clusters (with separate VDSes) as long as you are using a shared datastore (vSAN) to support PersistentVolumes.
- When deploying TKGI on VCF, do not generate a new certificate for NSX-T Manager. If you do it will break the VCF to NSX-T Manager communication. You must use the existing NSX-T Manager certificate when configuring TKGI.
- You must deploy the NSX-T Management plane in the VCF Management Domain using a VIP and an external load balancer in front of the NSX Manager nodes. See [Configuring an NSX-T Management Plane Load Balancer](#) for more information.
- You can use the Management Console or Ops Manager to deploy TKGI on VCF, but there is a dependency on the type of [topology](#) you want to deploy.

## Supported Topologies for TKGI on VCF

TKGI supports two topologies for VCF: a Workload Domain with a single vSphere Cluster and a Workload Domain with multiple vSphere clusters.

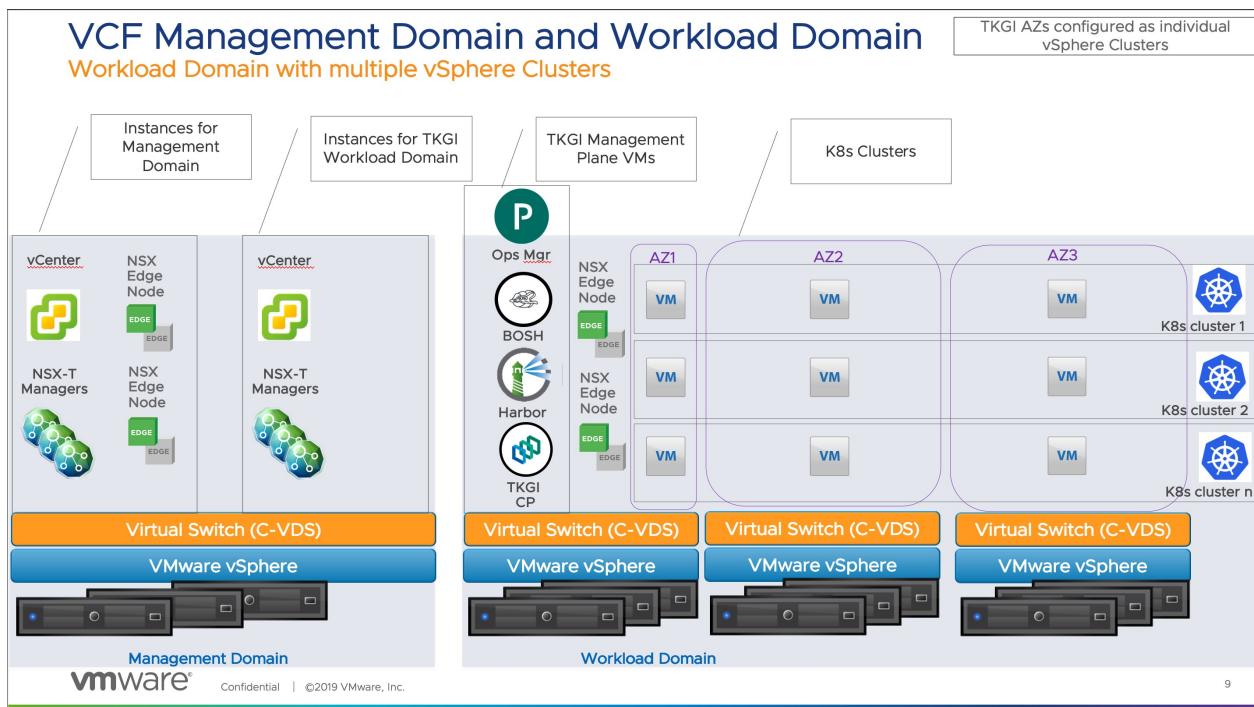
### Topology 1: Workload Domain with a Single vSphere Cluster

This topology provides a single vSphere cluster in the Workload Domain managed by the vCenter Server instance on the Management Domain. This topology is supported by the TKGI Management Console and using Ops Manager to install TKGI.



## Topology 2: Workload Domain with Multiple vSphere Clusters

This topology provides two or more vSphere clusters in the Workload Domain managed by the vCenter instance on the Management Domain. You must use Ops Manager to install TKGI for this topology. You cannot implement this topology using the TKGI Management Console.



## TKGI on VCF Deployment Procedure

This section provides instructions for deploying TKGI once the Management and Workload Domains are created using VCF.

The instructions provided are high-level and assume hands-on experience deploying VCF, NSX-T,

and TKGI. For assistance with installing VCF, see [VMWare Cloud Foundation Deployment Guide](#). For assistance with installing NSX-T 3.0, see [Installing and Configuring NSX-T Data Center v3.0 for Tanzu Kubernetes Grid Integrated Edition](#). For assistance with installing TKGI, see [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#).

1. Install VCF 4.0 and deploy the Management Domain and a single Workload Domain. VCF installs and configures vSphere 7.0 with vSAN and NSX-T 3.0. See the [VMWare Cloud Foundation Deployment Guide](#) for guidance.
2. Create one or two vSphere clusters in the Workload Domain depending on the chosen topology.
3. Log into the NSX-T Management plane in the Workload Domain and create the following NSX-T objects:
  - ◊ Configure a VLAN trunk port group as the network interface for **each** NSX-T Edge Node.
  - ◊ Configure an Uplink Profile for the Edge Nodes.
  - ◊ Deploy two Edge Nodes of size Large VM form factor.
  - ◊ Configure an Edge Cluster comprising the Edge Nodes.
4. If you are using Ops Manager to deploy TKGI, create the following additional NSX-T objects:
  - ◊ Configure a Tier-0 router and add the Edge Node uplink interfaces to get reachability to the physical switches.
  - ◊ Configure an Overlay logical switch for the deployment network.
  - ◊ Configure a Tier-1 router and add the interface to the Overlay logical switch. Redistribute connected routes.
  - ◊ Configure two IP blocks with mask /16: one for cluster nodes and one for cluster pods.
  - ◊ Configure one Floating IP pool for the Load Balancer VIP.
5. In vCenter, create two Resource Pool objects, one for deployment and one for Kubernetes node VMs. These will be used for Availability Zones.
6. Deploy the Ops Manager or TKGI Management Console OVA in the Management Domain on the management network.
7. Configure the tile or installer with the connection information for vCenter in the Workload Domain.
8. Configure the tile or installer with the connection information for the NSX-T Management plane in the Workload Domain.
9. Configure NSX-T networking as follows:
  - ◊ If you are using the Management Console, configure the Edge Cluster, Tier-0 router, IPAMs, and IP pools.
  - ◊ If you are using Ops Manager, provide IP addresses for Edge Node-1 uplink, Edge Node-2 uplink, HA VIP for the uplink, and the default gateway.
  - ◊ Provide deployment network information.
  - ◊ Provide the Node and Pod CIDRs.

- Provide the Load Balancer Floating IP range. NAT IP addresses will be taken from from this pool.
  - Get the Active cluster certificate from NSX-T and provide the same for the NSX certificate.
10. Create Availability Zones.
  11. Select vSAN datastore for Ephemeral and Persistent storage.
  12. Create Kubernetes plans.
  13. Enable integration with vROPs and vRLI.
  14. Enable Harbor. If you are using the Ops Manager installation, deploy Harbor after TKGI.
  15. Deploy TKGI.
  16. Test the TKGI on VCF deployment by installing the TKGI CLI, provisioning a Kubernetes cluster, and deploying a workload.

## Installing Tanzu Kubernetes Grid Integrated Edition on Google Cloud Platform (GCP)

This topic lists the procedures to follow to install VMware Tanzu Kubernetes Grid Integrated Edition on Google Cloud Platform (GCP).



**Note:** Support for GCP is deprecated and will be entirely removed in a future TKGI version.

## Install Tanzu Kubernetes Grid Integrated Edition on GCP

To install Tanzu Kubernetes Grid Integrated Edition on GCP, follow the instructions below:

- [Prerequisites and Resource Requirements](#)
- [Installing and Configuring Ops Manager on GCP](#)
- [Creating Service Accounts in GCP for Tanzu Kubernetes Grid Integrated Edition](#)
- [Creating a GCP Load Balancer for the TKGI API](#)
- [Installing Tanzu Kubernetes Grid Integrated Edition on GCP](#)
- [Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on GCP](#)

## Install the TKGI and Kubernetes CLIs

The TKGI CLI and Kubernetes CLI help you interact with your Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes clusters and Kubernetes workloads.

To install the CLIs, follow the instructions below:

- [Installing the TKGI CLI](#)
- [Installing the Kubernetes CLI](#)

## GCP Prerequisites and Resource Requirements

This topic describes the prerequisites and resource requirements for installing VMware Tanzu Kubernetes Grid Integrated Edition on Google Cloud Platform (GCP).



**Note:** Support for GCP is deprecated and will be entirely removed in a future TKGI version.

### Prerequisites

You can install Tanzu Kubernetes Grid Integrated Edition on GCP manually.

Complete the following before deploying Tanzu Kubernetes Grid Integrated Edition:

1. Review [Resource Requirements](#) below.
2. Install and configure Ops Manager. To install and configure Ops Manager, follow the instructions in [Installing and Configuring Ops Manager on GCP](#).
3. Create service accounts for Kubernetes control plane and worker nodes. To create the service accounts, follow the instructions in [Creating Service Accounts in GCP for Tanzu Kubernetes Grid Integrated Edition](#).



**Note:** Perform this step after you install and configure Ops Manager.

4. Create a load balancer to access the TKGI API from outside the network and run `tkgi` commands from your local workstation. To create a load balancer in GCP, follow the instructions in [Creating a GCP Load Balancer for the TKGI API](#).



**Note:** Perform this step before you install Tanzu Kubernetes Grid Integrated Edition. After you install Tanzu Kubernetes Grid Integrated Edition, you must complete the load balancer configuration. To complete the load balancer configuration, do the procedure in [Create a Network Tag for the Firewall Rule](#).

### Resource Requirements

Installing Ops Manager and Tanzu Kubernetes Grid Integrated Edition requires the following virtual machines (VMs):

| VM            | CPU | Memory (GB) | Ephemeral Disk (GB) |
|---------------|-----|-------------|---------------------|
| BOSH Director | 2   | 8           | 16                  |
| Ops Manager   | 1   | 8           | 160                 |
| TKGI API      | 2   | 8           | 64                  |
| TKGI Database | 2   | 8           | 64                  |



**NOTE:** VMware recommends deploying TKGI on its own dedicated Ops Manager instance, rather than on a shared Ops Manager that also hosts other runtimes such as Tanzu Application Service.

## Storage Requirements for Large Numbers of Pods

If you expect the cluster workload to run a large number of pods continuously, then increase the size of persistent disk storage allocated to the TKGI Database VM as follows:

| Number of Pods | Persistent Disk Requirements (GB) |
|----------------|-----------------------------------|
| 1,000 pods     | 20                                |
| 5,000 pods     | 100                               |
| 10,000 pods    | 200                               |
| 50,000 pods    | 1,000                             |

## Kubernetes Cluster Resources

Each Kubernetes cluster provisioned through Tanzu Kubernetes Grid Integrated Edition deploys the VMs listed below.

If you deploy more than one Kubernetes cluster, you must scale your allocated resources appropriately.

| VM            | VM Count | CPU Cores | Memory (GB) | Ephemeral Disk (GB) | Persistent Disk (GB) |
|---------------|----------|-----------|-------------|---------------------|----------------------|
| Control Plane | 1        | 2         | 4           | 32                  | 5                    |
| Worker        | 1        | 2         | 4           | 32                  | 50                   |

## Installing and Configuring Ops Manager on GCP

This topic describes how to install and configure Ops Manager before deploying VMware Tanzu Kubernetes Grid Integrated Edition on Google Cloud Platform (GCP).



**Note:** Support for GCP is deprecated and will be entirely removed in a future TKGI version.

## Prerequisites

You use Ops Manager to install and configure Tanzu Kubernetes Grid Integrated Edition. Before you install Ops Manager, review [GCP Prerequisites and Resource Requirements](#).

## Install and Configure Ops Manager

Each version of Tanzu Kubernetes Grid Integrated Edition is compatible with multiple versions of Ops Manager. To determine Ops Manager compatibility, see [VMware Tanzu Network](#).

To install and configure Ops Manager, follow the instructions in the table below:

| Version           | Manual Instructions                                                                                                                                                                                                                                          |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Ops Manager v2.10 | <ol style="list-style-type: none"> <li>1. <a href="#">Preparing to Deploy Ops Manager on GCP Manually</a></li> <li>2. <a href="#">Deploying Ops Manager to GCP Manually</a></li> <li>3. <a href="#">Configuring BOSH Director on GCP Manually</a></li> </ol> |

## Next Installation Step

Proceed to [Creating Service Accounts in GCP for Tanzu Kubernetes Grid Integrated Edition](#).

## Creating Service Accounts in GCP for Tanzu Kubernetes Grid Integrated Edition

This topic describes the steps required to create service accounts for VMware Tanzu Kubernetes Grid Integrated Edition on Google Cloud Platform (GCP).



**Note:** Support for GCP is deprecated and will be entirely removed in a future TKGI version.

In order for Kubernetes to create load balancers and attach persistent disks to pods, you must create service accounts with sufficient permissions.

You need separate service accounts for Kubernetes cluster control plane and worker node VMs. VMware recommends configuring each service account with the least permissive privileges and unique credentials.

## Create the Control Plane Node Service Account

1. From the GCP Console, select **IAM & admin > Service accounts**.
2. Click **Create Service Account**.
3. Enter a name for the service account, and add the following roles:
  - **Compute Engine**
    - **Compute Instance Admin (v1)**
    - **Compute Network Admin**
    - **Compute Security Admin**
    - **Compute Storage Admin**
    - **Compute Viewer**
  - **Service Accounts**
    - **Service Account User**
4. Click **Create**.

## Create the Worker Node Service Account

1. From the GCP Console, select **IAM & admin > Service accounts**.
2. Click **Create Service Account**.
3. Enter a name for the service account, and add the **Compute Engine > Compute Viewer** role.
4. Click **Create**.

## Next Installation Step

To create a load balancer in GCP, follow the instructions in [Creating a GCP Load Balancer for the TKGI API](#).

## Creating a GCP Load Balancer for the TKGI API

This topic describes how to create a load balancer for the TKGI API using Google Cloud Platform (GCP).



**Note:** Support for GCP is deprecated and will be entirely removed in a future TKGI version.

## Overview

Before you install VMware Tanzu Kubernetes Grid Integrated Edition, you must configure an external TCP load balancer to access the TKGI API from outside the network. You can use any external TCP load balancer of your choice.

Refer to the procedures in this topic to create a load balancer using GCP. If you choose to use a different load balancer, use the configuration in this topic as a guide.



**Note:** This procedure uses example commands which you should modify to represent the details of your Tanzu Kubernetes Grid Integrated Edition installation.

To create a GCP load balancer for the TKGI API, do the following:

1. [Create a Load Balancer](#)
2. [Create a Firewall Rule](#)
3. [Create a DNS Entry](#)
4. [Install Tanzu Kubernetes Grid Integrated Edition](#)

## Create a Load Balancer

To create a load balancer using GCP, perform the following steps:

1. In a browser, navigate to the [GCP console](#).
2. Navigate to **Network Services > Load balancing** and click **CREATE LOAD BALANCER**.

3. Under **TCP Load Balancing**, click **Start configuration**.
4. Under **Internet facing or internal only**, select **From Internet to my VMs**.
5. Under **Multiple regions or single region**, select **Single region only**.
6. Click **Continue**.
7. Name your load balancer. VMware recommends naming your load balancer `tkgi-api`.
8. Select **Backend configuration**.
  - ◊ Under **Region**, select the region where you deployed Ops Manager.
  - ◊ Under **Backends**, select **Select existing instances**. This will be automatically configured when updating the Resource Config section of the Tanzu Kubernetes Grid Integrated Edition tile.
  - ◊ (Optional) Under **Backup pool**, select a backup pool. If you select a backup pool, set a **Failover ratio**.
  - ◊ (Optional) Under **Health check**, select whether or not you want to create a health check.
  - ◊ Under **Session affinity**, select a session affinity configuration.
  - ◊ (Optional) Select **Advanced configurations** to configure the **Connection draining timeout**.
9. Select **Frontend configuration**.
  - ◊ (Optional) Name your frontend.
  - ◊ (Optional) Click **Add a description** and provide a description.
  - ◊ Select **Create IP address** to reserve an IP address for the TKGI API endpoint.
    1. Enter a name for your reserved IP address. For example, `tkgi-api-ip`. GCP assigns a static IP address that appears next to the name.
    2. (Optional) Enter a description.
    3. Click **Reserve**.
  - ◊ Under **Port**, enter `9021`. Your external load balancer forwards traffic to the TKGI API VM using the UAA endpoint on port 8443 and the TKGI API endpoint on port 9021.
  - ◊ Click **Done**.
  - ◊ Click **New Frontend IP and Port**.
    1. Enter a name for the frontend IP-port mapping, such as `tkgi-api-uaa`.
    2. (Optional) Add a description.
    3. Under **IP** select the same static IP address that GCP assigned in the previous step.
    4. Under **Port**, enter `8443`.
    5. Click **Done**.
10. Click **Review and finalize** to review your load balancer configuration.

11. Click **Create**.

## Create a Firewall Rule

To create a firewall rule that allows traffic between the load balancer and the TKGI API VM, do the following:

1. From the GCP console, navigate to **VPC Network > Firewall rules** and click **CREATE FIREWALL RULE**.
2. Configure the following:
  - Name your firewall rule.
  - (Optional) Provide a description for your firewall rule.
  - Under **Network**, select the VPC network you created in the [Create a GCP Network with Subnets](#) step of *Preparing to Deploy Ops Manager on GCP Manually*.
  - Under **Priority**, enter a priority number between `0` and `65535`.
  - Under **Direction of traffic**, select **Ingress**.
  - Under **Action on match**, select **Allow**.
  - Under **Targets**, select **Specified target tags**.
  - Under **Target tags**, enter the load balancer name. Specify the same load balancer name that you used for **Load Balancer**, for example `tkgi-api`.



**Note:** When deploying the TKGI API VM, Ops Manager will tag the TKGI API VM with the specified load balancer and this firewall rule will then be applied to the TKGI API VM.

3. Click **Create**.
- Under **Source filter**, select **IP ranges**.
  - Under **Source IP ranges**, enter `0.0.0.0/0`.
  - Under **Protocols and ports**, select **Specified protocols and ports** and enter `tcp:8443,9021`.

## Create a DNS Entry

To create a DNS entry in GCP for your TKGI API domain, do the following:

1. From the GCP console, navigate to **Network Services > Cloud DNS**.
2. If you do not already have a DNS zone, click **Create zone**.
  - Provide a **Zone name** and a **DNS name**.
  - Specify whether the **DNSSEC** state of the zone is **Off**, **On**, or **Transfer**.
  - (Optional) Enter a **Description**.
  - Click **Create**.

3. Click **Add record set**.
4. Under **DNS Name**, enter a subdomain for the load balancer. For example, if your domain is `example.com`, enter `api.tkgi` in this field to use `api.tkgi.example.com` as your TKGI API load balancer hostname.
5. Under **Resource Record Type**, select **A** to create a DNS address record.
6. Enter a value for **TTL** and select a **TTL Unit**.
7. Enter the static IP address that GCP assigned when you created the load balancer in [Create a Load Balancer](#).
8. Click **Create**.

## Install Tanzu Kubernetes Grid Integrated Edition

Follow the instructions in [Installing Tanzu Kubernetes Grid Integrated Edition on GCP](#) to deploy Tanzu Kubernetes Grid Integrated Edition.

### Installing Tanzu Kubernetes Grid Integrated Edition on GCP (Antrea and Flannel Networking)

This topic describes how to install and configure VMware Tanzu Kubernetes Grid Integrated Edition on Google Cloud Platform (GCP).



**Note:** Support for GCP is deprecated and will be entirely removed in a future TKGI version.

## Prerequisites

Before performing the procedures in this topic, you must have deployed and configured Ops Manager. For more information, see [GCP Prerequisites and Resource Requirements](#).

If you use an instance of Ops Manager that you configured previously to install other runtimes, perform the following steps before you install Tanzu Kubernetes Grid Integrated Edition:

1. Navigate to Ops Manager.
2. Open the **Director Config** pane.
3. Select the **Enable Post Deploy Scripts** checkbox.
4. Click the **Installation Dashboard** link to return to the Installation Dashboard.
5. Click **Review Pending Changes**. Select all products you intend to deploy and review the changes. For more information, see [Reviewing Pending Product Changes](#).
6. Click **Apply Changes**.

## Step 1: Install Tanzu Kubernetes Grid Integrated Edition

To install Tanzu Kubernetes Grid Integrated Edition, do the following:

1. Download the product file from [VMware Tanzu Network](#).
2. Navigate to <https://YOUR-OPS-MANAGER-FQDN/> in a browser to log in to the Ops Manager Installation Dashboard.
3. Click **Import a Product** to upload the product file.
4. Under **Tanzu Kubernetes Grid Integrated Edition** in the left column, click the plus sign to add this product to your staging area.

## Step 2: Configure Tanzu Kubernetes Grid Integrated Edition

Click the orange **Tanzu Kubernetes Grid Integrated Edition** tile to start the configuration process.

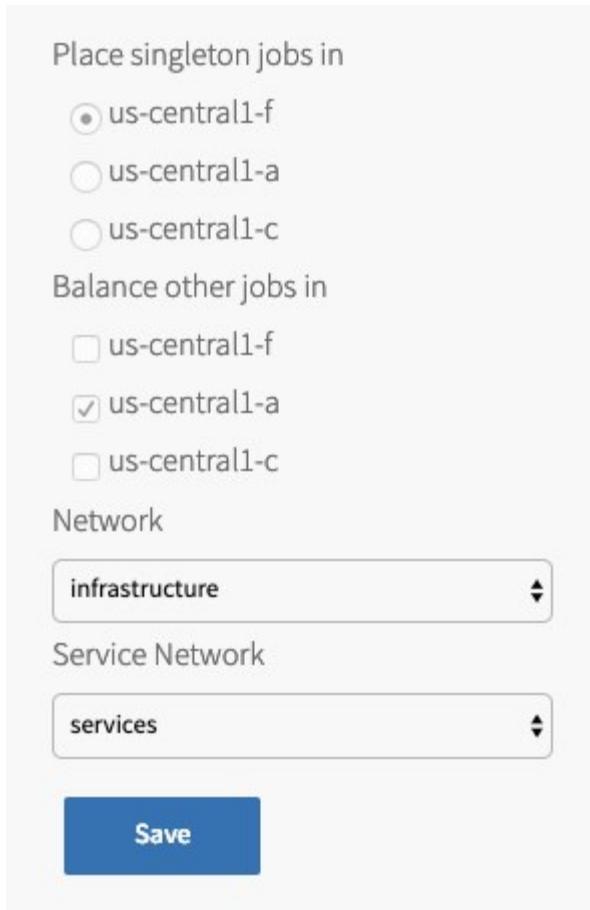


**Warning:** When you configure the Tanzu Kubernetes Grid Integrated Edition tile, do not use spaces in any field entries. This includes spaces between characters as well as leading and trailing spaces. If you use a space in any field entry, the deployment of Tanzu Kubernetes Grid Integrated Edition fails.

## Assign AZs and Networks

To configure the availability zones (AZs) and networks used by the Tanzu Kubernetes Grid Integrated Edition control plane:

1. Click **Assign AZs and Networks**.
2. Under **Place singleton jobs in**, select the AZ where you want to deploy the TKGI API and TKGI Database.



- Under **Balance other jobs in**, select the AZ for balancing other Tanzu Kubernetes Grid Integrated Edition control plane jobs.



**Note:** You must specify the **Balance other jobs in** AZ, but the selection has no effect in the current version of Tanzu Kubernetes Grid Integrated Edition.

- Under **Network**, select the infrastructure subnet that you created for Tanzu Kubernetes Grid Integrated Edition component VMs, such as the TKGI API and TKGI Database VMs.
- Under **Service Network**, select the services subnet that you created for Kubernetes cluster VMs.
- Click **Save**.

## TKGI API

Perform the following steps:

- Click **TKGI API**.
- Under **Certificate to secure the TKGI API**, provide a certificate and private key pair.

## TKGI API Service

Certificate to secure the TKGI API \*

Certificate PEM

Private Key PEM

[Generate RSA Certificate](#)

API Hostname (FQDN) \*

tkgi.api.example.com

Worker VM Max in Flight \*

4

**Save**

The certificate that you supply must cover the specific subdomain that routes to the TKGI API VM with TLS termination on the ingress. If you use UAA as your OIDC provider, this certificate must be a proper certificate chain and have a SAN field.

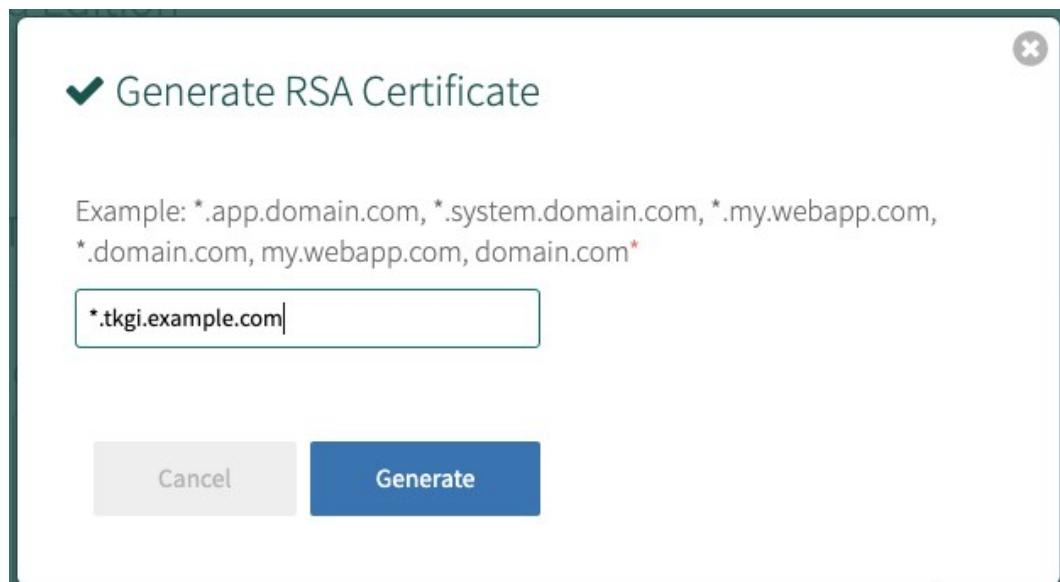


**Warning:** TLS certificates generated for wildcard DNS records only work for a single domain level. For example, a certificate generated for `*.tkgi.EXAMPLE.com` does not permit communication to `*.api.tkgi.EXAMPLE.com`. If the certificate does not contain the correct FQDN for the TKGI API, calls to the API will fail.

You can enter your own certificate and private key pair, or have Ops Manager generate one for you.

To generate a certificate using Ops Manager:

1. Click **Generate RSA Certificate** for a new install or **Change** to update a previously-generated certificate.
2. Enter the domain for your API hostname. This must match the domain you configure under **TKGI API > API Hostname (FQDN)** below, in the same pane. It can be a standard FQDN or a wildcard domain.
3. Click **Generate**.



- Note:** If you deployed a global HTTP load balancer for Ops Manager without a certificate, you can configure the load balancer to use this newly-generated certificate. To configure your Ops Manager load balancer front end certificate, see [Configure Front End](#) in *Preparing to Deploy Ops Manager on GCP Manually*.
3. Under **API Hostname (FQDN)**, enter the FQDN that you registered to point to the TKGI API load balancer, such as `api.tkgi.example.com`. To retrieve the public IP address or FQDN of the TKGI API load balancer, log in to your IaaS console.



**Note:** The FQDN for the TKGI API must not contain uppercase letters or trailing whitespace.

4. Under **Worker VM Max in Flight**, enter the maximum number of non-canary worker instances to create or update in parallel within an availability zone.

This field sets the `max_in_flight` variable value. The `max_in_flight` setting limits the number of component instances the TKGI CLI creates or starts simultaneously when running `tkgi create-cluster` or `tkgi update-cluster`. By default, `max_in_flight` is set to `4`, limiting the TKGI CLI to creating or starting a maximum of four component instances in parallel.

5. Click **Save**.

## Plans

A plan defines a set of resource types used for deploying a cluster.

## Activate a Plan

You must first activate and configure **Plan 1**, and afterwards you can optionally activate **Plan 2** through **Plan 10**.

To activate and configure a plan, perform the following steps:

1. Click the plan that you want to activate.



**Note:** Plans 11, 12, and 13 support Windows worker-based Kubernetes clusters on vSphere with NSX-T, and are a beta feature on vSphere with Flannel.

2. Select **Active** to activate the plan and make it available to developers deploying clusters.

## Configuration for Plan 1

Select 'Active' to allow users of the PKS CLI to create a cluster using this template plan.

Plan\*

Active

Name \*

small

Description \*

Example: This plan will configure a lightweight kubernetes cluster. Not recommended for production workloads.

Master/ETCD Node Instances ( min: 1, max: 5 ) \*

1

Master/ETCD VM Type\*

Automatic: medium.disk (cpu: 2, ram: 4 GB, disk: 32 GB)

Master Persistent Disk Type\*

Automatic: 10 GB

Master/ETCD Availability Zones \*

us-central1-f

us-central1-a

us-central1-c

3. Under **Name**, provide a unique name for the plan.
4. Under **Description**, edit the description as needed. The plan description appears in the Services Marketplace, which developers can access by using the TKGI CLI.
5. Under **Master/ETCD Node Instances**, select the default number of Kubernetes control

plane/etcdb nodes to provision for each cluster. You can enter 1, 3, or 5.



**Note:** If you deploy a cluster with multiple control plane/etcdb node VMs, confirm that you have sufficient hardware to handle the increased load on disk write and network traffic. For more information, see [Hardware recommendations](#) in the etcd documentation.

In addition to meeting the hardware requirements for a multi-control plane node cluster, we recommend configuring monitoring for etcd to monitor disk latency, network latency, and other indicators for the health of the cluster. For more information, see [Configuring Telegraf in TKGI](#).



**WARNING:** To change the number of control plane/etcdb nodes for a plan, you must ensure that no existing clusters use the plan. Tanzu Kubernetes Grid Integrated Edition does not support changing the number of control plane/etcdb nodes for plans with existing clusters.

6. Under **Master/ETCD VM Type**, select the type of VM to use for Kubernetes control plane/etcdb nodes. For more information, including control plane node VM customization options, see the [Control Plane Node VM Size](#) section of *VM Sizing for Tanzu Kubernetes Grid Integrated Edition Clusters*.
7. Under **Master Persistent Disk Type**, select the size of the persistent disk for the Kubernetes control plane node VM.
8. Under **Master/ETCD Availability Zones**, select one or more AZs for the Kubernetes clusters deployed by Tanzu Kubernetes Grid Integrated Edition. If you select more than one AZ, Tanzu Kubernetes Grid Integrated Edition deploys the control plane VM in the first AZ and the worker VMs across the remaining AZs. If you are using multiple control plane nodes, Tanzu Kubernetes Grid Integrated Edition deploys the control plane and worker VMs across the AZs in round-robin fashion.



**Note:** Tanzu Kubernetes Grid Integrated Edition does not support changing the AZs of existing control plane nodes.

9. Under **Maximum number of workers on a cluster**, set the maximum number of Kubernetes worker node VMs that Tanzu Kubernetes Grid Integrated Edition can deploy for each cluster. Enter any whole number in this field.

Maximum number of workers on a cluster (min: 1) \*

Worker Node Instances (min: 1) \*

Worker VM Type\*

Automatic: medium.disk (cpu: 2, ram: 4 GB, disk: 32 GB)

Worker Persistent Disk Type\*

Automatic: 50 GB

Worker Availability Zones \*

us-central1-f

us-central1-a

us-central1-c

10. Under **Worker Node Instances**, specify the default number of Kubernetes worker nodes the TKGI CLI provisions for each cluster. The **Worker Node Instances** setting must be less than, or equal to, the **Maximum number of workers on a cluster** setting.

For high availability, create clusters with a minimum of three worker nodes, or two per AZ if you intend to use PersistentVolumes (PVs). For example, if you deploy across three AZs, you should have six worker nodes. For more information about PVs, see [PersistentVolumes in Maintaining Workload Uptime](#). Provisioning a minimum of three worker nodes, or two nodes per AZ is also recommended for stateless workloads.

For more information about creating clusters, see [Creating Clusters](#).



**Note:** Changing a plan's **Worker Node Instances** setting does not alter the number of worker nodes on existing clusters. For information about scaling an existing cluster, see [Scale Horizontally by Changing the Number of Worker Nodes Using the TKGI CLI in Scaling Existing Clusters](#).

11. Under **Worker VM Type**, select the type of VM to use for Kubernetes worker node VMs. For more information, including worker node VM customization options, see [Worker Node VM Number and Size in VM Sizing for Tanzu Kubernetes Grid Integrated Edition Clusters](#).
12. Under **Worker Persistent Disk Type**, select the size of the persistent disk for the Kubernetes worker node VMs.
13. Under **Worker Availability Zones**, select one or more AZs for the Kubernetes worker

nodes. Tanzu Kubernetes Grid Integrated Edition deploys worker nodes equally across the AZs you select.

14. Under **Kubelet customization - system-reserved**, enter resource values that Kubelet can use to reserve resources for system daemons. For example, `memory=250Mi, cpu=150m`. For more information about system-reserved values, see the [Kubernetes documentation](#).

#### Kubelet customization - system-reserved

#### Kubelet customization - eviction-hard

#### Errand VM Type\*

Automatic: t3.micro (cpu: 1, ram: 1 GB, disk: 8 GB) 

15. Under **Kubelet customization - eviction-hard**, enter threshold limits that Kubelet can use to evict pods when they exceed the limit. Enter limits in the format `EVICTION-SIGNAL=QUANTITY`. For example, `memory.available=100Mi, nodefs.available=10%, nodefs.inodesFree=5%`. For more information about eviction thresholds, see the [Kubernetes documentation](#).



**WARNING:** Use the Kubelet customization fields with caution. If you enter values that are invalid or that exceed the limits the system supports, Kubelet might fail to start. If Kubelet fails to start, you cannot create clusters.

16. Under **Errand VM Type**, select the size of the VM that contains the errand. The smallest instance possible is sufficient, as the only errand running on this VM is the one that applies the **Default Cluster App** YAML configuration.
17. (Optional) Under **(Optional) Add-ons - Use with caution**, enter additional YAML configuration to add custom workloads to each cluster in this plan. You can specify multiple files using `---` as a separator. For more information, see [Adding Custom Linux Workloads](#).

## (Optional) Add-ons - Use with caution

Allow Privileged

## Admission Plugins

- PodSecurityPolicy
- SecurityContextDeny

18. (Optional) To allow users to create pods with privileged containers, select the **Allow Privileged** option. For more information, see [Pods](#) in the Kubernetes documentation.



**Note:** Enabling the [Allow Privileged](#) option means that all containers in the cluster will run in privileged mode. [Pod Security Policy](#) provides a **privileged** parameter that can be used to activate or deactivate Pods running in privileged mode. As a best practice, if you activate [Allow Privileged](#), define PSP to limit which Pods run in privileged mode. If you are implementing PSP for privileged pods, you must activate [Allow Privileged](#) mode.

19. (Optional) Activate or deactivate one or more admission controller plugins: **PodSecurityPolicy** and **SecurityContextDeny**. For more information see [Using Admission Control Plugins for Tanzu Kubernetes Grid Integrated Edition Clusters](#).



**Note:** To use PodSecurityPolicy features, you must use Ops Manager v2.10.17 or later.

20. (Optional) Under **Node Drain Timeout(mins)**, enter the timeout in minutes for the node to drain pods. If you set this value to [0](#), the node drain does not terminate.

Node Drain Timeout(mins) ( min: 0, max: 1440 )

Pod Shutdown Grace Period (seconds) ( min: -1, max: 86400 )

Force node to drain even if it has running pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or StatefulSet.

Force node to drain even if it has running DaemonSet-managed pods.

Force node to drain even if it has running running pods using emptyDir.

Force node to drain even if pods are still running after timeout.

21. (Optional) Under **Pod Shutdown Grace Period (seconds)**, enter a timeout in seconds for the node to wait before it forces the pod to terminate. If you set this value to `-1`, the default timeout is set to the one specified by the pod.
22. (Optional) To configure when the node drains, activate the following:
  - Force node to drain even if it has running pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or StatefulSet.
  - Force node to drain even if it has running DaemonSet-managed pods.
  - Force node to drain even if it has running running pods using emptyDir.
  - Force node to drain even if pods are still running after timeout.



**Warning:** If you select **Force node to drain even if pods are still running after timeout**, the node halts all running workloads on pods. Before enabling this configuration, set **Node Drain Timeout** to a value greater than `0`.

For more information about configuring default node drain behavior, see [Worker Node Hangs Indefinitely](#) in [Troubleshooting](#).

23. Click **Save**.

## Deactivate a Plan

To deactivate a plan, perform the following steps:

1. Click the plan that you want to deactivate.

2. Select **Inactive**.
3. Click **Save**.

## Kubernetes Cloud Provider

To configure your Kubernetes cloud provider settings, follow the procedures below:

1. Click **Kubernetes Cloud Provider**.
2. Under **Choose your IaaS**, select **GCP**.
3. Ensure the values in the following procedure match those in the **Google Config** section of the Ops Manager tile as follows:

Choose your IaaS\*

GCP

GCP Project ID \*

VPC Network \*

GCP Master Service Account ID \*

GCP Worker Service Account ID \*

GCP Subnetwork

vSphere  
 AWS  
 Azure

**Save**

1. Enter your **GCP Project ID**, which is the name of the deployment in your Ops Manager environment. To find the project ID, go to **BOSH Director for GCP > Google Config > Project ID**.
2. Enter your **VPC Network**, which is the VPC network name for your Ops Manager environment.
3. Enter your **GCP Master Service Account ID**. This is the email address associated with the control plane node service account. You configured the control plane node service account in [Create the Control Plane Node Service Account](#) in *Creating*

*Service Accounts in GCP for Tanzu Kubernetes Grid Integrated Edition.*

4. Enter your **GCP Worker Service Account ID**. This is the email address associated with the worker node service account. You configured the worker node service account in [Create the Worker Node Service Account](#) in *Creating Service Accounts in GCP for Tanzu Kubernetes Grid Integrated Edition*.
5. (Optional) Enter your **GCP Subnetwork**. This is the name of the services subnetwork that you created for Kubernetes cluster VMs in GCP.



**Note:** If you want to create GCP internal load balancers through Services of type `LoadBalancer`, you must configure the **GCP Subnetwork** field.

4. Click **Save**.

## Networking

To configure networking, do the following:

1. Click **Networking**.
2. Under **Container Networking Interface**, select **Flannel**.

### Networking Configurations

---

Container Networking Interface\*

Flannel

Kubernetes Pod Network CIDR Range \*

10.200.0.0/16

Kubernetes Service Network CIDR Range \*

10.100.200.0/24

NSX-T

HTTP/HTTPS Proxy (for vSphere only)\*

Disabled

Enabled

Allow outbound internet access from Kubernetes cluster vms (IaaS-dependent)

Enable outbound internet access

**Save**

3. (Optional) Enter values for **Kubernetes Pod Network CIDR Range** and **Kubernetes Service Network CIDR Range**.
  - ◊ Ensure that the CIDR ranges do not overlap and have sufficient space for your deployed services.
  - ◊ Ensure that the CIDR range for the **Kubernetes Pod Network CIDR Range** is large enough to accommodate the expected maximum number of pods.
4. (Optional) If you do not use a NAT instance, select **Allow outbound internet access from Kubernetes cluster vms (IaaS-dependent)**. Enabling this functionality assigns external IP addresses to VMs in clusters.
5. Click **Save**.

## UAA

To configure the UAA server:

1. Click **UAA**.
2. Under **TKGI API Access Token Lifetime**, enter a time in seconds for the TKGI API access token lifetime. This field defaults to **600**.

The screenshot shows a configuration interface titled "UAA Configuration". It contains four input fields, each with a placeholder value and a red asterisk indicating it is required. The fields are arranged vertically:

- "PKS API Access Token Lifetime (in seconds)" with value "600"
- "PKS API Refresh Token Lifetime (in seconds)" with value "21600"
- "PKS Cluster Access Token Lifetime (in seconds)" with value "600"
- "PKS Cluster Refresh Token Lifetime (in seconds)" with value "21600"

3. Under **TKGI API Refresh Token Lifetime**, enter a time in seconds for the TKGI API refresh token lifetime. This field defaults to **21600**.
4. Under **TKGI Cluster Access Token Lifetime**, enter a time in seconds for the cluster access token lifetime. This field defaults to **600**.
5. Under **TKGI Cluster Refresh Token Lifetime**, enter a time in seconds for the cluster refresh

token lifetime. This field defaults to [21600](#).



**Note:** VMware recommends using the default UAA token timeout values. By default, access tokens expire after ten minutes and refresh tokens expire after six hours.

6. Under **Configure created clusters to use UAA as the OIDC provider**, select **Enabled** or **Disabled**. This is a global default setting for TKGI-provisioned clusters. For more information, see [OIDC Provider for Kubernetes Clusters](#).

To configure Tanzu Kubernetes Grid Integrated Edition to use UAA as the OIDC provider:

1. Under **Configure created clusters to use UAA as the OIDC provider**, select **Enabled**.

Configure created clusters to use UAA as the OIDC provider.\*

Disabled

Enabled

UAA OIDC Groups Claim \*

roles

UAA OIDC Groups Prefix \*

oidc:

UAA OIDC Username Claim \*

user\_name

UAA OIDC Username Prefix \*

oidc:

TKGI cluster client redirect URIs

Configure your UAA user account store with either internal or external authentication mechanisms \*

Internal UAA

LDAP Server

SAML Identity Provider

**Save**

2. For **UAA OIDC Groups Claim**, enter the name of your groups claim. This is used to set a user's group in the JSON Web Token (JWT) claim. The default value is `roles`.
3. For **UAA OIDC Groups Prefix**, enter a prefix for your groups claim. This prevents conflicts with existing names. For example, if you enter the prefix `oidc:`, UAA creates a group name like `oidc:developers`. The default value is `oidc:`.
4. For **UAA OIDC Username Claim**, enter the name of your username claim. This is used to set a user's username in the JWT claim. The default value is `user_name`. Depending on your provider, you can enter claims besides `user_name`, like `email` or `name`.
5. For **UAA OIDC Username Prefix**, enter a prefix for your username claim. This prevents conflicts with existing names. For example, if you enter the prefix `oidc:`, UAA creates a username like `oidc:admin`. The default value is `oidc:`.



**Warning:** VMware recommends adding OIDC prefixes to prevent users and groups from gaining unintended cluster privileges. If you change the above values for a pre-existing Tanzu Kubernetes Grid Integrated Edition installation, you must change any existing role bindings that bind to a username or group. If you do not change your role bindings, developers cannot access Kubernetes clusters. For instructions, see [Managing Cluster Access and Permissions](#).

7. (Optional) For **TKGI cluster client redirect URIs**, enter one or more comma-delimited UAA redirect URIs. Configure **TKGI cluster client redirect URIs** to assign persistent UAA `cluster_client_redirect_uri` URIs to your clusters. UAA redirect URIs configured in the **TKGI cluster client redirect URIs** field persist through cluster updates and TKGI upgrades.
8. Select one of the following options:
  - ❖ To use an internal user account store for UAA, select **Internal UAA**. Click **Save** and continue to [\(Optional\) Host Monitoring](#).
  - ❖ To use LDAP for UAA, select **LDAP Server** and continue to [Connecting Tanzu Kubernetes Grid Integrated Edition to an LDAP Server](#).
  - ❖ To use SAML for UAA, select **SAML Identity Provider** and continue to [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#).

## (Optional) Host Monitoring

In **Host Monitoring**, you can configure monitoring of nodes and VMs using Syslog, or Telegraf.

## Configure TKGI Monitoring Features on Host

Enable Syslog for TKGI?\*

- No
- Yes

Enable Telegraf Outputs?\*

- No
- Yes

**Save**

You can configure one or more of the following:

- **Syslog:** To configure Syslog, see [Syslog](#) below. Syslog forwards log messages from all BOSH-deployed VMs to a syslog endpoint.
- **Telegraf:** To configure Telegraf, see [Configuring Telegraf in TKGI](#). The Telegraf agent sends metrics from TKGI API, control plane node, and worker node VMs to a monitoring service, such as Wavefront or Datadog.

For more information about these components, see [Monitoring TKGI and TKGI-Provisioned Clusters](#).

### Syslog

To configure Syslog for all BOSH-deployed VMs in Tanzu Kubernetes Grid Integrated Edition:

1. Click **Host Monitoring**.
2. Under **Enable Syslog for TKGI**, select **Yes**.
3. Under **Address**, enter the destination syslog endpoint.
4. Under **Port**, enter the destination syslog port.
5. Under **Transport Protocol**, select a transport protocol for log forwarding.
6. (Optional) To enable TLS encryption during log forwarding, complete the following steps:
  1. Ensure **Enable TLS** is selected.



**Note:** Logs might contain sensitive information, such as cloud provider credentials. VMware recommends that you enable TLS encryption for log forwarding.

2. Under **Permitted Peer**, provide the accepted fingerprint (SHA1) or name of remote peer. For example, \*.YOUR-LOGGING-SYSTEM.com.

3. Under **TLS Certificate**, provide a TLS certificate for the destination syslog endpoint.



**Note:** You do not need to provide a new certificate if the TLS certificate for the destination syslog endpoint is signed by a Certificate Authority (CA) in your BOSH certificate store.

7. (Optional) Under **Max Message Size**, enter a maximum message size for logs that are forwarded to a syslog endpoint. By default, the **Max Message Size** field is 10,000 characters.
8. (Optional) Under **Custom Rsyslog Configuration**, enter your RSyslog rules configuration using RainerScript syntax. For more information, see [RainerScript](#) in the RSyslog documentation. For example RSyslog rule configurations, see [Example Custom Rules](#) in the Syslog BOSH GitHub repository.
9. Click **Save**.

## (Optional) In-Cluster Monitoring

In **In-Cluster Monitoring**, you can configure one or more observability components and integrations that run in Kubernetes clusters and capture logs and metrics about your workloads. For more information, see [Monitoring Workers and Workloads](#).

## Configure PKS Monitoring Features Deployed in Cluster

Wavefront Integration\*

- No  
 Yes

Deploy cAdvisor\*

- No  
 Yes
- Enable Metric Sink Resources  
 Enable Log Sink Resources  
 Enable node exporter on workers

**Save**

To configure in-cluster monitoring:

- To configure Wavefront, see [Wavefront](#).
- To configure cAdvisor, see [cAdvisor](#).
- To configure sink resources, see:
  - [Metric Sink Resources](#)
  - [Log Sink Resources](#)

You can enable both log and metric sink resources or only one of them.

### Wavefront

You can monitor Kubernetes clusters and pods metrics externally using the integration with [Wavefront by VMware](#).



**Note:** Before you configure Wavefront integration, you must have an active Wavefront account and access to a Wavefront instance. You provide your Wavefront access token during configuration. For additional information, see the [Wavefront documentation](#).

To use Wavefront with Windows worker-based clusters, developers must install Wavefront to their clusters manually, using Helm.

To enable and configure Wavefront monitoring:

1. In the Tanzu Kubernetes Grid Integrated Edition tile, select **In-Cluster Monitoring**.
2. Under **Wavefront Integration**, select **Yes**.
3. Under **Wavefront URL**, enter the URL of your Wavefront subscription. For example:

```
https://try.wavefront.com/api
```

4. Under **Wavefront Access Token**, enter the API token for your Wavefront subscription.
5. (Optional) For installations that require a proxy server for outbound Internet access, enable access by entering values for **HTTP Proxy Host**, **HTTP Proxy Port**, **Proxy username**, and **Proxy password**.
6. Click **Save**.

The Tanzu Kubernetes Grid Integrated Edition tile does not validate your Wavefront configuration settings. To verify your setup, look for cluster and pod metrics in Wavefront.

## cAdvisor

cAdvisor is an open source tool for monitoring, analyzing, and exposing Kubernetes container resource usage and performance statistics.

To deploy a cAdvisor container:

1. Select **In-Cluster Monitoring**.
2. Under **Deploy cAdvisor**, select **Yes**.
3. Click **Save**.



**Note:** For information about configuring cAdvisor to monitor your running Kubernetes containers, see [cAdvisor](#) in the cAdvisor GitHub repository. For general information about Kubernetes cluster monitoring, see [Tools for Monitoring Resources](#) in the Kubernetes documentation.

## Metric Sink Resources

You can configure TKGI-provisioned clusters to send Kubernetes node metrics and pod metrics to metric sinks. For more information about metric sink resources and what to do after you enable them in the tile, see [Sink Resources](#) in *Monitoring Workers and Workloads*.

To enable clusters to send Kubernetes node metrics and pod metrics to metric sinks:

1. In **In-Cluster Monitoring**, select **Enable Metric Sink Resources**. If you enable this checkbox, Tanzu Kubernetes Grid Integrated Edition deploys Telegraf as a [DaemonSet](#), a pod that runs on each worker node in all your Kubernetes clusters.
2. (Optional) To enable Node Exporter to send worker node metrics to metric sinks of kind

[ClusterMetricSink](#), select **Enable node exporter on workers**. If you enable this checkbox, Tanzu Kubernetes Grid Integrated Edition deploys Node Exporter as a [DaemonSet](#), a pod that runs on each worker node in all your Kubernetes clusters.

For instructions on how to create a metric sink of kind [ClusterMetricSink](#) for Node Exporter metrics, see [Create a ClusterMetricSink Resource for Node Exporter Metrics](#) in *Creating and Managing Sink Resources*.

3. Click **Save**.

## Log Sink Resources

You can configure TKGI-provisioned clusters to send Kubernetes API events and pod logs to log sinks. For more information about log sink resources and what to do after you enable them in the tile, see [Sink Resources](#) in *Monitoring Workers and Workloads*.

To enable clusters to send Kubernetes API events and pod logs to log sinks:

1. Select **Enable Log Sink Resources**. If you enable this checkbox, Tanzu Kubernetes Grid Integrated Edition deploys Fluent Bit as a [DaemonSet](#), a pod that runs on each worker node in all your Kubernetes clusters.
2. Click **Save**.

## Tanzu Mission Control

Tanzu Kubernetes Grid Integrated Edition does not support Tanzu Mission Control integration on GCP. Skip this configuration pane.

## VMware CEIP

Tanzu Kubernetes Grid Integrated Edition-provisioned clusters send usage data to the TKGI control plane for storage. The VMware Customer Experience Improvement Program (CEIP) provides the option to also send the cluster usage data to VMware to improve customer experience.

To configure Tanzu Kubernetes Grid Integrated Edition CEIP Program settings:

1. Click **CEIP**.
2. Review the information about the CEIP.

## About the CEIP Program

VMware's Customer Experience Improvement Program ("CEIP") provides VMware with information that enables VMware to improve its products and services, to fix problems, and to advise you on how best to deploy and use our products. As part of the CEIP, VMware collects technical information about your organization's use of VMware products and services on a regular basis in association with your organization's VMware license key(s). This information does not personally identify any individual.

Customers who participate in the CEIP receive proactive support benefits that include a weekly report based on telemetry data. Contact your Customer Success Manager to subscribe to this report. You can view a sample report at <https://via.vmware.com/TKGI>.

Additional information regarding the data collected through CEIP and the purposes for which it is used by VMware is set forth in the Trust & Assurance Center at <http://www.vmware.com/trustvmware/ceip.html>. If you prefer not to participate in VMware's CEIP for this product, you should uncheck the box below. You may join or leave VMware's CEIP for this product at any time.

Join the VMware Customer Experience Improvement Program\*

- No
- Yes

Please enter your Entitlement Account Number or your Tanzu Customer Number. We need this information to generate a report for you

Please enter a label for this TKGI Installation

Assign a label to this TKGI Installation for use in your reports

Please select how you will be using this TKGI Installation \*

- Demo or Proof-of-concept
- Development or Pre-production
- Production
- I do not wish to provide this information

**Save**

[View a larger version of this image.](#)

3. If you wish to participate in CEIP, select **Yes**. Otherwise, select **No**.
4. If you selected the **Yes**, complete the following:
  - ◊ (Optional) Enter your entitlement account number or Tanzu customer number. If you are a VMware customer, you can find your entitlement account number in your **Account Summary** on [my.vmware.com](https://my.vmware.com). If you are a Pivotal customer, you can find your Pivotal Customer Number in your Pivotal Order Confirmation email.
  - ◊ (Optional) Enter a descriptive name for your TKGI installation. The label you assign to this installation will be used in CEIP reports to identify the environment.
5. To provide information about the purpose for this installation, select an option.

Please select how you will be using this TKGI Installation \*

- Demo or Proof-of-concept
- Development or Pre-production
- Production
- I do not wish to provide this information

6. Click **Save**.

## Errands

Errands are scripts that run at designated points during an installation.

To configure which post-deploy and pre-delete errands run for Tanzu Kubernetes Grid Integrated Edition:

1. Make a selection in the dropdown next to each errand.



**Note:** We recommend that you use the default settings for all errands except for the **Run smoke tests** errand.

2. (Optional) Set the **Run smoke tests** errand to **On**.

This errand uses the TKGI CLI to create a Kubernetes cluster and then delete it. If the creation or deletion fails, the errand fails and the installation of the Tanzu Kubernetes Grid Integrated Edition tile is aborted.

3. (Optional) To ensure that all of your cluster VMs are patched, configure the **Upgrade all clusters errand** errand to **On**.

Updating the Tanzu Kubernetes Grid Integrated Edition tile with a new Linux stemcell and the **Upgrade all clusters errand** enabled triggers the rolling of every Linux VM in each Kubernetes cluster. Similarly, updating the Tanzu Kubernetes Grid Integrated Edition tile with a new Windows stemcell triggers the rolling of every Windows VM in your Kubernetes clusters.



**Note:** VMware recommends that you review the Tanzu Network metadata and confirm stemcell version compatibility before using the Tanzu Network APIs to update the stemcells in your automated pipeline. For more information, see the [API reference](#).

## Resource Config

To modify the resource configuration of Tanzu Kubernetes Grid Integrated Edition and specify your TKGI API load balancer, follow the steps below:

1. Select **Resource Config**.
2. For each job, review the **Automatic** values in the following fields:
  - **INSTANCES:** Tanzu Kubernetes Grid Integrated Edition defaults to the minimum configuration. If you want a highly available configuration (beta), scale the number of VM instances as follows:
    1. To configure your Tanzu Kubernetes Grid Integrated Edition database for high availability (beta), increase the **INSTANCES** value for **TKGI Database** to **3**.
    2. To configure your Tanzu Kubernetes Grid Integrated Edition API and UAA for high availability (beta), increase the **INSTANCES** value for **TKGI API** to **2** or more.



**Warning:** High availability mode is a beta feature. Do not scale your **TKGI API** or **TKGI Database** to more than one instance in production environments.

- **VM TYPE:** By default, the **TKGI Database** and **TKGI API** jobs are set to the same **Automatic** VM type. If you want to adjust this value, we recommend that you select the same VM type for both jobs.



**Note:** The **Automatic VM TYPE** values match the recommended resource configuration for the **TKGI API** and **TKGI Database** jobs.

- ◊ **PERSISTENT DISK TYPE:** By default, the **TKGI Database** and **TKGI API** jobs are set to the same persistent disk type. If you want to adjust this value, you can change the persistent disk type for each of the jobs independently. Using the same persistent disk type for both jobs is not required.

3. For the **TKGI Database** job:

- ◊ Leave the **LOAD BALANCERS** field blank.
- ◊ (Optional) If you do not use a NAT instance, select **INTERNET CONNECTED**. This allows component instances direct access to the internet.

4. For the **TKGI API** job:

- ◊ Enter the name of your TKGI API load balancer in the **LOAD BALANCERS** field, prefixed with `tcp:..`. For example, `tcp:TKGI-API-LB`. Replace `TKGI-API-LB` with the name of your TKGI API load balancer. The name of your TKGI API load balancer is the name you configured in the [Create a Load Balancer](#) section of *Creating a GCP Load Balancer for the TKGI API*.



**Note:** After you click **Apply Changes** for the first time, BOSH assigns the TKGI API VM an IP address. BOSH uses the name you provide in the **LOAD BALANCERS** field to locate your load balancer and then connect the load balancer to the TKGI API VM using its new IP address.

\* (Optional) If you do not use a NAT instance, select **INTERNET CONNECTED**. This allows component instances direct access to the internet.



**Warning:** To avoid workload downtime, use the resource configuration recommended in [About Tanzu Kubernetes Grid Integrated Edition Upgrades](#) and [Maintaining Workload Uptime](#).

## Step 3: Apply Changes

1. Return to the Ops Manager Installation Dashboard.
2. Click **Review Pending Changes**. Select the product that you intend to deploy and review the changes. For more information, see [Reviewing Pending Product Changes](#).
3. Click **Apply Changes**.

## Step 4: Retrieve the TKGI API Endpoint

You need to retrieve the TKGI API endpoint to allow your organization to use the API to create, update, and delete Kubernetes clusters.

To retrieve the TKGI API endpoint, do the following:

1. Navigate to the Ops Manager [Installation Dashboard](#).
2. Click the **Tanzu Kubernetes Grid Integrated Edition** tile.
3. Click the **Status** tab and locate the **TKGI API** job. The IP address of the TKGI API job is the TKGI API endpoint.

## Step 5: Configure External Load Balancer

If you are installing Tanzu Kubernetes Grid Integrated Edition manually, follow the procedure in the [Create a Network Tag for the Firewall Rule](#) section of *Creating a GCP Load Balancer for the TKGI API*.

## Step 6: Install the TKGI and Kubernetes CLIs

The TKGI CLI and the Kubernetes CLI help you interact with your Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes clusters and Kubernetes workloads. To install the CLIs, follow the instructions below:

- [Installing the TKGI CLI](#)
- [Installing the Kubernetes CLI](#)

## Step 7: Configure Authentication for Tanzu Kubernetes Grid Integrated Edition

Follow the procedures in [Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on GCP](#).

## Next Steps

After installing Tanzu Kubernetes Grid Integrated Edition on GCP, you might want to do one or more of the following:

- Create a load balancer for your Tanzu Kubernetes Grid Integrated Edition clusters. For more information, see [Creating and Configuring a GCP Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters](#).
- Create your first Tanzu Kubernetes Grid Integrated Edition cluster. For more information, see [Creating Clusters](#).

## Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on GCP

This topic describes how to create admin users in VMware Tanzu Kubernetes Grid Integrated Edition with User Account and Authentication (UAA). Creating at least one admin user is a necessary step during the initial set up of Tanzu Kubernetes Grid Integrated Edition.



**Note:** Support for GCP is deprecated and will be entirely removed in a future TKGI version.

## Overview

UAA is the identity management service for Tanzu Kubernetes Grid Integrated Edition. Tanzu Kubernetes Grid Integrated Edition includes a UAA server, which is hosted on the TKGI API VM.

To interact with the UAA server, you can use the UAA Command Line Interface (UAAC). You can either run UAAC commands from the Ops Manager VM or install UAAC on your local workstation.

## Prerequisites

Before setting up admin users for Tanzu Kubernetes Grid Integrated Edition, you must have one of the following:

- SSH access to the Ops Manager VM
- A machine that can connect to your TKGI API VM

## Step 1: Connect to the TKGI API VM

You can connect to the TKGI API VM from the Ops Manager VM or from a different machine such as your local workstation.

### Option 1: Connect through the Ops Manager VM

You can connect to the TKGI API VM by logging in to the Ops Manager VM through SSH. To SSH into the Ops Manager VM on GCP, do the following:

1. Confirm that you have installed the gcloud Command Line Interface (CLI). For more information, see [Downloading gcloud](#) in the Google Cloud Platform (GCP) documentation.
2. From the GCP console, click **Compute Engine**.
3. Locate the Ops Manager VM in the **VM Instances** list.
4. Click the **SSH** menu button.
5. Copy the SSH command that appears in the pop-up window.
6. SSH into the Ops Manager VM by pasting the command into your terminal. For example:

```
$ gcloud compute ssh om-pcf-1a --zone us-central1-b
```

7. Switch to the `ubuntu` user by running the `sudo su - ubuntu` command.
8. Proceed to the [Log In as a UAA Admin](#) section to manage users with UAAC.

### Option 2: Connect through a Non-Ops Manager Machine

To connect to the TKGI API VM and run UAA commands, do the following:

1. Install UAAC on your machine. For example:

```
gem install cf-uaac
```

2. Download a copy of your Ops Manager root CA certificate to the machine. To download the certificate, do the following:

1. In a web browser, navigate to the FQDN of Ops Manager and log in.
  2. In Ops Manager, navigate to **Settings** in the drop-down menu under your username.
  3. Click **Advanced Options**.
  4. On the **Advanced Options** configuration page, click **Download Root CA Cert**.
  5. Move the certificate to a secure location on your machine and record the path.
3. Proceed to the [Log In as a UAA Admin](#) section to create admin users with UAAC.

## Step 2: Log In as a UAA Admin

Before creating TKGI users, you must log in to the UAA server as a UAA admin. To log in to the UAA server, do the following:

1. Retrieve the UAA management admin client secret:
  1. In a web browser, navigate to the Ops Manager **Installation Dashboard** and click the **Tanzu Kubernetes Grid Integrated Edition** tile.
  2. Click the **Credentials** tab.
  3. Click **Link to Credential** next to **Pks Uaa Management Admin Client** and copy the value of **secret**.
2. Target your UAA server by running the following command:

```
uaac target https://TKGI-API:8443 --ca-cert CERTIFICATE-PATH
```

Where:

- ◊ **TKGI-API** is the domain name of your TKGI API server. You entered this domain name in the **Tanzu Kubernetes Grid Integrated Edition** tile > **TKGI API** > **API Hostname (FQDN)**.
- ◊ **CERTIFICATE-PATH** is the path to your Ops Manager root CA certificate. Provide this certificate to validate the TKGI API certificate with SSL.
  - If you are logged in to the Ops Manager VM, specify `/var/tempest/workspaces/default/root_ca_certificate` as the path. This is the default location of the root certificate on the Ops Manager VM.
  - If you downloaded the Ops Manager root CA certificate to your machine, specify the path where you stored the certificate.

For example:

```
$ uaac target api.tkgi.example.com:8443 -ca-cert /var/tempest/workspaces/default/root_ca_certificate
```



**Note:** If you receive an `Unknown key: Max-Age = 86400` warning message, you can ignore it because it has no impact.

3. Authenticate with UAA by running the following command:

```
uaac token client get admin -s ADMIN-CLIENT-SECRET
```

Where `ADMIN-CLIENT-SECRET` is your UAA management admin client secret that you retrieved in a previous step. The client username is `admin`.

## Step 3: Assign Tanzu Kubernetes Grid Integrated Edition Cluster Scopes

The `pks.clusters.manage` and `pks.clusters.admin` UAA scopes grant users the ability to create and manage Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition. For information about UAA scopes in Tanzu Kubernetes Grid Integrated Edition, see [UAA Scopes for Tanzu Kubernetes Grid Integrated Edition Users](#).

To create Tanzu Kubernetes Grid Integrated Edition users with the `pks.clusters.manage` or `pks.clusters.admin` UAA scope, perform one or more of the following procedures based on the needs of your deployment:

- To assign TKGI cluster scopes to an individual user, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an Individual User](#). Follow this procedure if you selected **Internal UAA** when you configured **UAA** in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Installing Tanzu Kubernetes Grid Integrated Edition on GCP](#).
- To assign TKGI cluster scopes to an LDAP group, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an External LDAP Group](#). Follow this procedure if you selected **LDAP Server** when you configured **UAA** in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Installing Tanzu Kubernetes Grid Integrated Edition TKGI on GCP](#).
- To assign TKGI cluster scopes to a SAML group, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an External SAML Group](#). Follow this procedure if you selected **SAML Identity Provider** when you configured **UAA** in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Installing Tanzu Kubernetes Grid Integrated Edition TKGI on GCP](#).
- To assign TKGI cluster scopes to a client, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to a Client](#).

## Next Step

After you create admin users in Tanzu Kubernetes Grid Integrated Edition, the admin users can create and manage Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition. For more information, see [Managing Kubernetes Clusters and Workloads](#).

## Installing Tanzu Kubernetes Grid Integrated Edition on Amazon Web Services (AWS)

This topic lists the procedures to follow to install VMware Tanzu Kubernetes Grid Integrated Edition

on Amazon Web Services (AWS).

## Install Tanzu Kubernetes Grid Integrated Edition on AWS

To install Tanzu Kubernetes Grid Integrated Edition on AWS, follow the instructions below:

- [AWS Prerequisites and Resource Requirements](#)
- [Installing and Configuring Ops Manager on AWS](#)
- [Installing Tanzu Kubernetes Grid Integrated Edition on AWS](#)
- [Using Proxies with Tanzu Kubernetes Grid Integrated Edition on AWS](#)
- [Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on AWS](#)

## Install the TKGI and Kubernetes CLIs

The TKGI CLI and Kubernetes CLI help you interact with your Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes clusters and Kubernetes workloads.

To install the CLIs, follow the instructions below:

- [Installing the TKGI CLI](#)
- [Installing the Kubernetes CLI](#)

## AWS Prerequisites and Resource Requirements

This topic describes the prerequisites and resource requirements for installing VMware Tanzu Kubernetes Grid Integrated Edition on Amazon Web Services (AWS).

## Prerequisites

Before installing Tanzu Kubernetes Grid Integrated Edition:

1. Review the sections below.
2. Install and configure Ops Manager. To install Ops Manager, follow the instructions in [Installing and Configuring Ops Manager on AWS](#).

## Resource Requirements

Installing Ops Manager and Tanzu Kubernetes Grid Integrated Edition requires the following virtual machines (VMs):

| VM            | VM Type  | Default VM Count |
|---------------|----------|------------------|
| BOSH Director | m4.large | 1                |
| TKGI API      | m4.large | 1                |
| TKGI Database | m4.large | 1                |

 **NOTE:** VMware recommends deploying TKGI on its own dedicated Ops Manager instance, rather than on a shared Ops Manager that also hosts other runtimes such as Tanzu Application Service.

## Storage Requirements for Large Numbers of Pods

If you expect the cluster workload to run a large number of pods continuously, then increase the size of persistent disk storage allocated to the TKGI Database VM as follows:

| Number of Pods | Persistent Disk Requirements (GB) |
|----------------|-----------------------------------|
| 1,000 pods     | 20                                |
| 5,000 pods     | 100                               |
| 10,000 pods    | 200                               |
| 50,000 pods    | 1,000                             |

## Kubernetes Cluster Resources

Each Kubernetes cluster provisioned through Tanzu Kubernetes Grid Integrated Edition deploys the VMs listed below. If you deploy more than one Kubernetes cluster, you must scale your allocated resources appropriately.

| VM            | VM Count | CPU Cores | Memory (GB) | Ephemeral Disk (GB) | Persistent Disk (GB) |
|---------------|----------|-----------|-------------|---------------------|----------------------|
| Control Plane | 1        | 2         | 4           | 32                  | 5                    |
| Worker        | 1        | 2         | 4           | 32                  | 50                   |

## Installing and Configuring Ops Manager on AWS

This topic describes how to install and configure Ops Manager before deploying VMware Tanzu Kubernetes Grid Integrated Edition on Amazon Web Services (AWS).

## Prerequisites

You use Ops Manager to install and configure Tanzu Kubernetes Grid Integrated Edition. Before you install Ops Manager, review [AWS Prerequisites and Resource Requirements](#).

## Install and Configure Ops Manager

Each version of Tanzu Kubernetes Grid Integrated Edition is compatible with multiple versions of Ops Manager. To determine Ops Manager compatibility, see [VMware Tanzu Network](#).

To install and configure Ops Manager, follow the instructions in the table below:

| Version           | Instructions                                                                                                                                                            |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Ops Manager v2.10 | <ol style="list-style-type: none"> <li>1. <a href="#">Deploying Ops Manager on AWS Manually</a></li> <li>2. <a href="#">Configuring BOSH Director on AWS</a></li> </ol> |

## Next Installation Step

To install and configure Tanzu Kubernetes Grid Integrated Edition, follow the instructions in [Installing Tanzu Kubernetes Grid Integrated Edition on AWS](#).

## Installing Tanzu Kubernetes Grid Integrated Edition on AWS (Antrea and Flannel Networking)

This topic describes how to install and configure VMware Tanzu Kubernetes Grid Integrated Edition on Amazon Web Services (AWS).

## Prerequisites

Before performing the procedures in this topic, you must have deployed and configured Ops Manager. For more information, see [AWS Prerequisites and Resource Requirements](#).

This topic assumes that you have prepared the AWS environment for this VMware Tanzu Kubernetes Grid Integrated Edition deployment. For more information, see [Installing and Configuring Ops Manager on AWS](#).

If you use an instance of Ops Manager that you configured previously to install other runtimes, perform the following steps before you install Tanzu Kubernetes Grid Integrated Edition:

1. Navigate to Ops Manager.
2. Open the **Director Config** pane.
3. Select the **Enable Post Deploy Scripts** checkbox.
4. Click the **Installation Dashboard** link to return to the Installation Dashboard.
5. Click **Review Pending Changes**. Select all products you intend to deploy and review the changes. For more information, see [Reviewing Pending Product Changes](#).
6. Click **Apply Changes**.

## Step 1: Install Tanzu Kubernetes Grid Integrated Edition

To install Tanzu Kubernetes Grid Integrated Edition, do the following:

1. Download the product file from [VMware Tanzu Network](#).
2. Navigate to <https://YOUR-OPS-MANAGER-FQDN/> in a browser to log in to the Ops Manager Installation Dashboard.
3. Click **Import a Product** to upload the product file.
4. Under **Tanzu Kubernetes Grid Integrated Edition** in the left column, click the plus sign to add this product to your staging area.

## Step 2: Configure Tanzu Kubernetes Grid Integrated Edition

Click the orange **Tanzu Kubernetes Grid Integrated Edition** tile to start the configuration process.

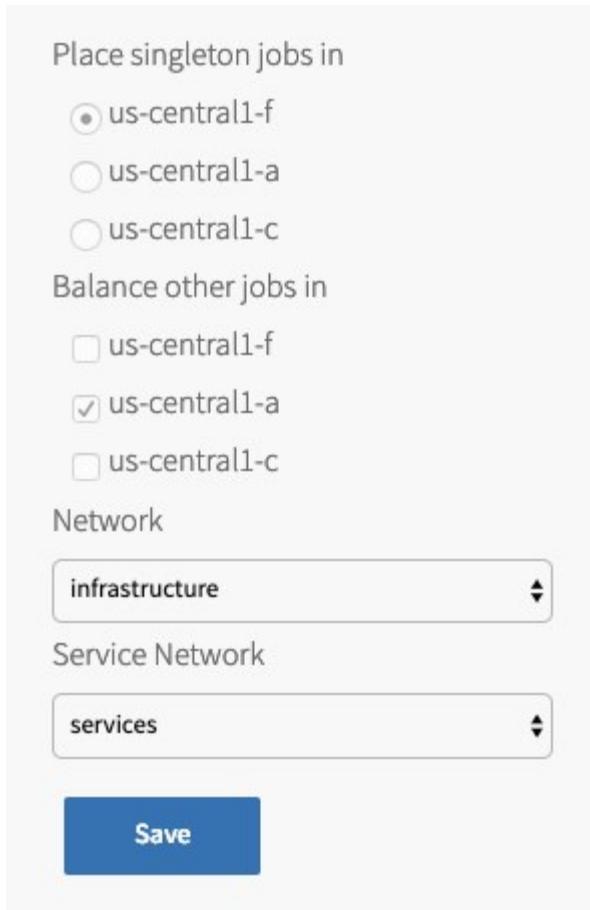


**WARNING:** When you configure the Tanzu Kubernetes Grid Integrated Edition tile, do not use spaces in any field entries. This includes spaces between characters as well as leading and trailing spaces. If you use a space in any field entry, the deployment of Tanzu Kubernetes Grid Integrated Edition fails.

## Assign AZs and Networks

To configure the availability zones (AZs) and networks used by the Tanzu Kubernetes Grid Integrated Edition control plane:

1. Click **Assign AZs and Networks**.
2. Under **Place singleton jobs in**, select the AZ where you want to deploy the TKGI API and TKGI Database.



- Under **Balance other jobs in**, select the AZ for balancing other Tanzu Kubernetes Grid Integrated Edition control plane jobs.



**Note:** You must specify the **Balance other jobs in** AZ, but the selection has no effect in the current version of Tanzu Kubernetes Grid Integrated Edition.

- Under **Network**, select the infrastructure subnet that you created for Tanzu Kubernetes Grid Integrated Edition component VMs, such as the TKGI API and TKGI Database VMs.
- Under **Service Network**, select the services subnet that you created for Kubernetes cluster VMs.
- Click **Save**.

## TKGI API

Perform the following steps:

- Click **TKGI API**.
- Under **Certificate to secure the TKGI API**, provide a certificate and private key pair.

## TKGI API Service

Certificate to secure the TKGI API \*

Certificate PEM

Private Key PEM

[Generate RSA Certificate](#)

API Hostname (FQDN) \*

tkgi.api.example.com

Worker VM Max in Flight \*

4

[Save](#)

The certificate that you supply must cover the specific subdomain that routes to the TKGI API VM with TLS termination on the ingress. If you use UAA as your OIDC provider, this certificate must be a proper certificate chain and have a SAN field.

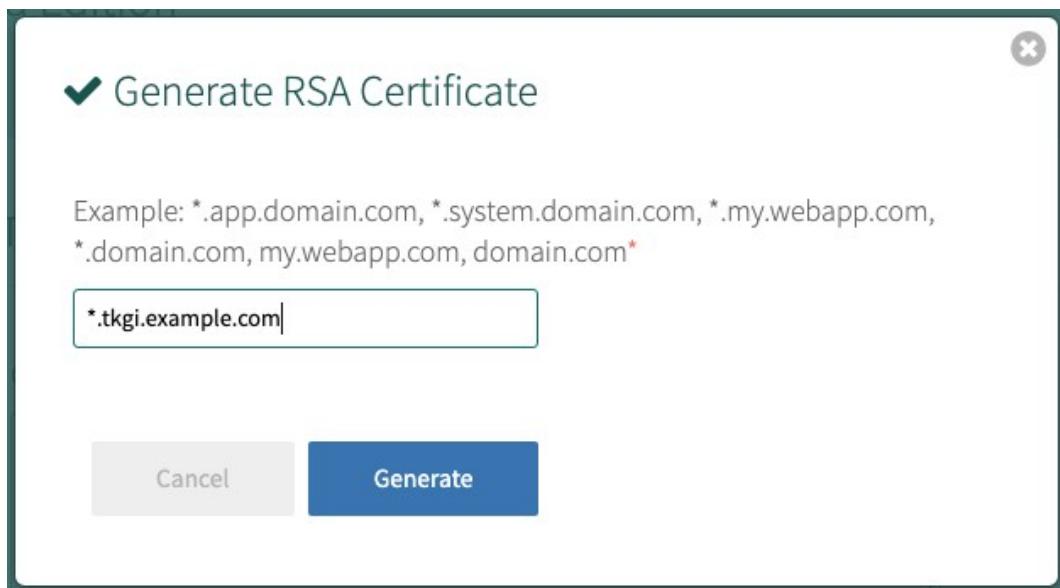


**Warning:** TLS certificates generated for wildcard DNS records only work for a single domain level. For example, a certificate generated for `*.tkgi.EXAMPLE.com` does not permit communication to `*.api.tkgi.EXAMPLE.com`. If the certificate does not contain the correct FQDN for the TKGI API, calls to the API will fail.

You can enter your own certificate and private key pair, or have Ops Manager generate one for you.

To generate a certificate using Ops Manager:

1. Click **Generate RSA Certificate** for a new install or **Change** to update a previously-generated certificate.
2. Enter the domain for your API hostname. This must match the domain you configure under **TKGI API > API Hostname (FQDN)** below, in the same pane. It can be a standard FQDN or a wildcard domain.
3. Click **Generate**.



3. Under **API Hostname (FQDN)**, enter the FQDN that you registered to point to the TKGI API load balancer, such as [api.tkgi.example.com](http://api.tkgi.example.com). To retrieve the public IP address or FQDN of the TKGI API load balancer, log in to your IaaS console.



**Note:** The FQDN for the TKGI API must not contain uppercase letters or trailing whitespace.

4. Under **Worker VM Max in Flight**, enter the maximum number of non-canary worker instances to create or update in parallel within an availability zone.

This field sets the `max_in_flight` variable value. The `max_in_flight` setting limits the number of component instances the TKGI CLI creates or starts simultaneously when running `tkgi create-cluster` or `tkgi update-cluster`. By default, `max_in_flight` is set to `4`, limiting the TKGI CLI to creating or starting a maximum of four component instances in parallel.

5. Click **Save**.

## Plans

A plan defines a set of resource types used for deploying a cluster.

### Activate a Plan

You must first activate and configure **Plan 1**, and afterwards you can optionally activate **Plan 2** through **Plan 10**.

To activate and configure a plan, perform the following steps:

1. Click the plan that you want to activate.



**Note:** Plans 11, 12, and 13 support Windows worker-based Kubernetes clusters on vSphere with NSX-T, and are a beta feature on vSphere with Flannel.

2. Select **Active** to activate the plan and make it available to developers deploying clusters.

## Configuration for Plan 1

Select 'Active' to allow users of the PKS CLI to create a cluster using this template plan.

Plan\*

Active

Name \*

small

Description \*

Example: This plan will configure a lightweight kubernetes cluster. Not recommended for production workloads.

Master/ETCD Node Instances ( min: 1, max: 5 ) \*

1

Master/ETCD VM Type\*

Automatic: medium.disk (cpu: 2, ram: 4 GB, disk: 32 GB)

Master Persistent Disk Type\*

Automatic: 10 GB

Master/ETCD Availability Zones \*

us-central1-f

us-central1-a

us-central1-c

3. Under **Name**, provide a unique name for the plan.
4. Under **Description**, edit the description as needed. The plan description appears in the Services Marketplace, which developers can access by using the TKGI CLI.
5. Under **Master/ETCD Node Instances**, select the default number of Kubernetes control

plane/etcdb nodes to provision for each cluster. You can enter 1, 3, or 5.



**Note:** If you deploy a cluster with multiple control plane/etcdb node VMs, confirm that you have sufficient hardware to handle the increased load on disk write and network traffic. For more information, see [Hardware recommendations](#) in the etcd documentation.

In addition to meeting the hardware requirements for a multi-control plane node cluster, we recommend configuring monitoring for etcd to monitor disk latency, network latency, and other indicators for the health of the cluster. For more information, see [Configuring Telegraf in TKGI](#).



**WARNING:** To change the number of control plane/etcdb nodes for a plan, you must ensure that no existing clusters use the plan. Tanzu Kubernetes Grid Integrated Edition does not support changing the number of control plane/etcdb nodes for plans with existing clusters.

6. Under **Master/ETCD VM Type**, select the type of VM to use for Kubernetes control plane/etcdb nodes. For more information, including control plane node VM customization options, see the [Control Plane Node VM Size](#) section of *VM Sizing for Tanzu Kubernetes Grid Integrated Edition Clusters*.
7. Under **Master Persistent Disk Type**, select the size of the persistent disk for the Kubernetes control plane node VM.
8. Under **Master/ETCD Availability Zones**, select one or more AZs for the Kubernetes clusters deployed by Tanzu Kubernetes Grid Integrated Edition. If you select more than one AZ, Tanzu Kubernetes Grid Integrated Edition deploys the control plane VM in the first AZ and the worker VMs across the remaining AZs. If you are using multiple control plane nodes, Tanzu Kubernetes Grid Integrated Edition deploys the control plane and worker VMs across the AZs in round-robin fashion.



**Note:** Tanzu Kubernetes Grid Integrated Edition does not support changing the AZs of existing control plane nodes.

9. Under **Maximum number of workers on a cluster**, set the maximum number of Kubernetes worker node VMs that Tanzu Kubernetes Grid Integrated Edition can deploy for each cluster. Enter any whole number in this field.

Maximum number of workers on a cluster (min: 1) \*

Worker Node Instances (min: 1) \*

Worker VM Type\*

Automatic: medium.disk (cpu: 2, ram: 4 GB, disk: 32 GB)

Worker Persistent Disk Type\*

Automatic: 50 GB

Worker Availability Zones \*

- us-central1-f
- us-central1-a
- us-central1-c

10. Under **Worker Node Instances**, specify the default number of Kubernetes worker nodes the TKGI CLI provisions for each cluster. The **Worker Node Instances** setting must be less than, or equal to, the **Maximum number of workers on a cluster** setting.

For high availability, create clusters with a minimum of three worker nodes, or two per AZ if you intend to use PersistentVolumes (PVs). For example, if you deploy across three AZs, you should have six worker nodes. For more information about PVs, see [PersistentVolumes in Maintaining Workload Uptime](#). Provisioning a minimum of three worker nodes, or two nodes per AZ is also recommended for stateless workloads.

For more information about creating clusters, see [Creating Clusters](#).



**Note:** Changing a plan's **Worker Node Instances** setting does not alter the number of worker nodes on existing clusters. For information about scaling an existing cluster, see [Scale Horizontally by Changing the Number of Worker Nodes Using the TKGI CLI in Scaling Existing Clusters](#).

11. Under **Worker VM Type**, select the type of VM to use for Kubernetes worker node VMs. For more information, including worker node VM customization options, see [Worker Node VM Number and Size in VM Sizing for Tanzu Kubernetes Grid Integrated Edition Clusters](#).



**Note:** Tanzu Kubernetes Grid Integrated Edition requires a **Worker VM Type** with an ephemeral disk size of 32 GB or more.

12. Under **Worker Persistent Disk Type**, select the size of the persistent disk for the Kubernetes worker node VMs.
13. Under **Worker Availability Zones**, select one or more AZs for the Kubernetes worker nodes. Tanzu Kubernetes Grid Integrated Edition deploys worker nodes equally across the AZs you select.
14. Under **Kubelet customization - system-reserved**, enter resource values that Kubelet can use to reserve resources for system daemons. For example, `memory=250Mi, cpu=150m`. For more information about system-reserved values, see the [Kubernetes documentation](#).

#### Kubelet customization - system-reserved

#### Kubelet customization - eviction-hard

#### Errand VM Type\*

Automatic: t3.micro (cpu: 1, ram: 1 GB, disk: 8 GB) ▾

15. Under **Kubelet customization - eviction-hard**, enter threshold limits that Kubelet can use to evict pods when they exceed the limit. Enter limits in the format `EVICTION-SIGNAL=QUANTITY`. For example, `memory.available=100Mi, nodefs.available=10%, nodefs.inodesFree=5%`. For more information about eviction thresholds, see the [Kubernetes documentation](#).



**WARNING:** Use the Kubelet customization fields with caution. If you enter values that are invalid or that exceed the limits the system supports, Kubelet might fail to start. If Kubelet fails to start, you cannot create clusters.

16. Under **Errand VM Type**, select the size of the VM that contains the errand. The smallest instance possible is sufficient, as the only errand running on this VM is the one that applies the **Default Cluster App** YAML configuration.
17. (Optional) Under **(Optional) Add-ons - Use with caution**, enter additional YAML configuration to add custom workloads to each cluster in this plan. You can specify multiple files using `---` as a separator. For more information, see [Adding Custom Linux Workloads](#).

## (Optional) Add-ons - Use with caution

 Allow Privileged

## Admission Plugins

 PodSecurityPolicy SecurityContextDeny

18. (Optional) To allow users to create pods with privileged containers, select the **Allow Privileged** option. For more information, see [Pods](#) in the Kubernetes documentation.



**Note:** Enabling the [Allow Privileged](#) option means that all containers in the cluster will run in privileged mode. [Pod Security Policy](#) provides a [privileged](#) parameter that can be used to activate or deactivate Pods running in privileged mode. As a best practice, if you activate [Allow Privileged](#), define PSP to limit which Pods run in privileged mode. If you are implementing PSP for privileged pods, you must activate [Allow Privileged](#) mode.

19. (Optional) Activate or deactivate one or more admission controller plugins: **PodSecurityPolicy** and **SecurityContextDeny**. For more information see [Using Admission Control Plugins for Tanzu Kubernetes Grid Integrated Edition Clusters](#).



**Note:** To use PodSecurityPolicy features, you must use Ops Manager v2.10.17 or later.

20. (Optional) Under **Node Drain Timeout(mins)**, enter the timeout in minutes for the node to drain pods. If you set this value to [0](#), the node drain does not terminate.

Node Drain Timeout(mins) ( min: 0, max: 1440 )  
0

Pod Shutdown Grace Period (seconds) ( min: -1, max: 86400 )  
10

Force node to drain even if it has running pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or StatefulSet.

Force node to drain even if it has running DaemonSet-managed pods.

Force node to drain even if it has running running pods using emptyDir.

Force node to drain even if pods are still running after timeout.

21. (Optional) Under **Pod Shutdown Grace Period (seconds)**, enter a timeout in seconds for the node to wait before it forces the pod to terminate. If you set this value to `-1`, the default timeout is set to the one specified by the pod.
22. (Optional) To configure when the node drains, activate the following:
  - Force node to drain even if it has running pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or StatefulSet.
  - Force node to drain even if it has running DaemonSet-managed pods.
  - Force node to drain even if it has running running pods using emptyDir.
  - Force node to drain even if pods are still running after timeout.



**Warning:** If you select **Force node to drain even if pods are still running after timeout**, the node halts all running workloads on pods. Before enabling this configuration, set **Node Drain Timeout** to a value greater than `0`.

For more information about configuring default node drain behavior, see [Worker Node Hangs Indefinitely](#) in [Troubleshooting](#).

23. Click **Save**.

## Deactivate a Plan

To deactivate a plan, perform the following steps:

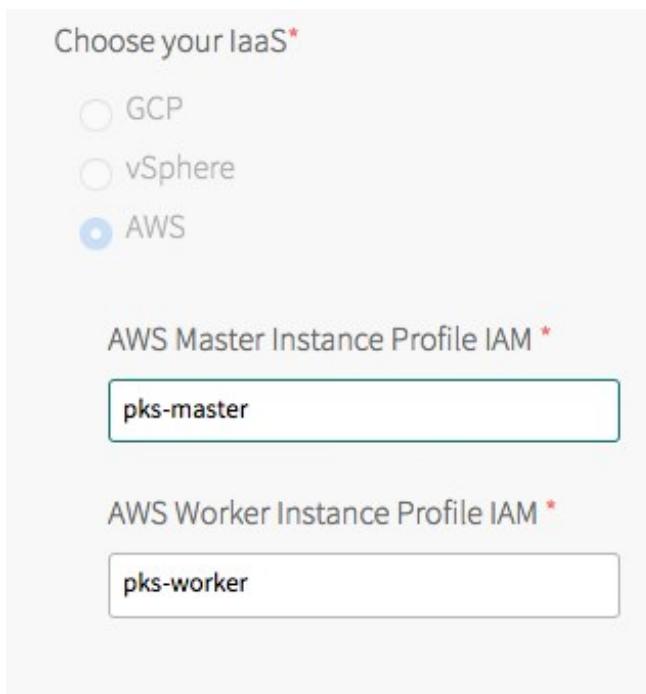
1. Click the plan that you want to deactivate.

2. Select **Inactive**.
3. Click **Save**.

## Kubernetes Cloud Provider

To configure your Kubernetes cloud provider settings, follow the procedures below:

1. Click **Kubernetes Cloud Provider**.
2. Under **Choose your IaaS**, select **AWS**.



3. Enter your **AWS Master Instance Profile IAM**. This is the instance profile name associated with the control plane node.
4. Enter your **AWS Worker Instance Profile IAM**. This is the instance profile name associated with the worker node.
5. Click **Save**.

## Networking

To configure networking, do the following:

1. Click **Networking**.
2. Under **Container Networking Interface**, select **Antrea**.

## Networking Configurations

Container Networking Interface\*

- Antrea (Switching from Flannel to Antrea is only supported on upgrade)

Kubernetes Pod Network CIDR Range \*

Kubernetes Service Network CIDR Range \*

- Flannel (Will be deprecated in TKGI 1.11)

- NSX-T

HTTP/HTTPS Proxy (for vSphere and AWS only)\*

- Disabled
- Enabled

Allow outbound internet access from Kubernetes cluster vms (IaaS-dependent)

- Enable outbound internet access

**Save**

Antrea is selected as the default Container Networking Interface (CNI). VMware recommends that you use Antrea as your CNI.



**Note:** Support for the Flannel Container Networking Interface (CNI) is deprecated. VMware recommends that you switch your Flannel CNI-configured clusters to the Antrea CNI. For more information about Flannel CNI deprecation, see [About Switching from the Flannel CNI to the Antrea CNI](#) in [About Tanzu Kubernetes Grid Integrated Edition Upgrades](#).

3. (Optional) Enter values for **Kubernetes Pod Network CIDR Range** and **Kubernetes Service Network CIDR Range**.
  - Ensure that the CIDR ranges do not overlap and have sufficient space for your deployed services.
  - Ensure that the CIDR range for the **Kubernetes Pod Network CIDR Range** is large enough to accommodate the expected maximum number of pods.
4. (Optional) Configure a global proxy for all outgoing HTTP and HTTPS traffic from your Kubernetes clusters and the TKGI API server. See [Using Proxies with Tanzu Kubernetes Grid Integrated Edition on AWS](#) for instructions to enable a proxy.
5. (Optional) If you do not use a NAT instance, select **Allow outbound internet access from**

**Kubernetes cluster vms (IaaS-dependent).** Enabling this functionality assigns external IP addresses to VMs in clusters.

6. Click **Save**.

## UAA

To configure the UAA server:

1. Click **UAA**.
2. Under **TKGI API Access Token Lifetime**, enter a time in seconds for the TKGI API access token lifetime. This field defaults to **600**.

**UAA Configuration**

---

PKS API Access Token Lifetime (in seconds) \*

600

PKS API Refresh Token Lifetime (in seconds) \*

21600

PKS Cluster Access Token Lifetime (in seconds) \*

600

PKS Cluster Refresh Token Lifetime (in seconds) \*

21600

3. Under **TKGI API Refresh Token Lifetime**, enter a time in seconds for the TKGI API refresh token lifetime. This field defaults to **21600**.
4. Under **TKGI Cluster Access Token Lifetime**, enter a time in seconds for the cluster access token lifetime. This field defaults to **600**.
5. Under **TKGI Cluster Refresh Token Lifetime**, enter a time in seconds for the cluster refresh token lifetime. This field defaults to **21600**.



**Note:** VMware recommends using the default UAA token timeout values. By default, access tokens expire after ten minutes and refresh tokens expire after six hours.

6. Under **Configure created clusters to use UAA as the OIDC provider**, select **Enabled** or **Disabled**. This is a global default setting for TKGI-provisioned clusters. For more information,

see [OIDC Provider for Kubernetes Clusters](#).

To configure Tanzu Kubernetes Grid Integrated Edition to use UAA as the OIDC provider:

- Under **Configure created clusters to use UAA as the OIDC provider**, select **Enabled**.

Configure created clusters to use UAA as the OIDC provider.\*

Disabled

Enabled

UAA OIDC Groups Claim \*

UAA OIDC Groups Prefix \*

UAA OIDC Username Claim \*

UAA OIDC Username Prefix \*

TKGI cluster client redirect URIs

Configure your UAA user account store with either internal or external authentication mechanisms \*

Internal UAA

LDAP Server

SAML Identity Provider

**Save**

- For **UAA OIDC Groups Claim**, enter the name of your groups claim. This is used to set a user's group in the JSON Web Token (JWT) claim. The default value is `roles`.
- For **UAA OIDC Groups Prefix**, enter a prefix for your groups claim. This prevents conflicts with existing names. For example, if you enter the prefix `oidc:`, UAA creates a group name like `oidc:developers`. The default value is `oidc:`.
- For **UAA OIDC Username Claim**, enter the name of your username claim. This is used to set a user's username in the JWT claim. The default value is `user_name`. Depending on your provider, you can enter claims besides `user_name`, like `email` or

name.

5. For **UAA OIDC Username Prefix**, enter a prefix for your username claim. This prevents conflicts with existing names. For example, if you enter the prefix `oidc:`, UAA creates a username like `oidc:admin`. The default value is `oidc:`.



**Warning:** VMware recommends adding OIDC prefixes to prevent users and groups from gaining unintended cluster privileges. If you change the above values for a pre-existing Tanzu Kubernetes Grid Integrated Edition installation, you must change any existing role bindings that bind to a username or group. If you do not change your role bindings, developers cannot access Kubernetes clusters. For instructions, see [Managing Cluster Access and Permissions](#).

7. (Optional) For **TKGI cluster client redirect URIs**, enter one or more comma-delimited UAA redirect URIs. Configure **TKGI cluster client redirect URIs** to assign persistent UAA `cluster_client_redirect_uri` URIs to your clusters. UAA redirect URIs configured in the **TKGI cluster client redirect URIs** field persist through cluster updates and TKGI upgrades.
8. Select one of the following options:
  - ◊ To use an internal user account store for UAA, select **Internal UAA**. Click **Save** and continue to [\(Optional\) Host Monitoring](#).
  - ◊ To use LDAP for UAA, select **LDAP Server** and continue to [Connecting Tanzu Kubernetes Grid Integrated Edition to an LDAP Server](#).
  - ◊ To use SAML for UAA, select **SAML Identity Provider** and continue to [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#).

## (Optional) Host Monitoring

In **Host Monitoring**, you can configure monitoring of nodes and VMs using Syslog, or Telegraf.

## Configure TKGI Monitoring Features on Host

Enable Syslog for TKGI?\*

- No  
 Yes

Enable Telegraf Outputs?\*

- No  
 Yes

**Save**

You can configure one or more of the following:

- **Syslog:** To configure Syslog, see [Syslog](#) below. Syslog forwards log messages from all BOSH-deployed VMs to a syslog endpoint.
- **Telegraf:** To configure Telegraf, see [Configuring Telegraf in TKGI](#). The Telegraf agent sends metrics from TKGI API, control plane node, and worker node VMs to a monitoring service, such as Wavefront or Datadog.

For more information about these components, see [Monitoring TKGI and TKGI-Provisioned Clusters](#).

### Syslog

To configure Syslog for all BOSH-deployed VMs in Tanzu Kubernetes Grid Integrated Edition:

1. Click **Host Monitoring**.
2. Under **Enable Syslog for TKGI**, select **Yes**.
3. Under **Address**, enter the destination syslog endpoint.
4. Under **Port**, enter the destination syslog port.
5. Under **Transport Protocol**, select a transport protocol for log forwarding.
6. (Optional) To enable TLS encryption during log forwarding, complete the following steps:
  1. Ensure **Enable TLS** is selected.



**Note:** Logs might contain sensitive information, such as cloud provider credentials. VMware recommends that you enable TLS encryption for log forwarding.

2. Under **Permitted Peer**, provide the accepted fingerprint (SHA1) or name of remote peer. For example, \*.YOUR-LOGGING-SYSTEM.com.

3. Under **TLS Certificate**, provide a TLS certificate for the destination syslog endpoint.



**Note:** You do not need to provide a new certificate if the TLS certificate for the destination syslog endpoint is signed by a Certificate Authority (CA) in your BOSH certificate store.

7. (Optional) Under **Max Message Size**, enter a maximum message size for logs that are forwarded to a syslog endpoint. By default, the **Max Message Size** field is 10,000 characters.
8. (Optional) Under **Custom Rsyslog Configuration**, enter your RSyslog rules configuration using RainerScript syntax. For more information, see [RainerScript](#) in the RSyslog documentation. For example RSyslog rule configurations, see [Example Custom Rules](#) in the Syslog BOSH GitHub repository.
9. Click **Save**.

## (Optional) In-Cluster Monitoring

In **In-Cluster Monitoring**, you can configure one or more observability components and integrations that run in Kubernetes clusters and capture logs and metrics about your workloads. For more information, see [Monitoring Workers and Workloads](#).

## Configure PKS Monitoring Features Deployed in Cluster

Wavefront Integration\*

- No  
 Yes

Deploy cAdvisor\*

- No  
 Yes
- Enable Metric Sink Resources  
 Enable Log Sink Resources  
 Enable node exporter on workers

**Save**

To configure in-cluster monitoring:

- To configure Wavefront, see [Wavefront](#).
- To configure cAdvisor, see [cAdvisor](#).
- To configure sink resources, see:
  - [Metric Sink Resources](#)
  - [Log Sink Resources](#)

You can enable both log and metric sink resources or only one of them.

### Wavefront

You can monitor Kubernetes clusters and pods metrics externally using the integration with [Wavefront by VMware](#).



**Note:** Before you configure Wavefront integration, you must have an active Wavefront account and access to a Wavefront instance. You provide your Wavefront access token during configuration. For additional information, see the [Wavefront documentation](#).

To use Wavefront with Windows worker-based clusters, developers must install Wavefront to their clusters manually, using Helm.

To enable and configure Wavefront monitoring:

1. In the Tanzu Kubernetes Grid Integrated Edition tile, select **In-Cluster Monitoring**.
2. Under **Wavefront Integration**, select **Yes**.
3. Under **Wavefront URL**, enter the URL of your Wavefront subscription. For example:

```
https://try.wavefront.com/api
```

4. Under **Wavefront Access Token**, enter the API token for your Wavefront subscription.
5. (Optional) For installations that require a proxy server for outbound Internet access, enable access by entering values for **HTTP Proxy Host**, **HTTP Proxy Port**, **Proxy username**, and **Proxy password**.
6. Click **Save**.

The Tanzu Kubernetes Grid Integrated Edition tile does not validate your Wavefront configuration settings. To verify your setup, look for cluster and pod metrics in Wavefront.

## cAdvisor

cAdvisor is an open source tool for monitoring, analyzing, and exposing Kubernetes container resource usage and performance statistics.

To deploy a cAdvisor container:

1. Select **In-Cluster Monitoring**.
2. Under **Deploy cAdvisor**, select **Yes**.
3. Click **Save**.



**Note:** For information about configuring cAdvisor to monitor your running Kubernetes containers, see [cAdvisor](#) in the cAdvisor GitHub repository. For general information about Kubernetes cluster monitoring, see [Tools for Monitoring Resources](#) in the Kubernetes documentation.

## Metric Sink Resources

You can configure TKGI-provisioned clusters to send Kubernetes node metrics and pod metrics to metric sinks. For more information about metric sink resources and what to do after you enable them in the tile, see [Sink Resources](#) in *Monitoring Workers and Workloads*.

To enable clusters to send Kubernetes node metrics and pod metrics to metric sinks:

1. In **In-Cluster Monitoring**, select **Enable Metric Sink Resources**. If you enable this checkbox, Tanzu Kubernetes Grid Integrated Edition deploys Telegraf as a [DaemonSet](#), a pod that runs on each worker node in all your Kubernetes clusters.
2. (Optional) To enable Node Exporter to send worker node metrics to metric sinks of kind

[ClusterMetricSink](#), select **Enable node exporter on workers**. If you enable this checkbox, Tanzu Kubernetes Grid Integrated Edition deploys Node Exporter as a [DaemonSet](#), a pod that runs on each worker node in all your Kubernetes clusters.

For instructions on how to create a metric sink of kind [ClusterMetricSink](#) for Node Exporter metrics, see [Create a ClusterMetricSink Resource for Node Exporter Metrics](#) in *Creating and Managing Sink Resources*.

3. Click **Save**.

## Log Sink Resources

You can configure TKGI-provisioned clusters to send Kubernetes API events and pod logs to log sinks. For more information about log sink resources and what to do after you enable them in the tile, see [Sink Resources](#) in *Monitoring Workers and Workloads*.

To enable clusters to send Kubernetes API events and pod logs to log sinks:

1. Select **Enable Log Sink Resources**. If you enable this checkbox, Tanzu Kubernetes Grid Integrated Edition deploys Fluent Bit as a [DaemonSet](#), a pod that runs on each worker node in all your Kubernetes clusters.
2. Click **Save**.

## Tanzu Mission Control

Tanzu Mission Control integration lets you monitor and manage Tanzu Kubernetes Grid Integrated Edition clusters from the Tanzu Mission Control console, which makes the Tanzu Mission Control console a single point of control for all Kubernetes clusters. For more information about Tanzu Mission Control, see the [VMware Tanzu Mission Control](#) home page.

To integrate Tanzu Kubernetes Grid Integrated Edition with Tanzu Mission Control:

1. Confirm that the TKGI API VM has internet access and can connect to [cna.tmc.cloud.vmware.com](https://cna.tmc.cloud.vmware.com) and the other outbound URLs listed in the [What Happens When You Attach a Cluster](#) section of the Tanzu Mission Control Product documentation.
2. Navigate to the **Tanzu Kubernetes Grid Integrated Edition** tile > the **Tanzu Mission Control** pane and select **Yes** under **Tanzu Mission Control Integration**.

Tanzu Mission Control Integration\*

No  
 Yes

Tanzu Mission Control URL \*

VMware Cloud Services API Token \*

Tanzu Mission Control Cluster Group \*

Tanzu Mission Control Cluster Name Prefix \*

**Save**

3. Configure the fields below:

- **Tanzu Mission Control URL:** Enter the Org URL of your Tanzu Mission Control subscription, without a trailing slash (/). For example, `YOUR-ORG.tmc.cloud.vmware.com`.
- **VMware Cloud Services API token:** Enter your API token to authenticate with VMware Cloud Services APIs. You can retrieve this token by logging in to [VMware Cloud Services](#) and viewing your account information.
- **Tanzu Mission Control Cluster Group:** Enter the name of a Tanzu Mission Control cluster group.

The name can be `default` or another value, depending on your role and access policy:

- `Org Member` users in VMware cloud services have a `service.admin` role in Tanzu Mission Control. These users:
  - By default, can create and attach clusters only in the `default` cluster group.
  - Can create and attach clusters to other cluster groups after an `organization.admin` user grants them the `clustergroup.admin` or

`clustergroup.edit` role for those groups.

- `Org Owner` users in VMware cloud services have `organization.admin` permissions in Tanzu Mission Control. These users:
  - Can create cluster groups.
  - Can grant `clustergroup` roles to `service.admin` users through the Tanzu Mission Control Access Policy view.

For more information about role and access policy, see [Access Control](#) in the VMware Tanzu Mission Control Product documentation.

- ◊ **Tanzu Mission Control Cluster Name Prefix:** Enter a name prefix for identifying the Tanzu Kubernetes Grid Integrated Edition clusters in Tanzu Mission Control.

4. Click **Save**.



**Warning:** After the Tanzu Kubernetes Grid Integrated Edition tile is deployed with a configured cluster group, the cluster group cannot be updated.



**Note:** When you upgrade your Kubernetes clusters and have Tanzu Mission Control integration enabled, existing clusters will be attached to Tanzu Mission Control.

## VMware CEIP

Tanzu Kubernetes Grid Integrated Edition-provisioned clusters send usage data to the TKGI control plane for storage. The VMware Customer Experience Improvement Program (CEIP) provides the option to also send the cluster usage data to VMware to improve customer experience.

To configure Tanzu Kubernetes Grid Integrated Edition CEIP Program settings:

1. Click **CEIP**.
2. Review the information about the CEIP.

## About the CEIP Program

VMware's Customer Experience Improvement Program ("CEIP") provides VMware with information that enables VMware to improve its products and services, to fix problems, and to advise you on how best to deploy and use our products. As part of the CEIP, VMware collects technical information about your organization's use of VMware products and services on a regular basis in association with your organization's VMware license key(s). This information does not personally identify any individual.

Customers who participate in the CEIP receive proactive support benefits that include a weekly report based on telemetry data. Contact your Customer Success Manager to subscribe to this report. You can view a sample report at <https://via.vmware.com/TKGI>.

Additional information regarding the data collected through CEIP and the purposes for which it is used by VMware is set forth in the Trust & Assurance Center at <http://www.vmware.com/trustvmware/ceip.html>. If you prefer not to participate in VMware's CEIP for this product, you should uncheck the box below. You may join or leave VMware's CEIP for this product at any time.

Join the VMware Customer Experience Improvement Program\*

- No
- Yes

Please enter your Entitlement Account Number or your Tanzu Customer Number. We need this information to generate a report for you

Please enter a label for this TKGI Installation

Assign a label to this TKGI Installation for use in your reports

Please select how you will be using this TKGI Installation \*

- Demo or Proof-of-concept
- Development or Pre-production
- Production
- I do not wish to provide this information

**Save**

[View a larger version of this image.](#)

3. If you wish to participate in CEIP, select **Yes**. Otherwise, select **No**.
4. If you selected the **Yes**, complete the following:
  - ◊ (Optional) Enter your entitlement account number or Tanzu customer number. If you are a VMware customer, you can find your entitlement account number in your **Account Summary** on [my.vmware.com](https://my.vmware.com). If you are a Pivotal customer, you can find your Pivotal Customer Number in your Pivotal Order Confirmation email.
  - ◊ (Optional) Enter a descriptive name for your TKGI installation. The label you assign to this installation will be used in CEIP reports to identify the environment.
5. To provide information about the purpose for this installation, select an option.

Please select how you will be using this TKGI Installation \*

- Demo or Proof-of-concept
- Development or Pre-production
- Production
- I do not wish to provide this information

6. Click **Save**.

## Errands

Errands are scripts that run at designated points during an installation.

To configure which post-deploy and pre-delete errands run for Tanzu Kubernetes Grid Integrated Edition:

1. Make a selection in the dropdown next to each errand.



**Note:** We recommend that you use the default settings for all errands except for the **Run smoke tests** errand.

2. (Optional) Set the **Run smoke tests** errand to **On**.

This errand uses the TKGI CLI to create a Kubernetes cluster and then delete it. If the creation or deletion fails, the errand fails and the installation of the Tanzu Kubernetes Grid Integrated Edition tile is aborted.

3. (Optional) To ensure that all of your cluster VMs are patched, configure the **Upgrade all clusters errand** errand to **On**.

Updating the Tanzu Kubernetes Grid Integrated Edition tile with a new Linux stemcell and the **Upgrade all clusters errand** enabled triggers the rolling of every Linux VM in each Kubernetes cluster. Similarly, updating the Tanzu Kubernetes Grid Integrated Edition tile with a new Windows stemcell triggers the rolling of every Windows VM in your Kubernetes clusters.



**Note:** VMware recommends that you review the Tanzu Network metadata and confirm stemcell version compatibility before using the Tanzu Network APIs to update the stemcells in your automated pipeline. For more information, see the [API reference](#).

## Resource Config

To modify the resource configuration of Tanzu Kubernetes Grid Integrated Edition and specify your TKGI API load balancer, follow the steps below:

1. Select **Resource Config**.
2. For each job, review the **Automatic** values in the following fields:
  - **INSTANCES:** Tanzu Kubernetes Grid Integrated Edition defaults to the minimum configuration. If you want a highly available configuration (beta), scale the number of VM instances as follows:
    1. To configure your Tanzu Kubernetes Grid Integrated Edition database for high availability (beta), increase the **INSTANCES** value for **TKGI Database** to **3**.
    2. To configure your Tanzu Kubernetes Grid Integrated Edition API and UAA for high availability (beta), increase the **INSTANCES** value for **TKGI API** to **2** or more.



**Warning:** High availability mode is a beta feature. Do not scale your **TKGI API** or **TKGI Database** to more than one instance in production environments.

- **VM TYPE:** By default, the **TKGI Database** and **TKGI API** jobs are set to the same **Automatic** VM type. If you want to adjust this value, we recommend that you select the same VM type for both jobs.



**Note:** The **Automatic VM TYPE** values match the recommended resource configuration for the **TKGI API** and **TKGI Database** jobs.

- ◊ **PERSISTENT DISK TYPE:** By default, the **TKGI Database** and **TKGI API** jobs are set to the same persistent disk type. If you want to adjust this value, you can change the persistent disk type for each of the jobs independently. Using the same persistent disk type for both jobs is not required.

3. For the **TKGI Database** job:

- ◊ Leave the **LOAD BALANCERS** field blank.
- ◊ (Optional) If you do not use a NAT instance, select **INTERNET CONNECTED**. This allows component instances direct access to the internet.

4. For the **TKGI API** job:

- ◊ In the **LOAD BALANCERS** field, enter all values of `pks_api_target_groups` from the Terraform output, prefixed with `alb:`. For example, `alb:ENV-pks-tg-9021,alb:ENV-pks-tg-8443`. Replace `ENV` with the `env_name` that you defined when you set up Terraform. For example, `alb:pcf-pks-tg-9021,alb:pcf-pks-tg-8443`.



**Note:** After you click **Apply Changes** for the first time, BOSH assigns the TKGI API VM an IP address. BOSH uses the name you provide in the **LOAD BALANCERS** field to locate your load balancer and then connect the load balancer to the TKGI API VM using its new IP address.

\* (Optional) If you do not use a NAT instance, select **INTERNET CONNECTED**. This allows component instances direct access to the internet.



**Warning:** To avoid workload downtime, use the resource configuration recommended in [About Tanzu Kubernetes Grid Integrated Edition Upgrades](#) and [Maintaining Workload Uptime](#).

## Step 3: Apply Changes

1. Return to the Ops Manager Installation Dashboard.
2. Click **Review Pending Changes**. Select the product that you intend to deploy and review the changes. For more information, see [Reviewing Pending Product Changes](#).
3. Click **Apply Changes**.

## Step 4: Retrieve the TKGI API Endpoint

You need to retrieve the TKGI API endpoint to allow your organization to use the API to create, update, and delete Kubernetes clusters.

To retrieve the TKGI API endpoint, do the following:

1. Navigate to the Ops Manager [Installation Dashboard](#).
2. Click the **Tanzu Kubernetes Grid Integrated Edition** tile.
3. Click the **Status** tab and locate the **TKGI API** job. The IP address of the TKGI API job is the TKGI API endpoint.

## Step 5: Install the TKGI and Kubernetes CLIs

The TKGI CLI and the Kubernetes CLI help you interact with your Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes clusters and Kubernetes workloads. To install the CLIs, follow the instructions below:

- [Installing the TKGI CLI](#)
- [Installing the Kubernetes CLI](#)

## Step 6: Configure Authentication for Tanzu Kubernetes Grid Integrated Edition

Follow the procedures in [Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on AWS](#).

## Next Steps

After installing Tanzu Kubernetes Grid Integrated Edition on AWS, you might want to do one or more of the following:

- Create a load balancer for your Tanzu Kubernetes Grid Integrated Edition clusters. For more information, see [Creating and Configuring an AWS Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters](#).
- Create your first Tanzu Kubernetes Grid Integrated Edition cluster. For more information, see [Creating Clusters](#).

## Using Proxies with Tanzu Kubernetes Grid Integrated Edition on AWS

This topic describes how HTTP/HTTPS proxies work in Tanzu Kubernetes Grid Integrated Edition (TKGI) on AWS, and how to set proxies globally.

To configure proxy settings specifically for individual TKGI clusters, see [Configure Cluster Proxies](#).

## Overview

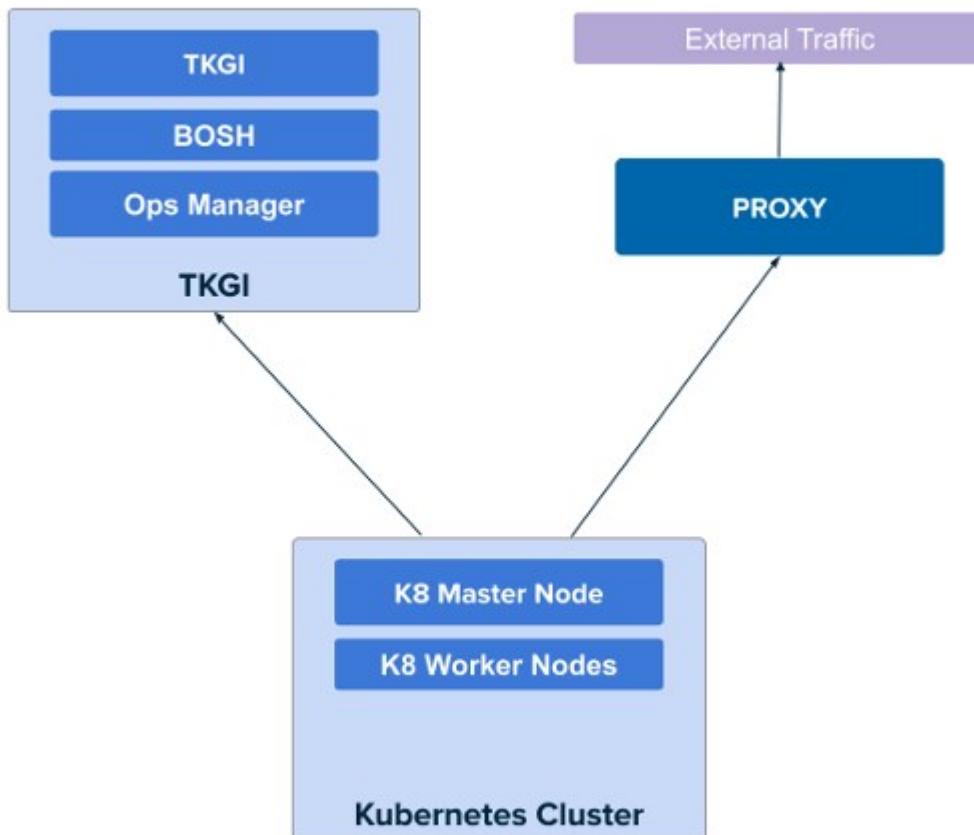
If your environment includes HTTP proxies, you can configure Tanzu Kubernetes Grid Integrated Edition on AWS to use these proxies so that Tanzu Kubernetes Grid Integrated Edition-deployed Kubernetes control plane and worker nodes access public Internet services and other internal services through a proxy.

In addition, Tanzu Kubernetes Grid Integrated Edition proxy settings apply to the TKGI API instance. When an Tanzu Kubernetes Grid Integrated Edition operator creates a Kubernetes cluster, the TKGI

API VM behind a proxy is able to manage AWS components on the standard network.

You can also proxy outgoing HTTP/HTTPS traffic from Ops Manager and the BOSH Director so that all Tanzu Kubernetes Grid Integrated Edition components use the same proxy service.

The following diagram illustrates the network architecture:



## Enable TKGI API and Kubernetes Proxy

To configure a global HTTP proxy for all outgoing HTTP/HTTPS traffic from the Kubernetes cluster nodes and the TKGI API server, perform the following steps:

1. Navigate to Ops Manager and log in.
2. Click the **Tanzu Kubernetes Grid Integrated Edition** tile.
3. Click **Networking**.
4. Under **HTTP/HTTPS Proxy**, select **Enabled** to configure an Tanzu Kubernetes Grid Integrated Edition global proxy for all outgoing HTTP and HTTPS traffic from your Kubernetes clusters.

**HTTP/HTTPS Proxy (for vSphere and AWS only)\***

Disabled  
 Enabled

**HTTP Proxy URL**

**HTTP Proxy Credentials**

Username  
 Password

**HTTPS Proxy URL**

**HTTPS Proxy Credentials**

Username  
 Password

**No Proxy**

5. (Optional) Configure Tanzu Kubernetes Grid Integrated Edition to use a proxy.

Production environments can deny direct access to public Internet services and between internal services by placing an HTTP or HTTPS proxy in the network path between Kubernetes nodes and those services.

Configure Tanzu Kubernetes Grid Integrated Edition to use your proxy and activate the following:

- TKGI API access to public Internet services and other internal services.
- Tanzu Kubernetes Grid Integrated Edition-deployed Kubernetes nodes access to public Internet services and other internal services.
- Tanzu Kubernetes Grid Integrated Edition Telemetry ability to forward Telemetry data to the CEIP and Telemetry program.



**Note:** This setting does not set the proxy for running Kubernetes workloads or pods.

6. To complete your global proxy configuration for all outgoing HTTP/HTTPS traffic from your

Kubernetes clusters, perform the following steps:

1. To proxy outgoing HTTP traffic, enter the URL of your HTTP proxy endpoint under **HTTP Proxy URL**. For example, `http://myproxy.com:1234`.
2. (Optional) If your outgoing HTTP proxy uses basic authentication, enter the username and password in the **HTTP Proxy Credentials** fields.
3. To proxy outgoing HTTPS traffic, enter the URL of your HTTP proxy endpoint under **HTTPS Proxy URL**. For example, `http://myproxy.com:1234`.



**Note:** Using an HTTPS connection to the proxy server is not supported. HTTP and HTTPS proxy options can only be configured with an HTTP connection to the proxy server. You cannot populate either of the proxy URL fields with an HTTPS URL. The proxy host and port can be different for HTTP and HTTPS traffic, but the proxy protocol must be HTTP.

4. (Optional) If your HTTPS proxy uses basic authentication, enter the username and password in the **HTTPS Proxy Credentials** fields.
5. Under **No Proxy**, enter the comma-separated list of IP addresses that must bypass the proxy to allow for internal Tanzu Kubernetes Grid Integrated Edition communication.

The **No Proxy** list should include `127.0.0.1` and `localhost`.

Also include the following in the **No Proxy** list:

- Your Tanzu Kubernetes Grid Integrated Edition environment's CIDRs, such as the service network CIDR where your Tanzu Kubernetes Grid Integrated Edition cluster is deployed, the deployment network CIDR, the node network IP block CIDR, and the pod network IP block CIDR.
- The FQDN of any registry, such as the Harbor API FQDN, or component communicating with Tanzu Kubernetes Grid Integrated Edition, using a hostname instead of an IP address.
- Any additional IP addresses or domain names that should bypass the proxy.

The **No Proxy** property for AWS accepts wildcard domains denoted by a prefixed `\*`. or ..

For example:

```
127.0.0.1,localhost, *.example1.com, .example2.com, example3.com,
198.51.100.0/24, 203.0.113.0/24, 192.0.2.0/24
```



**Note:** By default the `169.254.169.254, 10.100.0.0/8` and

10.200.0.0/8 IP address ranges, `.internal`, `.svc`, `.svc.cluster.local`, `.svc.cluster`, and your Tanzu Kubernetes Grid Integrated Edition FQDN are not proxied. This allows internal Tanzu Kubernetes Grid Integrated Edition communication.

Do not use the `_` character in the **No Proxy** field. Entering an underscore character in this field can cause upgrades to fail.

Because some jobs in the VMs accept `\*` as a wildcard, while others only accept `.`, we recommend that you define a wildcard domain using both of them. For example, to denote `example.com` as a wildcard domain, add both `\*.example.com` and `example.com` to the **No Proxy** property.

7. To save your changes to the TKGI tile, click **Save**.
8. Proceed with any remaining Tanzu Kubernetes Grid Integrated Edition tile configurations and deploy Tanzu Kubernetes Grid Integrated Edition. See [Installing Tanzu Kubernetes Grid Integrated Edition on AWS](#).

## Enable Ops Manager and BOSH Proxy

To enable an HTTP proxy for outgoing HTTP/HTTPS traffic from Ops Manager and the BOSH Director, perform the following steps:

1. Log in to Ops Manager.
2. Select **User Name > Settings** in the upper right.
3. Click **Proxy Settings**.
4. Under **HTTP Proxy**, enter the FQDN or IP address of the HTTP proxy endpoint. For example, `http://myproxy.com:80`.
5. Under **HTTPS Proxy**, enter the FQDN or IP address of the HTTPS proxy endpoint. For example, `http://myproxy.com:80`.



**Note:** Using an HTTPS connection to the proxy server is not supported. Ops Manager and BOSH HTTP and HTTPS proxy options can be only configured with an HTTP connection to the proxy.

6. Under **No Proxy**, include the hosts that must bypass the proxy. This is required.

In addition to `127.0.0.1` and `localhost`, include the BOSH Director IP, Ops Manager IP, TKGI API VM IP, and the TKGI Database VM IP. If the TKGI Database is in HA mode (beta), enter all your database IPs in the **No Proxy** field.

|                                                                                      |
|--------------------------------------------------------------------------------------|
| 127.0.0.1,localhost,BOSH-DIRECTOR-IP,TKGI-API-IP,OPS-MANAGER-IP,TKGI-DATABASE-I<br>P |
|--------------------------------------------------------------------------------------|



**Note:** Ops Manager does not allow the use of a CIDR range in the **No Proxy** field. You must specify each individual IP address to bypass the proxy.

The **No Proxy** field does not accept wildcard domain notation, such as `.docker.io` and `.docker.com`. You must specify the exact IP or FQDN to bypass the proxy, such as `registry-1.docker.io`.

7. Click **Save**.
8. Return to the Ops Manager Installation Dashboard and click **Review Pending Changes**.
9. Click **Apply Changes** to deploy Ops Manager and the BOSH Director with the updated proxy settings.

## Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on AWS

This topic describes how to create admin users in VMware Tanzu Kubernetes Grid Integrated Edition with User Account and Authentication (UAA). Creating at least one admin user is a necessary step during the initial set up of Tanzu Kubernetes Grid Integrated Edition.

### Overview

UAA is the identity management service for Tanzu Kubernetes Grid Integrated Edition. Tanzu Kubernetes Grid Integrated Edition includes a UAA server, which is hosted on the TKGI API VM.

To interact with the UAA server, you can use the UAA Command Line Interface (UAAC). You can either run UAAC commands from the Ops Manager VM or install UAAC on your local workstation.

### Prerequisites

Before setting up admin users for Tanzu Kubernetes Grid Integrated Edition, you must have one of the following:

- SSH access to the Ops Manager VM
- A machine that can connect to your TKGI API VM

### Step 1: Connect to the TKGI API VM

You can connect to the TKGI API VM from the Ops Manager VM or from a different machine such as your local workstation.

#### Option 1: Connect through the Ops Manager VM

You can connect to the TKGI API VM by logging in to the Ops Manager VM through SSH. To SSH into the Ops Manager VM on AWS, do the following:

1. Retrieve the key pair you used when you created the Ops Manager VM. To see the name of the key pair:

1. In the AWS console, click the Ops Manager VM
2. Locate the **key pair name** in the properties.
2. On the AWS **EC2 instances** page, locate the Ops Manager FQDN.
3. Change the permissions on the **.pem** file to be more restrictive by running the **chmod 600** command. For example:

```
$ chmod 600 ops_mgr.pem
```

4. SSH into the Ops Manager VM by running the following command:

```
ssh -i ops_mgr.pem ubuntu@OPS-MANAGER-FQDN
```

Where **OPS-MANAGER-FQDN** is the FQDN of Ops Manager. For example:

```
$ ssh -i ops_mgr.pem ubuntu@my-opsmanager-fqdn.example.com
```

5. Proceed to the **Log In as a UAA Admin** section to manage users with UAAC.

## Option 2: Connect through a Non-Ops Manager Machine

To connect to the TKGI API VM and run UAA commands, do the following:

1. Install UAAC on your machine. For example:

```
gem install cf-uaac
```

2. Download a copy of your Ops Manager root CA certificate to the machine. To download the certificate, do the following:
  1. In a web browser, navigate to the FQDN of Ops Manager and log in.
  2. In Ops Manager, navigate to **Settings** in the drop-down menu under your username.
  3. Click **Advanced Options**.
  4. On the **Advanced Options** configuration page, click **Download Root CA Cert**.
  5. Move the certificate to a secure location on your machine and record the path.
3. Proceed to the **Log In as a UAA Admin** section to create admin users with UAAC.

## Step 2: Log In as a UAA Admin

Before creating TKGI users, you must log in to the UAA server as a UAA admin. To log in to the UAA server, do the following:

1. Retrieve the UAA management admin client secret:
  1. In a web browser, navigate to the Ops Manager **Installation Dashboard** and click the **Tanzu Kubernetes Grid Integrated Edition** tile.
  2. Click the **Credentials** tab.
  3. Click **Link to Credential** next to **Pks Uaa Management Admin Client** and copy the value of

`secret.`

- Target your UAA server by running the following command:

```
uaac target https://TKGI-API:8443 --ca-cert CERTIFICATE-PATH
```

Where:

- TKGI-API is the domain name of your TKGI API server. You entered this domain name in the **Tanzu Kubernetes Grid Integrated Edition** tile > **TKGI API** > **API Hostname (FQDN)**.
- CERTIFICATE-PATH is the path to your Ops Manager root CA certificate. Provide this certificate to validate the TKGI API certificate with SSL.
  - If you are logged in to the Ops Manager VM, specify `/var/tempest/workspaces/default/root_ca_certificate` as the path. This is the default location of the root certificate on the Ops Manager VM.
  - If you downloaded the Ops Manager root CA certificate to your machine, specify the path where you stored the certificate.

For example:

```
$ uaac target api.tkgi.example.com:8443 -ca-cert /var/tempest/workspaces/default/root_ca_certificate
```



**Note:** If you receive an `Unknown key: Max-Age = 86400` warning message, you can ignore it because it has no impact.

- Authenticate with UAA by running the following command:

```
uaac token client get admin -s ADMIN-CLIENT-SECRET
```

Where `ADMIN-CLIENT-SECRET` is your UAA management admin client secret that you retrieved in a previous step. The client username is `admin`.

## Step 3: Assign Tanzu Kubernetes Grid Integrated Edition Cluster Scopes

The `pks.clusters.manage` and `pks.clusters.admin` UAA scopes grant users the ability to create and manage Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition. For information about UAA scopes in Tanzu Kubernetes Grid Integrated Edition, see [UAA Scopes for Tanzu Kubernetes Grid Integrated Edition Users](#).

To create Tanzu Kubernetes Grid Integrated Edition users with the `pks.clusters.manage` or `pks.clusters.admin` UAA scope, perform one or more of the following procedures based on the needs of your deployment:

- To assign TKGI cluster scopes to an individual user, see [Grant Tanzu Kubernetes Grid](#)

[Integrated Edition Access to an Individual User](#). Follow this procedure if you selected **Internal UAA** when you configured **UAA** in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Installing Tanzu Kubernetes Grid Integrated Edition on AWS](#).

- To assign TKGI cluster scopes to an LDAP group, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an External LDAP Group](#). Follow this procedure if you selected **LDAP Server** when you configured **UAA** in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Installing Tanzu Kubernetes Grid Integrated Edition TKGI on AWS](#).
- To assign TKGI cluster scopes to a SAML group, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an External SAML Group](#). Follow this procedure if you selected **SAML Identity Provider** when you configured **UAA** in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Installing Tanzu Kubernetes Grid Integrated Edition TKGI on AWS](#).
- To assign TKGI cluster scopes to a client, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to a Client](#).

## Next Step

After you create admin users in Tanzu Kubernetes Grid Integrated Edition, the admin users can create and manage Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition. For more information, see [Managing Kubernetes Clusters and Workloads](#).

## Installing Tanzu Kubernetes Grid Integrated Edition on Microsoft Azure

This topic lists the procedures to follow to install VMware Tanzu Kubernetes Grid Integrated Edition on Microsoft Azure.

## Install Tanzu Kubernetes Grid Integrated Edition on Azure

To install Tanzu Kubernetes Grid Integrated Edition on Azure, follow the instructions below:

- [Azure Prerequisites and Resource Requirements](#)
- [Installing and Configuring Ops Manager on Azure](#)
- [Creating Managed Identities in Azure for Tanzu Kubernetes Grid Integrated Edition](#)
- [Installing Tanzu Kubernetes Grid Integrated Edition on Azure](#)
- [Configuring an Azure Load Balancer for the TKGI API](#)
- [Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on Azure](#)

## Install the TKGI and Kubernetes CLIs

The TKGI CLI and Kubernetes CLI help you interact with your Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes clusters and Kubernetes workloads.

To install the CLIs, follow the instructions below:

- [Installing the TKGI CLI](#)
- [Installing the Kubernetes CLI](#)

## Azure Prerequisites and Resource Requirements

This topic describes the prerequisites and resource requirements for installing VMware Tanzu Kubernetes Grid Integrated Edition on Microsoft Azure.

### Prerequisites

Before installing Tanzu Kubernetes Grid Integrated Edition:

1. Review the sections below.
2. Install and configure Ops Manager. To install Ops Manager, follow the instructions in [Installing and Configuring Ops Manager on Azure](#).

### Subscription Requirements

For Tanzu Kubernetes Grid Integrated Edition and Kubernetes services to run correctly, you must have at least a [standard](#) subscription tier.

### Resource Requirements

Installing Ops Manager and Tanzu Kubernetes Grid Integrated Edition requires the following virtual machines (VMs):

| VM            | CPU | Memory (GB) | Ephemeral Disk (GB) |
|---------------|-----|-------------|---------------------|
| BOSH Director | 2   | 8           | 16                  |
| Ops Manager   | 1   | 8           | 120                 |
| TKGI API      | 2   | 8           | 64                  |
| TKGI Database | 2   | 8           | 64                  |



**NOTE:** VMware recommends deploying TKGI on its own dedicated Ops Manager instance, rather than on a shared Ops Manager that also hosts other runtimes such as Tanzu Application Service.

### Storage Requirements for Large Numbers of Pods

If you expect the cluster workload to run a large number of pods continuously, then increase the size of persistent disk storage allocated to the TKGI Database VM as follows:

| Number of Pods | Persistent Disk Requirements (GB) |
|----------------|-----------------------------------|
| 1,000 pods     | 20                                |
| 5,000 pods     | 100                               |

|             |       |
|-------------|-------|
| 10,000 pods | 200   |
| 50,000 pods | 1,000 |

## Kubernetes Cluster Resources

Each Kubernetes cluster provisioned through Tanzu Kubernetes Grid Integrated Edition deploys the VMs listed below. If you deploy more than one Kubernetes cluster, you must scale your allocated resources appropriately.

| VM            | VM Count | CPU Cores | Memory (GB) | Ephemeral Disk (GB) | Persistent Disk (GB) |
|---------------|----------|-----------|-------------|---------------------|----------------------|
| Control Plane | 1        | 2         | 4           | 32                  | 5                    |
| Worker        | 1        | 2         | 4           | 32                  | 50                   |

## Installing and Configuring Ops Manager on Azure

This topic describes how to install and configure Ops Manager before deploying VMware Tanzu Kubernetes Grid Integrated Edition on Azure.

## Prerequisites

You use Ops Manager to install and configure Tanzu Kubernetes Grid Integrated Edition. Before you install Ops Manager, review [Azure Prerequisites and Resource Requirements](#).

## Install and Configure Ops Manager

Each version of Tanzu Kubernetes Grid Integrated Edition is compatible with multiple versions of Ops Manager. To determine Ops Manager compatibility, see [VMware Tanzu Network](#).

To install and configure Ops Manager, follow the instructions in the table below:

| Version           | Instructions                                                                                                                                                                                                                                                       |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Ops Manager v2.10 | <ol style="list-style-type: none"> <li>1. <a href="#">Preparing to Deploy Ops Manager on Azure Manually</a></li> <li>2. <a href="#">Deploying Ops Manager on Azure Manually</a></li> <li>3. <a href="#">Configuring BOSH Director on Azure Manually</a></li> </ol> |

## Next Installation Step

To create managed identities for Tanzu Kubernetes Grid Integrated Edition, follow the instructions in [Creating Managed Identities in Azure for Tanzu Kubernetes Grid Integrated Edition](#).

## Creating Managed Identities in Azure for Tanzu Kubernetes Grid Integrated Edition

This topic describes how to create managed identities for VMware Tanzu Kubernetes Grid Integrated Edition on Azure.

In order for Kubernetes to create load balancers and attach persistent disks to pods, you must create managed identities with sufficient permissions.

You need separate managed identities for the Kubernetes cluster control plane and worker node VMs. VMware recommends configuring each service account with the least permissive privileges and unique credentials.

## Retrieve Your Subscription ID and Resource Group

To perform the procedures in this topic, you must have your Azure Subscription ID and the name of your Tanzu Kubernetes Grid Integrated Edition Resource Group.

If you do not know your Subscription ID or Resource Group:

1. Navigate to the Azure portal.
2. Click **Resource groups**.
3. Determine the name of your Tanzu Kubernetes Grid Integrated Edition Resource Group.
4. Determine the Subscription ID for your Tanzu Kubernetes Grid Integrated Edition Resource Group.



**Note:** You specified the Subscription ID to use and your Tanzu Kubernetes Grid Integrated Edition Resource Group name when completing the steps in [Step 1: Create Network Resources in Deploying Ops Manager on Azure Manually](#).

## Create the Control Plane Node Managed Identity

Perform the following steps to create the managed identity for the control plane nodes:

1. Create a role definition using the following template:

```
{
 "Name": "TKGI control plane",
 "IsCustom": true,
 "Description": "Permissions for TKGI control plane",
 "Actions": [
 "Microsoft.Network/*",
 "Microsoft.Compute/disks/*",
 "Microsoft.Compute/virtualMachines/write",
 "Microsoft.Compute/virtualMachines/read",
 "Microsoft.Storage/storageAccounts/*"
],
 "NotActions": [
],
 "DataActions": [
],
 "NotDataActions": [
],
 "AssignableScopes": [
 "/subscriptions/SUBSCRIPTION-ID/resourceGroups/RESOURCE-GROUP"
]
}
```

```
}
```

Where:

- ◊ `SUBSCRIPTION-ID` is your Subscription ID.
- ◊ `RESOURCE-GROUP` is the name of your Tanzu Kubernetes Grid Integrated Edition Resource Group.

For more information about custom roles in Azure, see [Custom Roles in Azure](#) in the Azure documentation.

2. Save your template as `tkgi_master_role.json`.
3. To log in, run the following command with the Azure CLI:

```
az login
```

To authenticate, navigate to the URL in the output, enter the provided code, and click your account.

4. Create the role in Azure by running the following command from the directory with `tkgi_master_role.json`:

```
az role definition create --role-definition tkgi_master_role.json
```

5. Create a managed identity by running the following command:

```
az identity create -g RESOURCE-GROUP -n tkgi-master
```

Where `RESOURCE-GROUP` is the name of your Tanzu Kubernetes Grid Integrated Edition resource group.

For more information about managed identities, see [Create a user-assigned managed identity](#) in the Azure documentation.

6. Assign managed identity access to the Tanzu Kubernetes Grid Integrated Edition resource group by performing the following steps:
  1. Navigate to the Azure Portal and log in.
  2. Open the Tanzu Kubernetes Grid Integrated Edition resource group.
  3. Click **Access control (IAM)** on the left panel.
  4. Click **Add role assignment**.
  5. On the **Add role assignment** page, enter the following configurations:
    1. For **Assign access to**, select **User Assigned Managed Identity**.
    2. For **Role**, select **TKGI master**.
    3. For **Select**, select the `tkgi-master` identity created above.



**Note:** The **TKGI control plane** custom role created above is less permissive than the built-in roles provided by Azure. However, if you want to use the built-in roles instead of the recommended custom role, you can select the following three built-in

roles in Azure: **Storage Account Contributor**, **Network Contributor**, and **Virtual Machine Contributor**.

## Create the Worker Node Managed Identity

Perform the following steps to create the managed identity for the worker nodes:

1. Create a role definition using the following template:

```
{
 "Name": "TKGI worker",
 "IsCustom": true,
 "Description": "Permissions for TKGI worker",
 "Actions": [
 "Microsoft.Compute/virtualMachines/read",
 "Microsoft.Storage/storageAccounts/*"
],
 "NotActions": [
],
 "DataActions": [
],
 "NotDataActions": [
],
 "AssignableScopes": [
 "/subscriptions/SUBSCRIPTION-ID/resourceGroups/RESOURCE-GROUP"
]
}
```

Where:

- ◊ **SUBSCRIPTION-ID** is your Subscription ID.
- ◊ **RESOURCE-GROUP** is the name of your Tanzu Kubernetes Grid Integrated Edition Resource Group.

2. Save your template as `tkgi_worker_role.json`.
3. Create the role in Azure by running the following command from the directory with `tkgi_worker_role.json`:

```
az role definition create --role-definition tkgi_worker_role.json
```

4. Create a managed identity by running the following command:

```
az identity create -g RESOURCE-GROUP -n tkgi-worker
```

Where **RESOURCE-GROUP** is the name of your Tanzu Kubernetes Grid Integrated Edition resource group.

5. Assign managed identity access to the Tanzu Kubernetes Grid Integrated Edition resource group by performing the following steps:
  1. Navigate to the Azure Portal and log in.

2. Open the Tanzu Kubernetes Grid Integrated Edition resource group.
3. Click **Access control (IAM)** on the left panel.
4. Click **Add role assignment**.
5. On the **Add role assignment** page, enter the following configurations:
  1. For **Assign access to**, select **User Assigned Managed Identity**.
  2. For **Role**, select **TKGI worker**.
  3. For **Select**, select the **tkgi-worker** identity created above.



**Note:** The **TKGI worker** custom role created above is less permissive than the built-in roles provided by Azure. However, if you want to use the built-in roles instead of the recommended custom role, you can select the **Storage Account Contributor** built-in role in Azure.

## Next Installation Step

To install and configure Tanzu Kubernetes Grid Integrated Edition, follow the instructions in [Installing Tanzu Kubernetes Grid Integrated Edition on Azure](#).

## Installing Tanzu Kubernetes Grid Integrated Edition on Azure (Antrea and Flannel Networking)

This topic describes how to install and configure VMware Tanzu Kubernetes Grid Integrated Edition on Azure.

## Prerequisites

Before performing the procedures in this topic, you must have deployed and configured Ops Manager. For more information, see [Azure Prerequisites and Resource Requirements](#).

If you use an instance of Ops Manager that you configured previously to install other runtimes, perform the following steps before you install Tanzu Kubernetes Grid Integrated Edition:

1. Navigate to Ops Manager.
2. Open the **Director Config** pane.
3. Select the **Enable Post Deploy Scripts** checkbox.
4. Click the **Installation Dashboard** link to return to the Installation Dashboard.
5. Click **Review Pending Changes**. Select all products you intend to deploy and review the changes. For more information, see [Reviewing Pending Product Changes](#).
6. Click **Apply Changes**.

## Step 1: Install Tanzu Kubernetes Grid Integrated Edition

To install Tanzu Kubernetes Grid Integrated Edition, do the following:

1. Download the product file from [VMware Tanzu Network](#).
2. Navigate to <https://YOUR-OPS-MANAGER-FQDN/> in a browser to log in to the Ops Manager Installation Dashboard.
3. Click **Import a Product** to upload the product file.
4. Under **Tanzu Kubernetes Grid Integrated Edition** in the left column, click the plus sign to add this product to your staging area.

## Step 2: Configure Tanzu Kubernetes Grid Integrated Edition

Click the orange **Tanzu Kubernetes Grid Integrated Edition** tile to start the configuration process.

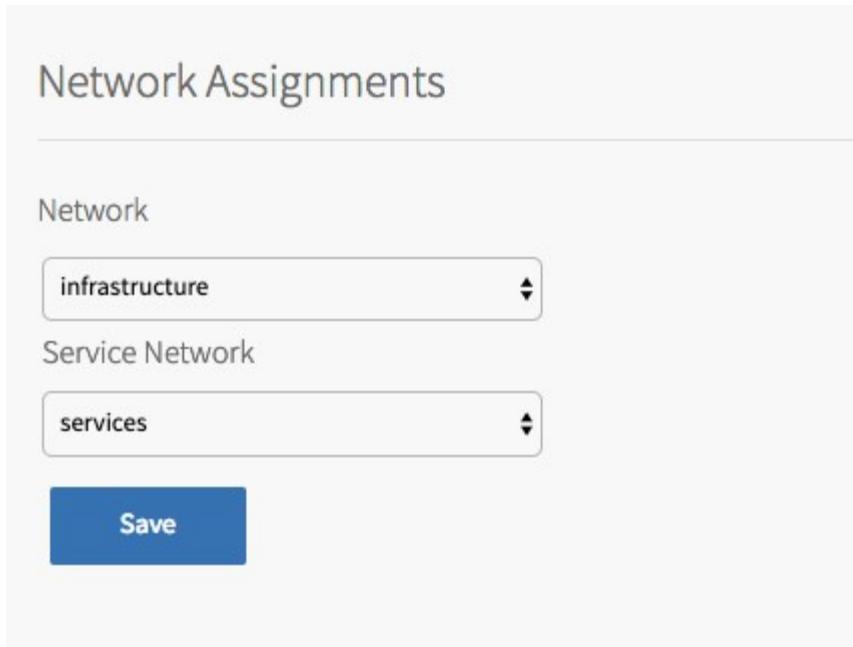


**WARNING:** When you configure the Tanzu Kubernetes Grid Integrated Edition tile, do not use spaces in any field entries. This includes spaces between characters as well as leading and trailing spaces. If you use a space in any field entry, the deployment of Tanzu Kubernetes Grid Integrated Edition fails.

## Assign Networks

To configure the networks used by the Tanzu Kubernetes Grid Integrated Edition control plane:

1. Click **Assign Networks**.



2. Under **Network**, select the infrastructure subnet that you created for Tanzu Kubernetes Grid Integrated Edition component VMs, such as the TKGI API and TKGI Database VMs. For example, `infrastructure`.
3. Under **Service Network**, select the services subnet that you created for Kubernetes cluster VMs. For example, `services`.
4. Click **Save**.

## TKGI API

Perform the following steps:

1. Click **TKGI API**.
2. Under **Certificate to secure the TKGI API**, provide a certificate and private key pair.

## TKGI API Service

Certificate to secure the TKGI API \*

Certificate PEM

Private Key PEM

[Generate RSA Certificate](#)

API Hostname (FQDN) \*

tkgi.api.example.com

Worker VM Max in Flight \*

4

[Save](#)

The certificate that you supply must cover the specific subdomain that routes to the TKGI API VM with TLS termination on the ingress. If you use UAA as your OIDC provider, this certificate must be a proper certificate chain and have a SAN field.

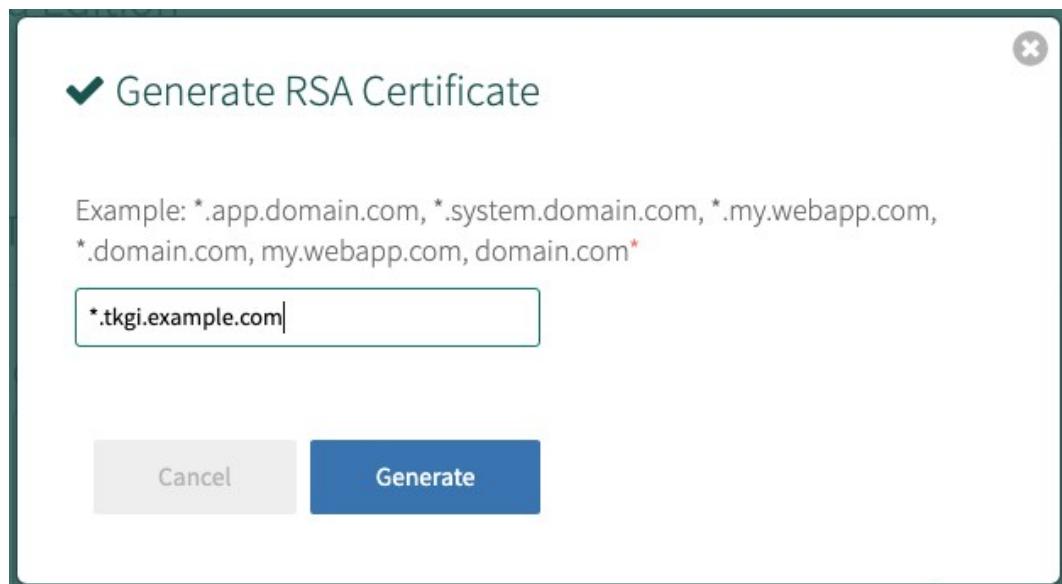


**Warning:** TLS certificates generated for wildcard DNS records only work for a single domain level. For example, a certificate generated for `*.tkgi.EXAMPLE.com` does not permit communication to `*.api.tkgi.EXAMPLE.com`. If the certificate does not contain the correct FQDN for the TKGI API, calls to the API will fail.

You can enter your own certificate and private key pair, or have Ops Manager generate one for you.

To generate a certificate using Ops Manager:

1. Click **Generate RSA Certificate** for a new install or **Change** to update a previously-generated certificate.
2. Enter the domain for your API hostname. This must match the domain you configure under **TKGI API > API Hostname (FQDN)** below, in the same pane. It can be a standard FQDN or a wildcard domain.
3. Click **Generate**.



3. Under **API Hostname (FQDN)**, enter the FQDN that you registered to point to the TKGI API load balancer, such as [api.tkgi.example.com](http://api.tkgi.example.com). To retrieve the public IP address or FQDN of the TKGI API load balancer, log in to your IaaS console.



**Note:** The FQDN for the TKGI API must not contain uppercase letters or trailing whitespace.

4. Under **Worker VM Max in Flight**, enter the maximum number of non-canary worker instances to create or update in parallel within an availability zone.

This field sets the `max_in_flight` variable value. The `max_in_flight` setting limits the number of component instances the TKGI CLI creates or starts simultaneously when running `tkgi create-cluster` or `tkgi update-cluster`. By default, `max_in_flight` is set to `4`, limiting the TKGI CLI to creating or starting a maximum of four component instances in parallel.

5. Click **Save**.

## Plans

A plan defines a set of resource types used for deploying a cluster.

### Activate a Plan

You must first activate and configure **Plan 1**, and afterwards you can optionally activate **Plan 2** through **Plan 10**.

To activate and configure a plan, perform the following steps:

1. Click the plan that you want to activate.



**Note:** Plans 11, 12, and 13 support Windows worker-based Kubernetes clusters on vSphere with NSX-T, and are a beta feature on vSphere with Flannel.

2. Select **Active** to activate the plan and make it available to developers deploying clusters.

## Configuration for Plan 1

Select 'Active' to allow users of the PKS CLI to create a cluster using this template plan.

### Plan\*

Active

### Name \*

small

### Description \*

Example: This plan will configure a lightweight kubernetes cluster. Not recommended for production workloads.

### Master/ETCD Node Instances ( min: 1, max: 5 ) \*

1

### Master/ETCD VM Type\*

Automatic: medium.disk (cpu: 2, ram: 4 GB, disk: 32 GB)



### Master Persistent Disk Type\*

Automatic: 10 GB



### Master/ETCD Availability Zones \*

us-central1-f

us-central1-a

us-central1-c

3. Under **Name**, provide a unique name for the plan.
4. Under **Description**, edit the description as needed. The plan description appears in the Services Marketplace, which developers can access by using the TKGI CLI.
5. Under **Master/ETCD Node Instances**, select the default number of Kubernetes control

plane/etcdb nodes to provision for each cluster. You can enter 1, 3, or 5.



**Note:** If you deploy a cluster with multiple control plane/etcdb node VMs, confirm that you have sufficient hardware to handle the increased load on disk write and network traffic. For more information, see [Hardware recommendations](#) in the etcd documentation.

In addition to meeting the hardware requirements for a multi-control plane node cluster, we recommend configuring monitoring for etcd to monitor disk latency, network latency, and other indicators for the health of the cluster. For more information, see [Configuring Telegraf in TKGI](#).



**WARNING:** To change the number of control plane/etcdb nodes for a plan, you must ensure that no existing clusters use the plan. Tanzu Kubernetes Grid Integrated Edition does not support changing the number of control plane/etcdb nodes for plans with existing clusters.

6. Under **Master/ETCD VM Type**, select the type of VM to use for Kubernetes control plane/etcdb nodes. For more information, including control plane node VM customization options, see the [Control Plane Node VM Size](#) section of *VM Sizing for Tanzu Kubernetes Grid Integrated Edition Clusters*.
7. Under **Master Persistent Disk Type**, select the size of the persistent disk for the Kubernetes control plane node VM.
8. Under **Master/ETCD Availability Zones**, select one or more AZs for the Kubernetes clusters deployed by Tanzu Kubernetes Grid Integrated Edition. If you select more than one AZ, Tanzu Kubernetes Grid Integrated Edition deploys the control plane VM in the first AZ and the worker VMs across the remaining AZs. If you are using multiple control plane nodes, Tanzu Kubernetes Grid Integrated Edition deploys the control plane and worker VMs across the AZs in round-robin fashion.



**Note:** Tanzu Kubernetes Grid Integrated Edition does not support changing the AZs of existing control plane nodes.

9. Under **Maximum number of workers on a cluster**, set the maximum number of Kubernetes worker node VMs that Tanzu Kubernetes Grid Integrated Edition can deploy for each cluster. Enter any whole number in this field.

Maximum number of workers on a cluster (min: 1) \*

Worker Node Instances (min: 1) \*

Worker VM Type\*

Automatic: medium.disk (cpu: 2, ram: 4 GB, disk: 32 GB)

Worker Persistent Disk Type\*

Automatic: 50 GB

Worker Availability Zones \*

- us-central1-f
- us-central1-a
- us-central1-c

10. Under **Worker Node Instances**, specify the default number of Kubernetes worker nodes the TKGI CLI provisions for each cluster. The **Worker Node Instances** setting must be less than, or equal to, the **Maximum number of workers on a cluster** setting.

For high availability, create clusters with a minimum of three worker nodes, or two per AZ if you intend to use PersistentVolumes (PVs). For example, if you deploy across three AZs, you should have six worker nodes. For more information about PVs, see [PersistentVolumes in Maintaining Workload Uptime](#). Provisioning a minimum of three worker nodes, or two nodes per AZ is also recommended for stateless workloads.

For more information about creating clusters, see [Creating Clusters](#).



**Note:** Changing a plan's **Worker Node Instances** setting does not alter the number of worker nodes on existing clusters. For information about scaling an existing cluster, see [Scale Horizontally by Changing the Number of Worker Nodes Using the TKGI CLI in Scaling Existing Clusters](#).

11. Under **Worker VM Type**, select the type of VM to use for Kubernetes worker node VMs. For more information, including worker node VM customization options, see [Worker Node VM Number and Size in VM Sizing for Tanzu Kubernetes Grid Integrated Edition Clusters](#).



**Note:** Tanzu Kubernetes Grid Integrated Edition requires a **Worker VM Type** with an ephemeral disk size of 32 GB or more.

12. Under **Worker Persistent Disk Type**, select the size of the persistent disk for the Kubernetes worker node VMs.
13. Under **Worker Availability Zones**, select one or more AZs for the Kubernetes worker nodes. Tanzu Kubernetes Grid Integrated Edition deploys worker nodes equally across the AZs you select.
14. Under **Kubelet customization - system-reserved**, enter resource values that Kubelet can use to reserve resources for system daemons. For example, `memory=250Mi`, `cpu=150m`. For more information about system-reserved values, see the [Kubernetes documentation](#).

#### Kubelet customization - system-reserved

#### Kubelet customization - eviction-hard

#### Errand VM Type\*

Automatic: t3.micro (cpu: 1, ram: 1 GB, disk: 8 GB) ▾

15. Under **Kubelet customization - eviction-hard**, enter threshold limits that Kubelet can use to evict pods when they exceed the limit. Enter limits in the format `EVICTION-SIGNAL=QUANTITY`. For example, `memory.available=100Mi`, `nodefs.available=10%`, `nodefs.inodesFree=5%`. For more information about eviction thresholds, see the [Kubernetes documentation](#).



**WARNING:** Use the Kubelet customization fields with caution. If you enter values that are invalid or that exceed the limits the system supports, Kubelet might fail to start. If Kubelet fails to start, you cannot create clusters.

16. Under **Errand VM Type**, select the size of the VM that contains the errand. The smallest instance possible is sufficient, as the only errand running on this VM is the one that applies the **Default Cluster App** YAML configuration.
17. (Optional) Under **(Optional) Add-ons - Use with caution**, enter additional YAML configuration to add custom workloads to each cluster in this plan. You can specify multiple files using `---` as a separator. For more information, see [Adding Custom Linux Workloads](#).

## (Optional) Add-ons - Use with caution

 Allow Privileged

## Admission Plugins

 PodSecurityPolicy SecurityContextDeny

18. (Optional) To allow users to create pods with privileged containers, select the **Allow Privileged** option. For more information, see [Pods](#) in the Kubernetes documentation.



**Note:** Enabling the `Allow Privileged` option means that all containers in the cluster will run in privileged mode. [Pod Security Policy](#) provides a `privileged` parameter that can be used to activate or deactivate Pods running in privileged mode. As a best practice, if you activate `Allow Privileged`, define PSP to limit which Pods run in privileged mode. If you are implementing PSP for privileged pods, you must activate `Allow Privileged` mode.

19. (Optional) Activate or deactivate one or more admission controller plugins: [PodSecurityPolicy](#) and [SecurityContextDeny](#). For more information see [Using Admission Control Plugins for Tanzu Kubernetes Grid Integrated Edition Clusters](#).



**Note:** To use PodSecurityPolicy features, you must use Ops Manager v2.10.17 or later.

20. (Optional) Under **Node Drain Timeout(mins)**, enter the timeout in minutes for the node to drain pods. If you set this value to `0`, the node drain does not terminate.

Node Drain Timeout(mins) ( min: 0, max: 1440 )  
0

Pod Shutdown Grace Period (seconds) ( min: -1, max: 86400 )  
10

Force node to drain even if it has running pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or StatefulSet.

Force node to drain even if it has running DaemonSet-managed pods.

Force node to drain even if it has running running pods using emptyDir.

Force node to drain even if pods are still running after timeout.

21. (Optional) Under **Pod Shutdown Grace Period (seconds)**, enter a timeout in seconds for the node to wait before it forces the pod to terminate. If you set this value to `-1`, the default timeout is set to the one specified by the pod.
22. (Optional) To configure when the node drains, activate the following:
  - Force node to drain even if it has running pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or StatefulSet.
  - Force node to drain even if it has running DaemonSet-managed pods.
  - Force node to drain even if it has running running pods using emptyDir.
  - Force node to drain even if pods are still running after timeout.



**Warning:** If you select **Force node to drain even if pods are still running after timeout**, the node halts all running workloads on pods. Before enabling this configuration, set **Node Drain Timeout** to a value greater than `0`.

For more information about configuring default node drain behavior, see [Worker Node Hangs Indefinitely](#) in [Troubleshooting](#).

23. Click **Save**.

## Deactivate a Plan

To deactivate a plan, perform the following steps:

1. Click the plan that you want to deactivate.

2. Select **Inactive**.
3. Click **Save**.

## Kubernetes Cloud Provider

To configure your Kubernetes cloud provider settings, follow the procedures below:

1. Click **Kubernetes Cloud Provider**.
2. Under **Choose your IaaS**, select **Azure**.

**Kubernetes Cloud Provider Configuration**

Kubernetes has the concept of a Cloud Provider, which is a module which provides an interface for managing TCP Load Balancers, Worker Nodes (Instances) and Networking Routes. Provide Kubernetes information about your cloud here.

Choose your IaaS\*

GCP  
 vSphere  
 AWS  
 Azure

Azure Cloud Name\*

Azure Public Cloud

Subscription ID \*

Tenant ID \*

Location \*

Resource Group \*

Virtual Network \*

Virtual Network Resource Group \*

Default Security Group \*

Primary Availability Set \*

The screenshot shows a configuration form for a Tanzu Kubernetes Grid. It includes fields for 'Master Managed Identity' and 'Worker Managed Identity', each with a placeholder box. Below these is a checkbox for 'Disable Outbound SNAT'. At the bottom is a large blue 'Save' button.

3. Under **Azure Cloud Name**, select the identifier of your Azure environment.
4. Enter **Subscription ID**. This is the ID of the Azure subscription that the cluster is deployed in.
5. Enter **Tenant ID**. This is the Azure Active Directory (AAD) tenant ID for the subscription that the cluster is deployed in.
6. Enter **Location**. This is the location of the resource group that the cluster is deployed in.
  1. If you do not already know the valid location value for your resource group, determine it:
    1. You set the location name in [Step 1: Create Network Resources in Deploying Ops Manager on Azure Manually](#).
    2. The location name property is a lower-case string without spaces. For example, if your resource group location is `Central US`, the location name property value is `centralus`.
    3. To determine the valid location value for your resource group location, list the valid locations:

```
az account list-locations
```

2. Enter **Location**. Enter the valid location value for your resource group location into the **Location** field.
7. Enter **Resource Group**. This is the name of the resource group that the cluster is deployed in.
8. Enter **Virtual Network**. This is the name of the virtual network that the cluster is deployed in.
9. Enter **Virtual Network Resource Group**. This is the name of the resource group that the virtual network is deployed in.
10. Enter **Default Security Group**. This is the name of the security group attached to the cluster's subnet.



**Note:** Tanzu Kubernetes Grid Integrated Edition automatically assigns the

default security group to each VM when you create a Kubernetes cluster.

However, on Azure this automatic assignment might not occur. For more information, see [Azure Default Security Group Is Not Automatically Assigned to Cluster VMs in Tanzu Kubernetes Grid Integrated Edition Release Notes](#).

1. Enter **Primary Availability Set**. This is the name of the availability set that will be used as the load balancer back end. Locate the name of the availability set within the Azure console.
2. For **Master Managed Identity**, enter `tkgi-master`. You created the managed identity for the control plane nodes in [Create the Control Plane Nodes Managed Identity](#) in *Creating Managed Identities in Azure for Tanzu Kubernetes Grid Integrated Edition*.
3. For **Worker Managed Identity**, enter `tkgi-worker`. You created the managed identity for the worker nodes in [Create the Worker Nodes Managed Identity](#) in *Creating Managed Identities in Azure for Tanzu Kubernetes Grid Integrated Edition*.
4. Select **Disable Outbound SNAT** to deactivate the default outbound SNAT rule for Azure.
5. Click **Save**.

## Networking

To configure networking, do the following:

1. Click **Networking**.
2. Under **Container Networking Interface**, select **Antrea**.

## Networking Configurations

Container Networking Interface\*

- Antrea (Switching from Flannel to Antrea is only supported on upgrade)

Kubernetes Pod Network CIDR Range \*

10.200.0.0/16

Kubernetes Service Network CIDR Range \*

10.10.0.200.0/24

- Flannel (Will be deprecated in TKGI 1.11)

- NSX-T

HTTP/HTTPS Proxy (for vSphere and AWS only)\*

- Disabled

- Enabled

Allow outbound internet access from Kubernetes cluster vms (IaaS-dependent)

Enable outbound internet access

Save

Antrea is selected as the default Container Networking Interface (CNI). VMware recommends that you use Antrea as your CNI.



**Note:** Support for the Flannel Container Networking Interface (CNI) is deprecated. VMware recommends that you switch your Flannel CNI-configured clusters to the Antrea CNI. For more information about Flannel CNI deprecation, see [About Switching from the Flannel CNI to the Antrea CNI](#) in [About Tanzu Kubernetes Grid Integrated Edition Upgrades](#).

3. (Optional) Enter values for **Kubernetes Pod Network CIDR Range** and **Kubernetes Service Network CIDR Range**.
  - Ensure that the CIDR ranges do not overlap and have sufficient space for your deployed services.
  - Ensure that the CIDR range for the **Kubernetes Pod Network CIDR Range** is large enough to accommodate the expected maximum number of pods.
4. Under **Allow outbound internet access from Kubernetes cluster vms (IaaS-dependent)**, leave the **Enable outbound internet access** checkbox unselected. You must leave this checkbox unselected due to an incompatibility between the public dynamic IPs provided by BOSH and load balancers on Azure.

5. Click **Save**.

## UAA

To configure the UAA server:

1. Click **UAA**.
2. Under **TKGI API Access Token Lifetime**, enter a time in seconds for the TKGI API access token lifetime. This field defaults to **600**.

**UAA Configuration**

---

PKS API Access Token Lifetime (in seconds) \*

600

PKS API Refresh Token Lifetime (in seconds) \*

21600

PKS Cluster Access Token Lifetime (in seconds) \*

600

PKS Cluster Refresh Token Lifetime (in seconds) \*

21600

3. Under **TKGI API Refresh Token Lifetime**, enter a time in seconds for the TKGI API refresh token lifetime. This field defaults to **21600**.
4. Under **TKGI Cluster Access Token Lifetime**, enter a time in seconds for the cluster access token lifetime. This field defaults to **600**.
5. Under **TKGI Cluster Refresh Token Lifetime**, enter a time in seconds for the cluster refresh token lifetime. This field defaults to **21600**.



**Note:** VMware recommends using the default UAA token timeout values. By default, access tokens expire after ten minutes and refresh tokens expire after six hours.

6. Under **Configure created clusters to use UAA as the OIDC provider**, select **Enabled** or **Disabled**. This is a global default setting for TKGI-provisioned clusters. For more information, see [OIDC Provider for Kubernetes Clusters](#).

To configure Tanzu Kubernetes Grid Integrated Edition to use UAA as the OIDC provider:

- Under **Configure created clusters to use UAA as the OIDC provider**, select **Enabled**.

Configure created clusters to use UAA as the OIDC provider.\*

Disabled  
 Enabled

UAA OIDC Groups Claim \*

roles

UAA OIDC Groups Prefix \*

oidc:

UAA OIDC Username Claim \*

user\_name

UAA OIDC Username Prefix \*

oidc:

TKGI cluster client redirect URIs

Configure your UAA user account store with either internal or external authentication mechanisms \*

Internal UAA  
 LDAP Server  
 SAML Identity Provider

**Save**

- For **UAA OIDC Groups Claim**, enter the name of your groups claim. This is used to set a user's group in the JSON Web Token (JWT) claim. The default value is `roles`.
- For **UAA OIDC Groups Prefix**, enter a prefix for your groups claim. This prevents conflicts with existing names. For example, if you enter the prefix `oidc:`, UAA creates a group name like `oidc:developers`. The default value is `oidc:`.
- For **UAA OIDC Username Claim**, enter the name of your username claim. This is used to set a user's username in the JWT claim. The default value is `user_name`. Depending on your provider, you can enter claims besides `user_name`, like `email` or `name`.
- For **UAA OIDC Username Prefix**, enter a prefix for your username claim. This

prevents conflicts with existing names. For example, if you enter the prefix `oidc:`, UAA creates a username like `oidc:admin`. The default value is `oidc:`.



**Warning:** VMware recommends adding OIDC prefixes to prevent users and groups from gaining unintended cluster privileges. If you change the above values for a pre-existing Tanzu Kubernetes Grid Integrated Edition installation, you must change any existing role bindings that bind to a username or group. If you do not change your role bindings, developers cannot access Kubernetes clusters. For instructions, see [Managing Cluster Access and Permissions](#).

7. (Optional) For **TKGI cluster client redirect URIs**, enter one or more comma-delimited UAA redirect URIs. Configure **TKGI cluster client redirect URIs** to assign persistent UAA `cluster_client_redirect_uri` URIs to your clusters. UAA redirect URIs configured in the **TKGI cluster client redirect URIs** field persist through cluster updates and TKGI upgrades.
8. Select one of the following options:
  - ◊ To use an internal user account store for UAA, select **Internal UAA**. Click **Save** and continue to [\(Optional\) Host Monitoring](#).
  - ◊ To use LDAP for UAA, select **LDAP Server** and continue to [Connecting Tanzu Kubernetes Grid Integrated Edition to an LDAP Server](#).
  - ◊ To use SAML for UAA, select **SAML Identity Provider** and continue to [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#).

## (Optional) Host Monitoring

In **Host Monitoring**, you can configure monitoring of nodes and VMs using Syslog, or Telegraf.

### Configure TKGI Monitoring Features on Host

---

Enable Syslog for TKGI?\*

No

Yes

Enable Telegraf Outputs?\*

No

Yes

[Save](#)

You can configure one or more of the following:

- **Syslog:** To configure Syslog, see [Syslog](#) below. Syslog forwards log messages from all BOSH-deployed VMs to a syslog endpoint.
- **Telegraf:** To configure Telegraf, see [Configuring Telegraf in TKGI](#). The Telegraf agent sends metrics from TKGI API, control plane node, and worker node VMs to a monitoring service, such as Wavefront or Datadog.

For more information about these components, see [Monitoring TKGI and TKGI-Provisioned Clusters](#).

## Syslog

To configure Syslog for all BOSH-deployed VMs in Tanzu Kubernetes Grid Integrated Edition:

1. Click **Host Monitoring**.
2. Under **Enable Syslog for TKGI**, select **Yes**.
3. Under **Address**, enter the destination syslog endpoint.
4. Under **Port**, enter the destination syslog port.
5. Under **Transport Protocol**, select a transport protocol for log forwarding.
6. (Optional) To enable TLS encryption during log forwarding, complete the following steps:
  1. Ensure **Enable TLS** is selected.



**Note:** Logs might contain sensitive information, such as cloud provider credentials. VMware recommends that you enable TLS encryption for log forwarding.

2. Under **Permitted Peer**, provide the accepted fingerprint (SHA1) or name of remote peer. For example, `*.YOUR-LOGGING-SYSTEM.com`.
3. Under **TLS Certificate**, provide a TLS certificate for the destination syslog endpoint.
4. Under **Max Message Size**, enter a maximum message size for logs that are forwarded to a syslog endpoint. By default, the **Max Message Size** field is 10,000 characters.
5. Under **Custom RSyslog Configuration**, enter your RSyslog rules configuration using RainerScript syntax. For more information, see [RainerScript](#) in the RSyslog documentation. For example RSyslog rule configurations, see [Example Custom Rules](#) in the Syslog BOSH GitHub repository.
6. Click **Save**.



**Note:** You do not need to provide a new certificate if the TLS certificate for the destination syslog endpoint is signed by a Certificate Authority (CA) in your BOSH certificate store.

## (Optional) In-Cluster Monitoring

In **In-Cluster Monitoring**, you can configure one or more observability components and integrations that run in Kubernetes clusters and capture logs and metrics about your workloads. For more information, see [Monitoring Workers and Workloads](#).

## Configure PKS Monitoring Features Deployed in Cluster

### Wavefront Integration\*

- No  
 Yes

### Deploy cAdvisor\*

- No  
 Yes
- Enable Metric Sink Resources  
 Enable Log Sink Resources  
 Enable node exporter on workers

**Save**

To configure in-cluster monitoring:

- To configure Wavefront, see [Wavefront](#).
- To configure cAdvisor, see [cAdvisor](#).
- To configure sink resources, see:
  - [Metric Sink Resources](#)
  - [Log Sink Resources](#)

You can enable both log and metric sink resources or only one of them.

### Wavefront

You can monitor Kubernetes clusters and pods metrics externally using the integration with [Wavefront by VMware](#).



**Note:** Before you configure Wavefront integration, you must have an active Wavefront account and access to a Wavefront instance. You provide your Wavefront access token during configuration. For additional information, see the [Wavefront documentation](#).

To use Wavefront with Windows worker-based clusters, developers must install Wavefront to their clusters manually, using Helm.

To enable and configure Wavefront monitoring:

1. In the Tanzu Kubernetes Grid Integrated Edition tile, select **In-Cluster Monitoring**.
2. Under **Wavefront Integration**, select **Yes**.
3. Under **Wavefront URL**, enter the URL of your Wavefront subscription. For example:

```
https://try.wavefront.com/api
```

4. Under **Wavefront Access Token**, enter the API token for your Wavefront subscription.
5. (Optional) For installations that require a proxy server for outbound Internet access, enable access by entering values for **HTTP Proxy Host**, **HTTP Proxy Port**, **Proxy username**, and **Proxy password**.
6. Click **Save**.

The Tanzu Kubernetes Grid Integrated Edition tile does not validate your Wavefront configuration settings. To verify your setup, look for cluster and pod metrics in Wavefront.

## cAdvisor

cAdvisor is an open source tool for monitoring, analyzing, and exposing Kubernetes container resource usage and performance statistics.

To deploy a cAdvisor container:

1. Select **In-Cluster Monitoring**.
2. Under **Deploy cAdvisor**, select **Yes**.
3. Click **Save**.



**Note:** For information about configuring cAdvisor to monitor your running Kubernetes containers, see [cAdvisor](#) in the cAdvisor GitHub repository. For general information about Kubernetes cluster monitoring, see [Tools for Monitoring Resources](#) in the Kubernetes documentation.

## Metric Sink Resources

You can configure TKGI-provisioned clusters to send Kubernetes node metrics and pod metrics to metric sinks. For more information about metric sink resources and what to do after you enable them in the tile, see [Sink Resources](#) in *Monitoring Workers and Workloads*.

To enable clusters to send Kubernetes node metrics and pod metrics to metric sinks:

1. In **In-Cluster Monitoring**, select **Enable Metric Sink Resources**. If you enable this checkbox, Tanzu Kubernetes Grid Integrated Edition deploys Telegraf as a [DaemonSet](#), a pod that runs on each worker node in all your Kubernetes clusters.
2. (Optional) To enable Node Exporter to send worker node metrics to metric sinks of kind

[ClusterMetricSink](#), select **Enable node exporter on workers**. If you enable this checkbox, Tanzu Kubernetes Grid Integrated Edition deploys Node Exporter as a [DaemonSet](#), a pod that runs on each worker node in all your Kubernetes clusters.

For instructions on how to create a metric sink of kind [ClusterMetricSink](#) for Node Exporter metrics, see [Create a ClusterMetricSink Resource for Node Exporter Metrics](#) in *Creating and Managing Sink Resources*.

3. Click **Save**.

## Log Sink Resources

You can configure TKGI-provisioned clusters to send Kubernetes API events and pod logs to log sinks. For more information about log sink resources and what to do after you enable them in the tile, see [Sink Resources](#) in *Monitoring Workers and Workloads*.

To enable clusters to send Kubernetes API events and pod logs to log sinks:

1. Select **Enable Log Sink Resources**. If you enable this checkbox, Tanzu Kubernetes Grid Integrated Edition deploys Fluent Bit as a [DaemonSet](#), a pod that runs on each worker node in all your Kubernetes clusters.
2. Click **Save**.

## Tanzu Mission Control

Tanzu Mission Control integration lets you monitor and manage Tanzu Kubernetes Grid Integrated Edition clusters from the Tanzu Mission Control console, which makes the Tanzu Mission Control console a single point of control for all Kubernetes clusters. For more information about Tanzu Mission Control, see the [VMware Tanzu Mission Control](#) home page.

To integrate Tanzu Kubernetes Grid Integrated Edition with Tanzu Mission Control:

1. Confirm that the TKGI API VM has internet access and can connect to [cna.tmc.cloud.vmware.com](https://cna.tmc.cloud.vmware.com) and the other outbound URLs listed in the [What Happens When You Attach a Cluster](#) section of the Tanzu Mission Control Product documentation.
2. Navigate to the **Tanzu Kubernetes Grid Integrated Edition** tile > the **Tanzu Mission Control** pane and select **Yes** under **Tanzu Mission Control Integration**.

Tanzu Mission Control Integration\*

No  
 Yes

Tanzu Mission Control URL \*

VMware Cloud Services API Token \*

Tanzu Mission Control Cluster Group \*

Tanzu Mission Control Cluster Name Prefix \*

**Save**

3. Configure the fields below:

- **Tanzu Mission Control URL:** Enter the Org URL of your Tanzu Mission Control subscription, without a trailing slash (/). For example, `YOUR-ORG.tmc.cloud.vmware.com`.
- **VMware Cloud Services API token:** Enter your API token to authenticate with VMware Cloud Services APIs. You can retrieve this token by logging in to [VMware Cloud Services](#) and viewing your account information.
- **Tanzu Mission Control Cluster Group:** Enter the name of a Tanzu Mission Control cluster group.

The name can be `default` or another value, depending on your role and access policy:

- `Org Member` users in VMware cloud services have a `service.admin` role in Tanzu Mission Control. These users:
  - By default, can create and attach clusters only in the `default` cluster group.
  - Can create and attach clusters to other cluster groups after an `organization.admin` user grants them the `clustergroup.admin` or

`clustergroup.edit` role for those groups.

- `Org Owner` users in VMware cloud services have `organization.admin` permissions in Tanzu Mission Control. These users:
  - Can create cluster groups.
  - Can grant `clustergroup` roles to `service.admin` users through the Tanzu Mission Control Access Policy view.

For more information about role and access policy, see [Access Control](#) in the VMware Tanzu Mission Control Product documentation.

- ◊ **Tanzu Mission Control Cluster Name Prefix:** Enter a name prefix for identifying the Tanzu Kubernetes Grid Integrated Edition clusters in Tanzu Mission Control.

4. Click **Save**.



**Warning:** After the Tanzu Kubernetes Grid Integrated Edition tile is deployed with a configured cluster group, the cluster group cannot be updated.



**Note:** When you upgrade your Kubernetes clusters and have Tanzu Mission Control integration enabled, existing clusters will be attached to Tanzu Mission Control.

## VMware CEIP

Tanzu Kubernetes Grid Integrated Edition-provisioned clusters send usage data to the TKGI control plane for storage. The VMware Customer Experience Improvement Program (CEIP) provides the option to also send the cluster usage data to VMware to improve customer experience.

To configure Tanzu Kubernetes Grid Integrated Edition CEIP Program settings:

1. Click **CEIP**.
2. Review the information about the CEIP.

## About the CEIP Program

VMware's Customer Experience Improvement Program ("CEIP") provides VMware with information that enables VMware to improve its products and services, to fix problems, and to advise you on how best to deploy and use our products. As part of the CEIP, VMware collects technical information about your organization's use of VMware products and services on a regular basis in association with your organization's VMware license key(s). This information does not personally identify any individual.

Customers who participate in the CEIP receive proactive support benefits that include a weekly report based on telemetry data. Contact your Customer Success Manager to subscribe to this report. You can view a sample report at <https://via.vmware.com/TKGI>.

Additional information regarding the data collected through CEIP and the purposes for which it is used by VMware is set forth in the Trust & Assurance Center at <http://www.vmware.com/trustvmware/ceip.html>. If you prefer not to participate in VMware's CEIP for this product, you should uncheck the box below. You may join or leave VMware's CEIP for this product at any time.

Join the VMware Customer Experience Improvement Program\*

- No
- Yes

Please enter your Entitlement Account Number or your Tanzu Customer Number. We need this information to generate a report for you

Please enter a label for this TKGI Installation

Assign a label to this TKGI Installation for use in your reports

Please select how you will be using this TKGI Installation \*

- Demo or Proof-of-concept
- Development or Pre-production
- Production
- I do not wish to provide this information

**Save**

[View a larger version of this image.](#)

3. If you wish to participate in CEIP, select **Yes**. Otherwise, select **No**.
4. If you selected the **Yes**, complete the following:
  - ◊ (Optional) Enter your entitlement account number or Tanzu customer number. If you are a VMware customer, you can find your entitlement account number in your **Account Summary** on [my.vmware.com](https://my.vmware.com). If you are a Pivotal customer, you can find your Pivotal Customer Number in your Pivotal Order Confirmation email.
  - ◊ (Optional) Enter a descriptive name for your TKGI installation. The label you assign to this installation will be used in CEIP reports to identify the environment.
5. To provide information about the purpose for this installation, select an option.

Please select how you will be using this TKGI Installation \*

- Demo or Proof-of-concept
- Development or Pre-production
- Production
- I do not wish to provide this information

6. Click **Save**.

## Errands

Errands are scripts that run at designated points during an installation.

To configure which post-deploy and pre-delete errands run for Tanzu Kubernetes Grid Integrated Edition:

1. Make a selection in the dropdown next to each errand.



**Note:** We recommend that you use the default settings for all errands except for the **Run smoke tests** errand.

2. (Optional) Set the **Run smoke tests** errand to **On**.

This errand uses the TKGI CLI to create a Kubernetes cluster and then delete it. If the creation or deletion fails, the errand fails and the installation of the Tanzu Kubernetes Grid Integrated Edition tile is aborted.

3. (Optional) To ensure that all of your cluster VMs are patched, configure the **Upgrade all clusters errand** errand to **On**.

Updating the Tanzu Kubernetes Grid Integrated Edition tile with a new Linux stemcell and the **Upgrade all clusters errand** enabled triggers the rolling of every Linux VM in each Kubernetes cluster. Similarly, updating the Tanzu Kubernetes Grid Integrated Edition tile with a new Windows stemcell triggers the rolling of every Windows VM in your Kubernetes clusters.



**Note:** VMware recommends that you review the Tanzu Network metadata and confirm stemcell version compatibility before using the Tanzu Network APIs to update the stemcells in your automated pipeline. For more information, see the [API reference](#).

## Resource Config

To modify the resource configuration of Tanzu Kubernetes Grid Integrated Edition and specify your TKGI API load balancer, follow the steps below:

1. Select **Resource Config**.
2. For each job, review the **Automatic** values in the following fields:
  - ❖ **INSTANCES:** Tanzu Kubernetes Grid Integrated Edition defaults to the minimum configuration. If you want a highly available configuration (beta), scale the number of VM instances as follows:
    1. To configure your Tanzu Kubernetes Grid Integrated Edition database for high availability (beta), increase the **INSTANCES** value for **TKGI Database** to **3**.
    2. To configure your Tanzu Kubernetes Grid Integrated Edition API and UAA for high availability (beta), increase the **INSTANCES** value for **TKGI API** to **2** or more.



**Warning:** High availability mode is a beta feature. Do not scale your **TKGI API** or **TKGI Database** to more than one instance in production environments.



**Note:** On Azure, you must reconfigure your TKGI API load balancer backend pool whenever you modify your TKGI API VM group. For more information about configuring your TKGI API load balancer backend pool, see [Create a Load Balancer](#)

in *Configuring an Azure Load Balancer for the TKGI API.*

- ◊ **VM TYPE:** By default, the **TKGI Database** and **TKGI API** jobs are set to the same **Automatic** VM type. If you want to adjust this value, we recommend that you select the same VM type for both jobs.



**Note:** The **Automatic VM TYPE** values match the recommended resource configuration for the **TKGI API** and **TKGI Database** jobs.

- ◊ **PERSISTENT DISK TYPE:** By default, the **TKGI Database** and **TKGI API** jobs are set to the same persistent disk type. If you want to adjust this value, you can change the persistent disk type for each of the jobs independently. Using the same persistent disk type for both jobs is not required.

3. For the **TKGI Database** job:

- ◊ Leave the **LOAD BALancers** field blank.
- ◊ (Optional) If you do not use a NAT instance, select **INTERNET CONNECTED**. This allows component instances direct access to the internet.

4. For the **TKGI API** job:

- ◊ Enter the name of your TKGI API load balancer in the **LOAD BALancers** field. For more information on the TKGI API load balancer, see [Configuring an Azure Load Balancer for the TKGI API](#).



**Note:** After you click **Apply Changes** for the first time, BOSH assigns the TKGI API VM an IP address. BOSH uses the name you provide in the **LOAD BALancers** field to locate your load balancer and then connect the load balancer to the TKGI API VM using its new IP address.

\* (Optional) If you do not use a NAT instance, select **INTERNET CONNECTED**. This allows component instances direct access to the internet.



**Warning:** To avoid workload downtime, use the resource configuration recommended in [About Tanzu Kubernetes Grid Integrated Edition Upgrades](#) and [Maintaining Workload Uptime](#).

## Step 3: Apply Changes

1. Return to the Ops Manager Installation Dashboard.
2. Click **Review Pending Changes**. Select the product that you intend to deploy and review the changes. For more information, see [Reviewing Pending Product Changes](#).
3. Click **Apply Changes**.

## Step 4: Retrieve the TKGI API Endpoint

You need to retrieve the TKGI API endpoint to allow your organization to use the API to create, update, and delete Kubernetes clusters.

To retrieve the TKGI API endpoint, do the following:

1. Navigate to the Ops Manager [Installation Dashboard](#).
2. Click the **Tanzu Kubernetes Grid Integrated Edition** tile.
3. Click the **Status** tab and locate the **TKGI API** job. The IP address of the TKGI API job is the TKGI API endpoint.

## Step 5: Configure an Azure Load Balancer for the TKGI API

Follow the procedures in [Configuring an Azure Load Balancer for the TKGI API](#) to configure an Azure load balancer for the TKGI API.

## Step 6: Install the TKGI and Kubernetes CLIs

The TKGI CLI and the Kubernetes CLI help you interact with your Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes clusters and Kubernetes workloads. To install the CLIs, follow the instructions below:

- [Installing the TKGI CLI](#)
- [Installing the Kubernetes CLI](#)

## Step 7: Configure Authentication for Tanzu Kubernetes Grid Integrated Edition

Follow the procedures in [Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on Azure](#).

## Next Steps

After installing Tanzu Kubernetes Grid Integrated Edition on Azure, you might want to do one or more of the following:

- Create a load balancer for your Tanzu Kubernetes Grid Integrated Edition clusters. For more information, see [Creating and Configuring an Azure Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters](#).
- Create your first Tanzu Kubernetes Grid Integrated Edition cluster. For more information, see [Creating Clusters](#).

## Configuring an Azure Load Balancer for the TKGI API

This topic describes how to create a load balancer for the VMware Tanzu Kubernetes Grid Integrated Edition API using Azure.

Refer to the procedures in this topic to create a load balancer using Azure. To use a different load balancer, use this topic as a guide.

## Overview

VMware recommends that you create a TKGI API load balancer when installing Tanzu Kubernetes Grid Integrated Edition on Azure. You simplify future upgrades of Tanzu Kubernetes Grid Integrated Edition by creating a load balancer when installing.

To configure your TKGI API Load Balancer on Azure, complete the following:

- [Create a Load Balancer](#)
- [Create a Backend Pool](#)
- [Create Health Probes](#)
- [Create Load Balancing Rules](#)
- [Create an Inbound Security Rule](#)
- [Add the TKGI API to the Backend Pool](#)
- [Verify TKGI API Hostname Resolution](#)

## Create a Load Balancer

To create a new load balancer:

1. In a browser, navigate to the [Azure Dashboard](#).
2. Open the **Load Balancers** service.
3. To add a new load balancer, click **Add** and complete the **Create load balancer** form as follows:
  1. **Name:** Enter a name for the load balancer.
  2. **Type:** Select **Public**.
  3. **SKU:** Select **Standard**.
  4. **Public IP address:** Select **Create new**.
  5. **Public IP address name:** Enter the name for the new IP address.
  6. **Availability zone:** Select an availability zone or **Zone-redundant**.
  7. **Subscription:** Select the subscription where Tanzu Kubernetes Grid Integrated Edition has been deployed.
  8. **Resource group:** Select the resource group where Tanzu Kubernetes Grid Integrated Edition has been deployed.
  9. **Location:** Select the location group where Tanzu Kubernetes Grid Integrated Edition has been deployed.
  10. Click **Create**.

## Create a Backend Pool

An Azure backend pool is a logical grouping of instances that receive similar traffic.

To create a backend pool for your load balancer:

1. From the Azure Dashboard, open the **Load Balancers** service.
2. Click the name of the load balancer that you created in [Create Load Balancer](#) above.
3. On your load balancer page, locate and record the IP address of your load balancer.
4. In the **Settings** menu, select **Backend pools**.
5. To add a new backend pool, click **Add** and complete the **Backend pools** form as follows:
  1. **Name:** Enter the name for the backend pool.
  2. Click **Add**.

For information about Azure backend pools, see [Backend pools](#) in the Azure documentation. For more information about configuring your backend pool, see [Remove or add VMs from the backend pool](#) in the Azure documentation.

## Create Health Probes

To create health probes for your load balancer and UAA:

1. From the Azure Dashboard, open the **Load Balancers** service.
2. To open the **Health probes** page, select **Health probes** in the **Settings** menu.
3. To create a new TKGI API server health probe, click **Add** and complete the form as follows:
  1. **Name:** Enter the name for the health probe.
  2. **Protocol:** Select **TCP**.
  3. **Port:** Enter **9021**.
  4. **Interval:** Enter the interval of time to wait between probe attempts.
  5. **Unhealthy Threshold:** Enter a number of consecutive probe failures that must occur before a VM is considered unhealthy.
4. To create a new UAA health probe, click **Add** and complete the form as follows:
  1. **Name:** Enter the name for the UAA health probe.
  2. **Protocol:** Select **TCP**.
  3. **Port:** Enter **8443**.
  4. **Interval:** Enter the interval of time to wait between probe attempts.
  5. **Unhealthy Threshold:** Enter a number of consecutive probe failures that must occur before a VM is considered unhealthy.
5. Click **OK**.

## Create Load Balancing Rules

To create load balancer rules for your load balancer:

1. From the Azure Dashboard, open the **Load Balancers** service.
2. To open the **Load balancing rules** page, select **Load Balancing Rules** in the **Settings** menu.
3. To create a new TKGI API server load balancer rule, click **Add** and complete the **Add load**

**balancing rules** form as follows:

1. **Name:** Enter a name for the load balancing rule.
  2. **IP Version:** Select **IPv4**.
  3. **Frontend IP address:** Select the appropriate IP address. Clients communicate with your load balancer on the selected IP address and service traffic is routed to the target VM by this NAT rule.
  4. **Protocol:** Select **TCP**.
  5. **Port:** Enter **9021**.
  6. **Backend port:** Enter **9021**.
  7. **Health Probe:** Select the TKGI API server health probe that you created in [Create Health Probe](#) above.
  8. **Session persistence:** Select **None**.
4. To create a new UAA load balancer rule, click **Add** and complete the **Add load balancing rules** form as follows:
1. **Name:** Enter a name for the UAA load balancing rule.
  2. **IP Version:** Select **IPv4**.
  3. **Frontend IP address:** Select the appropriate IP address. Clients communicate with your load balancer on the selected IP address and service traffic is routed to the target VM by this NAT rule.
  4. **Protocol:** Select **TCP**.
  5. **Port:** Enter **8443**.
  6. **Backend port:** Enter **8443**.
  7. **Health Probe:** Select the UAA health probe that you created in [Create Health Probe](#) above.
  8. **Session persistence:** Select **None**.
5. Click **OK**.

## Create an Inbound Security Rule

To create an inbound security rule for your load balancer:

1. From the Azure Dashboard, open the **Network Security Groups** service.
2. Click the name of the Security Group attached to the subnet where the TKGI API is deployed.
3. To open the **Inbound security rules** page, select **Inbound security rules** in the **Settings** menu for your security group.
4. To add a new inbound security rule, click **Add** and complete the **Add inbound security rule** form as follows:
  1. Click **Advanced**.
  2. **Name:** Enter the name for the inbound security rule.

3. **Source:** Select **Any**.
4. **Source port range:** Enter **\***.
5. **Destination:** Select **Any**.
6. **Destination port range:** Enter **9021,8443**.
7. Click **OK**.

## Add the TKGI API to the Backend Pool

To assign a load balancer to the TKGI API VM and add the TKGI API VM to the backend pool:

1. Open Ops Manager to the **Installation Dashboard** pane.
2. Click the **Tanzu Kubernetes Grid Integrated Edition** tile.
3. Open the **Resource Config** pane.
4. Select **TKGI API**.
5. Review **Load Balancers**.
6. If **Load Balancers** does not include the load balancer to use for the TKGI API VM:
  1. Input the load balancer to use for TKGI API VM.
  2. Click **Apply Changes**.

For information about Azure backend pools, see [Backend pools](#) in the Azure documentation. For more information about configuring your backend pool, see [Remove or add VMs from the backend pool](#) in the Azure documentation.

## Verify TKGI API Hostname Resolution

To verify that your TKGI API hostname resolves correctly:

1. Open Ops Manager to the **Installation Dashboard** pane.
2. Click the **Tanzu Kubernetes Grid Integrated Edition** tile.
3. Select **TKGI API**.
4. Record the **API Hostname (FQDN)**.
5. Verify that the TKGI API hostname resolves to the IP address of the load balancer.

## Next Step

After you have configured an Azure load balancer for the TKGI API, complete the Tanzu Kubernetes Grid Integrated Edition installation by returning to the [Install the TKGI and Kubernetes CLIs](#) step of *Installing Tanzu Kubernetes Grid Integrated Edition on Azure*.

## Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users on Azure

This topic describes how to create admin users in VMware Tanzu Kubernetes Grid Integrated Edition

with User Account and Authentication (UAA). Creating at least one admin user is a necessary step during the initial set up of Tanzu Kubernetes Grid Integrated Edition.

## Overview

UAA is the identity management service for Tanzu Kubernetes Grid Integrated Edition. Tanzu Kubernetes Grid Integrated Edition includes a UAA server, which is hosted on the TKGI API VM.

To interact with the UAA server, you can use the UAA Command Line Interface (UAAC). You can either run UAAC commands from the Ops Manager VM or install UAAC on your local workstation.

## Prerequisites

Before setting up admin users for Tanzu Kubernetes Grid Integrated Edition, you must have one of the following:

- SSH access to the Ops Manager VM
- A machine that can connect to your TKGI API VM

## Step 1: Connect to the TKGI API VM

You can connect to the TKGI API VM from the Ops Manager VM or from a different machine such as your local workstation.

### Option 1: Connect through the Ops Manager VM

You can connect to the TKGI API VM by logging in to the Ops Manager VM through SSH.

To log in to the Ops Manager VM using SSH on Azure, you need the SSH key pair you used when you created the Ops Manager VM. If you need to reset the SSH key, locate the Ops Manager VM in the Azure portal and click **Reset Password**.

To SSH into the Ops Manager VM on Azure, do the following:

1. From the Azure portal, locate the Ops Manager FQDN by selecting the VM.
2. Change the permissions for your SSH private key by running the following command:

```
chmod 600 PRIVATE-KEY
```

Where **PRIVATE-KEY** is the name of your SSH private key.

3. SSH into the Ops Manager VM by running the following command:

```
ssh -i PRIVATE-KEY ubuntu@OPS-MANAGER-FQDN
```

Where:

- **OPS-MANAGER-FQDN** is FQDN of Ops Manager.
- **PRIVATE-KEY** is the name of your SSH private key.

For example:

```
$ ssh -i id_rsa ubuntu@my-opsmanager-fqdn.example.com
```

4. Proceed to the [Log in as a UAA Admin](#) section to manage users with UAAC.

## Option 2: Connect through a Non-Ops Manager Machine

To connect to the TKGI API VM and run UAA commands, do the following:

1. Install UAAC on your machine. For example:

```
gem install cf-uaac
```

2. Download a copy of your Ops Manager root CA certificate to the machine. To download the certificate, do the following:
  1. In a web browser, navigate to the FQDN of Ops Manager and log in.
  2. In Ops Manager, navigate to **Settings** in the drop-down menu under your username.
  3. Click **Advanced Options**.
  4. On the **Advanced Options** configuration page, click **Download Root CA Cert**.
  5. Move the certificate to a secure location on your machine and record the path.
3. Proceed to the [Log In as a UAA Admin](#) section to create admin users with UAAC.

## Step 2: Log In as a UAA Admin

Before creating TKGI users, you must log in to the UAA server as a UAA admin. To log in to the UAA server, do the following:

1. Retrieve the UAA management admin client secret:
  1. In a web browser, navigate to the Ops Manager **Installation Dashboard** and click the **Tanzu Kubernetes Grid Integrated Edition** tile.
  2. Click the **Credentials** tab.
  3. Click **Link to Credential** next to **Pks Uaa Management Admin Client** and copy the value of **secret**.
2. Target your UAA server by running the following command:

```
uaac target https://TKGI-API:8443 --ca-cert CERTIFICATE-PATH
```

Where:

- ◊ **TKGI-API** is the domain name of your TKGI API server. You entered this domain name in the **Tanzu Kubernetes Grid Integrated Edition** tile > **TKGI API > API Hostname (FQDN)**.
- ◊ **CERTIFICATE-PATH** is the path to your Ops Manager root CA certificate. Provide this certificate to validate the TKGI API certificate with SSL.
  - If you are logged in to the Ops Manager VM, specify **/var/tempest/workspaces/default/root\_ca\_certificate** as the path. This is the default location of the root certificate on the Ops Manager VM.

- If you downloaded the Ops Manager root CA certificate to your machine, specify the path where you stored the certificate.

For example:

```
$ uaac target api.tkgi.example.com:8443 -ca-cert /var/tempest/wo
rkspaces/default/root_ca_certificate
```



**Note:** If you receive an `Unknown key: Max-Age = 86400` warning message, you can ignore it because it has no impact.

3. Authenticate with UAA by running the following command:

```
uaac token client get admin -s ADMIN-CLIENT-SECRET
```

Where `ADMIN-CLIENT-SECRET` is your UAA management admin client secret that you retrieved in a previous step. The client username is `admin`.

## Step 3: Assign Tanzu Kubernetes Grid Integrated Edition Cluster Scopes

The `pks.clusters.manage` and `pks.clusters.admin` UAA scopes grant users the ability to create and manage Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition. For information about UAA scopes in Tanzu Kubernetes Grid Integrated Edition, see [UAA Scopes for Tanzu Kubernetes Grid Integrated Edition Users](#).

To create Tanzu Kubernetes Grid Integrated Edition users with the `pks.clusters.manage` or `pks.clusters.admin` UAA scope, perform one or more of the following procedures based on the needs of your deployment:

- To assign TKGI cluster scopes to an individual user, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an Individual User](#). Follow this procedure if you selected **Internal UAA** when you configured **UAA** in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Installing Tanzu Kubernetes Grid Integrated Edition on Azure](#).
- To assign TKGI cluster scopes to an LDAP group, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an External LDAP Group](#). Follow this procedure if you selected **LDAP Server** when you configured **UAA** in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Installing Tanzu Kubernetes Grid Integrated Edition TKGI on Azure](#).
- To assign TKGI cluster scopes to a SAML group, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an External SAML Group](#). Follow this procedure if you selected **SAML Identity Provider** when you configured **UAA** in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Installing Tanzu Kubernetes Grid Integrated Edition TKGI on Azure](#).
- To assign TKGI cluster scopes to a client, see [Grant Tanzu Kubernetes Grid Integrated](#)

[Edition Access to a Client.](#)

## Next Step

After you create admin users in Tanzu Kubernetes Grid Integrated Edition, the admin users can create and manage Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition. For more information, see [Managing Kubernetes Clusters and Workloads](#).

## Firewall Ports and Protocols Requirements (Antrea and Flannel Networking)

This topic describes the firewall ports and protocols requirements for using VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) with Antrea and Flannel container networking.

If you are using TKGI on vSphere, see one of the follow topics instead:

- [Firewall Ports and Protocols Requirements for vSphere with NSX-T](#)
- [Firewall Ports and Protocols Requirements for vSphere \(Antrea and Flannel Networking\)](#)

## Overview

Apps frequently require the ability to pass internal communication between system components on different networks.

Firewalls and Kubernetes Pod Security Policy are used to filter traffic and limit access in environments with strict inter-network access control policies and your apps require one or more conduits through a secured environment's firewalls.

VMware recommends that rather than using a Kubernetes Pod Security Policy to filter traffic between networks and TKGI system components and clusters that you instead enable access to apps through standard Kubernetes load-balancers and ingress controller types. This enables you to designate specific ports and protocols as a firewall conduit.

Consult the following tables when configuring port settings to install or upgrade TKGI or configure a Kubernetes cluster:

- [TKGI Users Ports and Protocols](#)
- [TKGI Core Ports and Protocols](#)
- [Antrea Networking Ports and Protocols](#)



**Note:** To control which groups access deploying and scaling your organization's Tanzu Kubernetes Grid Integrated Edition-deployed Kubernetes clusters, configure your firewall settings as described on the Operator → TKGI API server lines below.

## TKGI Ports and Protocols

The following tables list ports and protocols required for network communications between Tanzu Kubernetes Grid Integrated Edition v1.5.0 and later, and other components.

### TKGI Users Ports and Protocols

The following table lists ports and protocols used for network communication between TKGI user interface components.

| Source Component                                  | Destination Component                 | Destination Protocol | Destination Port | Service                |
|---------------------------------------------------|---------------------------------------|----------------------|------------------|------------------------|
| Admin/Operator Console                            | All System Components                 | TCP                  | 22               | ssh                    |
| Admin/Operator Console                            | All System Components                 | TCP                  | 80               | http                   |
| Admin/Operator Console                            | All System Components                 | TCP                  | 443              | https                  |
| Admin/Operator Console                            | BOSH Director                         | TCP                  | 25555            | bosh director rest api |
| Admin/Operator Console                            | Ops Manager                           | TCP                  | 22               | ssh                    |
| Admin/Operator Console                            | Ops Manager                           | TCP                  | 443              | https                  |
| Admin/Operator Console                            | TKGI Controller                       | TCP                  | 9021             | tkgi api server        |
| Admin/Operator and Developer Consoles             | Harbor Private Image Registry         | TCP                  | 80               | http                   |
| Admin/Operator and Developer Consoles             | Harbor Private Image Registry         | TCP                  | 443              | https                  |
| Admin/Operator and Developer Consoles             | Harbor Private Image Registry         | TCP                  | 4443             | notary                 |
| Admin/Operator and Developer Consoles             | Kubernetes App Load-Balancer Svc      | TCP/UDP              | Varies           | varies with apps       |
| Admin/Operator and Developer Consoles             | Kubernetes Cluster API Server -LB VIP | TCP                  | 8443             | httpsca                |
| Admin/Operator and Developer Consoles             | Kubernetes Cluster Ingress Controller | TCP                  | 80               | http                   |
| Admin/Operator and Developer Consoles             | Kubernetes Cluster Ingress Controller | TCP                  | 443              | https                  |
| Admin/Operator and Developer Consoles             | Kubernetes Cluster Worker Node        | TCP/UDP              | 30000-32767      | kubernetes nodeport    |
| Admin/Operator and Developer Consoles             | TKGI Controller                       | TCP                  | 8443             | httpsca                |
| All User Consoles (Operator, Developer, Consumer) | Kubernetes App Load-Balancer Svc      | TCP/UDP              | Varies           | varies with apps       |

| Source Component                                  | Destination Component                 | Destination Protocol | Destination Port | Service             |
|---------------------------------------------------|---------------------------------------|----------------------|------------------|---------------------|
| All User Consoles (Operator, Developer, Consumer) | Kubernetes Cluster Ingress Controller | TCP                  | 80               | http                |
| All User Consoles (Operator, Developer, Consumer) | Kubernetes Cluster Ingress Controller | TCP                  | 443              | https               |
| All User Consoles (Operator, Developer, Consumer) | Kubernetes Cluster Worker Node        | TCP/UDP              | 30000-32767      | kubernetes nodeport |

## TKGI Core Ports and Protocols

The following table lists ports and protocols used for network communication between core TKGI components.

| Source Component                    | Destination Component                      | Destination Protocol | Destination Port | Service                |
|-------------------------------------|--------------------------------------------|----------------------|------------------|------------------------|
| All System Components               | Corporate Domain Name Server               | TCP/UDP              | 53               | dns                    |
| All System Components               | Network Time Server                        | UDP                  | 123              | ntp                    |
| All System Control Plane Components | AD/LDAP Directory Server                   | TCP/UDP              | 389/636          | ldap/ldaps             |
| Ops Manager                         | Admin/Operator Console                     | TCP                  | 22               | ssh                    |
| Ops Manager                         | BOSH Director                              | TCP                  | 6868             | bosh agent http        |
| Ops Manager                         | BOSH Director                              | TCP                  | 8443             | httpsca                |
| Ops Manager                         | BOSH Director                              | TCP                  | 8844             | credhub                |
| Ops Manager                         | BOSH Director                              | TCP                  | 25555            | bosh director rest api |
| Ops Manager                         | Harbor Private Image Registry              | TCP                  | 22               | ssh                    |
| Ops Manager                         | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 22               | ssh                    |
| Ops Manager                         | Kubernetes Cluster Worker Node             | TCP                  | 22               | ssh                    |
| Ops Manager                         | TKGI Controller                            | TCP                  | 22               | ssh                    |
| Ops Manager                         | TKGI Controller                            | TCP                  | 8443             | httpsca                |
| BOSH Compilation Job VM             | BOSH Director                              | TCP                  | 4222             | bosh nats server       |
| BOSH Compilation Job VM             | BOSH Director                              | TCP                  | 25250            | bosh blobstore         |
| BOSH Compilation Job VM             | BOSH Director                              | TCP                  | 25923            | health monitor daemon  |
| BOSH Compilation Job VM             | Harbor Private Image Registry              | TCP                  | 443              | https                  |
| BOSH Compilation Job VM             | Harbor Private Image Registry              | TCP                  | 8853             | bosh dns health        |

| Source Component                           | Destination Component                      | Destination Protocol | Destination Port | Service                |
|--------------------------------------------|--------------------------------------------|----------------------|------------------|------------------------|
| TKGI Controller                            | BOSH Director                              | TCP                  | 4222             | bosh nats server       |
| TKGI Controller                            | BOSH Director                              | TCP                  | 8443             | httpsca                |
| TKGI Controller                            | BOSH Director                              | TCP                  | 25250            | bosh blobstore         |
| TKGI Controller                            | BOSH Director                              | TCP                  | 25555            | bosh director rest api |
| TKGI Controller                            | BOSH Director                              | TCP                  | 25923            | health monitor daemon  |
| TKGI Controller                            | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 8443             | httpsca                |
| TKGI Controller                            | TKGI Database VM                           | TCP                  | 3306             | tkgi db proxy          |
| Harbor Private Image Registry              | BOSH Director                              | TCP                  | 4222             | bosh nats server       |
| Harbor Private Image Registry              | BOSH Director                              | TCP                  | 25250            | bosh blobstore         |
| Harbor Private Image Registry              | BOSH Director                              | TCP                  | 25923            | health monitor daemon  |
| Harbor Private Image Registry              | IP NAS Storage Array                       | TCP                  | 111              | nfs rpc portmapper     |
| Harbor Private Image Registry              | IP NAS Storage Array                       | TCP                  | 2049             | nfs                    |
| Harbor Private Image Registry              | Public CVE Source Database                 | TCP                  | 443              | https                  |
| kube-system pod/telemetry-agent            | TKGI Controller                            | TCP                  | 24224            | fluentd out_forward    |
| Kubernetes Cluster Control Plane/Etcd Node | BOSH Director                              | TCP                  | 4222             | bosh nats server       |
| Kubernetes Cluster Control Plane/Etcd Node | BOSH Director                              | TCP                  | 25250            | bosh blobstore         |
| Kubernetes Cluster Control Plane/Etcd Node | BOSH Director                              | TCP                  | 25923            | health monitor daemon  |
| Kubernetes Cluster Control Plane/Etcd Node | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 2379             | etcd client            |
| Kubernetes Cluster Control Plane/Etcd Node | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 2380             | etcd server            |
| Kubernetes Cluster Control Plane/Etcd Node | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 8443             | httpsca                |
| Kubernetes Cluster Control Plane/Etcd Node | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 8853             | bosh dns health        |
| Kubernetes Cluster Control Plane/Etcd Node | Kubernetes Cluster Worker Node             | TCP                  | 4194             | cadvisor               |
| Kubernetes Cluster Control Plane/Etcd Node | Kubernetes Cluster Worker Node             | TCP                  | 10250            | kubelet api            |

| Source Component                           | Destination Component                      | Destination Protocol | Destination Port | Service               |
|--------------------------------------------|--------------------------------------------|----------------------|------------------|-----------------------|
| Kubernetes Cluster Control Plane/Etcd Node | Kubernetes Cluster Worker Node             | TCP                  | 31194            | cadvisor              |
| Kubernetes Cluster Control Plane/Etcd Node | TKGI Controller                            | TCP                  | 8443             | httpsca               |
| Kubernetes Cluster Control Plane/Etcd Node | TKGI Controller                            | TCP                  | 8853             | bosh dns health       |
| Kubernetes Cluster Worker Node             | BOSH Director                              | TCP                  | 4222             | bosh nats server      |
| Kubernetes Cluster Worker Node             | BOSH Director                              | TCP                  | 25250            | bosh blobstore        |
| Kubernetes Cluster Worker Node             | BOSH Director                              | TCP                  | 25923            | health monitor daemon |
| Kubernetes Cluster Worker Node             | Harbor Private Image Registry              | TCP                  | 443              | https                 |
| Kubernetes Cluster Worker Node             | Harbor Private Image Registry              | TCP                  | 8853             | bosh dns health       |
| Kubernetes Cluster Worker Node             | IP NAS Storage Array                       | TCP                  | 111              | nfs rpc portmapper    |
| Kubernetes Cluster Worker Node             | IP NAS Storage Array                       | TCP                  | 2049             | nfs                   |
| Kubernetes Cluster Worker Node             | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 8443             | httpsca               |
| Kubernetes Cluster Worker Node             | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 8853             | bosh dns health       |
| Kubernetes Cluster Worker Node             | Kubernetes Cluster Control Plane/Etcd Node | TCP                  | 10250            | kubelet api           |
| pks-system pod/cert-generator              | TKGI Controller                            | TCP                  | 24224            | fluentd out_forward   |
| pks-system pod/fluent-bit                  | TKGI Controller                            | TCP                  | 24224            | fluentd out_forward   |

## Networking Ports and Protocols

The following tables list ports and protocols required for network communication.

## Antrea Networking Ports and Protocols

The following tables list ports and protocols required for network communication in Antrea

environments.

| Source Component       | Destination Component  | Destination Protocol | Destination Port | Service |
|------------------------|------------------------|----------------------|------------------|---------|
| Worker Node VMs        | Worker Node VMs        | UDP                  | 6081             | Geneve  |
| Control Plane Node VMs | Control Plane Node VMs | TCP                  | 8091             | TCP     |



**Note:** Port 6081 must be open on all of the worker node VMs and port 8091 must be open on all control plane node VMs in the clusters you create in an Antrea networking environment.

For more information, see [Network Requirements](#) in the Antrea GitHub repository.

## Installing the TKGI CLI

This topic describes how to install the VMware Tanzu Kubernetes Grid Integrated Edition Command Line Interface (TKGI CLI).

For information on the TKGI CLI, see [TKGI CLI](#).

## Install the TKGI CLI

To install the TKGI CLI, follow the procedures for your operating system to download the TKGI CLI from [VMware Tanzu Network](#). Binaries are only provided for 64-bit architectures.

### Mac OS X

1. Navigate to [VMware Tanzu Network](#) and log in.
2. Click **Tanzu Kubernetes Grid Integrated Edition**.
3. Select your desired release version from the **Releases** dropdown.
4. Click **TKGI CLI**.
5. Click **TKGI CLI - Mac** to download the Mac OS X binary.
6. Rename the downloaded binary file to `tkgi`.
7. On the command line, run the following command to make the TKGI CLI binary executable:
 

```
$ chmod +x tkgi
```
8. Move the binary file into your `PATH`.
9. Run `tkgi --version` to verify the version of your TKGI CLI installed locally.

### Linux

1. Navigate to [VMware Tanzu Network](#) and log in.
2. Click **Tanzu Kubernetes Grid Integrated Edition**.
3. Select your desired release version from the **Releases** dropdown.
4. Click **TKGI CLI**.

5. Click **TKGI CLI - Linux** to download the Linux binary.
6. Rename the downloaded binary file to `tkgi`.
7. On the command line, run the following command to make the TKGI CLI binary executable:

```
$ chmod +x tkgi
```
8. Move the binary file into your `PATH`.
9. Run `tkgi --version` to verify the version of your TKGI CLI installed locally.

## Windows

1. Navigate to [VMware Tanzu Network](#) and log in.
2. Click **Tanzu Kubernetes Grid Integrated Edition**.
3. Select your desired release version from the **Releases** dropdown.
4. Click **TKGI CLI**.
5. Click **TKGI CLI - Windows** to download the Windows executable file.
6. Rename the downloaded binary file to `tkgi.exe`.
7. Move the binary file into your `PATH`.
8. Run `tkgi --version` to verify the version of your TKGI CLI installed locally.

## Installing the Kubernetes CLI

This topic describes how to install the Kubernetes Command Line Interface (kubectl).

To install kubectl, follow the procedures for your operating system to download kubectl from [VMware Tanzu Network](#). Binaries are only provided for 64-bit architectures.

## Mac OS X

1. Navigate to [VMware Tanzu Network](#) and log in.
2. Click **VMware Tanzu Kubernetes Grid Integrated Edition**.
3. Click **Kubectl CLIs**.
4. Click **kubectl CLI - Mac** to download the kubectl binary.
5. Rename the downloaded binary to `kubectl`.
6. On the command line, run the following command to make the kubectl binary executable:

```
$ chmod +x kubectl
```
7. Move the binary into your `PATH`. For example:

```
$ mv kubectl /usr/local/bin/kubectl
```

## Linux

1. Navigate to [VMware Tanzu Network](#) and log in.
2. Click **VMware Tanzu Kubernetes Grid Integrated Edition**.
3. Click **Kubectl CLIs**.
4. Click **kubectl CLI - Linux** to download the kubectl binary.
5. Rename the downloaded binary to `kubectl1`.
6. On the command line, run the following command to make the kubectl binary executable:

```
$ chmod +x kubectl
```
7. Move the binary into your `PATH`. For example:

```
$ mv kubectl /usr/local/bin/kubectl
```

## Windows

1. Navigate to [VMware Tanzu Network](#) and log in.
2. Click **VMware Tanzu Kubernetes Grid Integrated Edition**.
3. Click **Kubectl CLIs**.
4. Click **kubectl CLI - Windows** to download the kubectl executable file.
5. Rename the downloaded binary to `kubectl.exe`.
6. Move the binary into your `PATH`.

# Upgrading Tanzu Kubernetes Grid Integrated Edition

This section describes how to upgrade VMware Tanzu Kubernetes Grid Integrated Edition. See the following topics:

- [Upgrade Tanzu Kubernetes Grid Integrated Edition Management Console](#)
- [Upgrading Tanzu Kubernetes Grid Integrated Edition with Ops Manager](#)

## Upgrade Tanzu Kubernetes Grid Integrated Edition Management Console

To upgrade an existing installation of Tanzu Kubernetes Grid Integrated Edition Management Console, you download and deploy a new version of the Tanzu Kubernetes Grid Integrated Edition Management Console VM. You then use the management console UI of the new version to migrate the configuration of the old installation to the new one.

You can only use the management console to upgrade an Tanzu Kubernetes Grid Integrated Edition installation that was deployed from a previous version of the management console. You cannot use the console to upgrade an instance of Tanzu Kubernetes Grid Integrated Edition that you installed manually.

## Prerequisites

- You have deployed and configured an older version of Tanzu Kubernetes Grid Integrated Edition Management Console.
- Download the new version of the Tanzu Kubernetes Grid Integrated Edition Management Console OVA template from <https://downloads.vmware.com>.
- Use an account with vSphere administrator privileges to log in to vSphere using the vSphere Client.
- (Optional) If you deployed the old version of the Tanzu Kubernetes Grid Integrated Edition Management Console with a static IP address, and you want the new version to retain the same IP address after the upgrade, reconfigure the old Management Console VM to use a temporary IP address before you start the upgrade procedure:
  1. Shut down the previous version of the management console VM by selecting **Shutdown Guest OS**.



**WARNING:** Do not select **Power Off**.

2. Access the vApp options for the TKGI Management Console VM:
  1. Select the TKGI Management Console VM.
  2. Select the **Configure** tab > **vApp Options**.
  3. Scroll to the **Properties** section.
3. Set a temporary IP address on the TKGI Management Console VM:
  1. Select the row for **2.1. Network IP Address**.
  2. Select **Set Value**.
4. Save your changes and power the TKGI Management Console VM back on.

## Step 1: Deploy the New OVA Template

Follow the instructions in [Deploy the Tanzu Kubernetes Grid Integrated Edition Management Console](#) to deploy and power on the new version of the management console VM from the new OVA template.



**Notes:** If you want to reuse the same IP address as before, and you assigned a temporary IP address to the old version of the management console, configure the network settings of the new Management Console to use the same static IP address as previously.

If you used custom certificates when you deployed the previous version of the management console, you must use the same certificates when you deploy the new version of the management console. If you do not provide the certificate details when you deploy the new version, self-signed certificates are generated.

## Step 2: Log In to the New Version of Tanzu Kubernetes Grid Integrated Edition Management Console

When the OVA deployment has completed successfully, you can access the new version of the management console.

1. In the vSphere Client, right-click the new Management Console VM and select **Power** > **Power On**.
2. When the new Management Console VM has booted, go to the **Summary** tab for the VM and copy its IP address, if you do not know it already.
3. Enter the IP address of the new Management Console VM in a browser.
4. At the VMware Tanzu Kubernetes Grid Integrated Edition log in page, enter username `root` and the root password that you set when you deployed the new version of the OVA template.

## Step 3: Migrate the Configuration from the Old Appliance to the New Version

Tanzu Kubernetes Grid Integrated Edition Management Console provides an upgrade wizard to help

you to migrate the configuration of your old deployment to the new version.

To get help in the wizard at any time, click the ? icon at the top of the page, or click the **More Info...** links in each section to see help topics relevant to that section. Click the i icons for tips about how to fill in specific fields.

1. On the VMware Tanzu Kubernetes Grid Integrated Edition Management Console landing page for the new version, click **Upgrade**.

| RESOURCE                                 | VERSION        |
|------------------------------------------|----------------|
| Ops Manager                              | 2.10.5         |
| Tanzu Kubernetes Grid Integrated Edition | 110.0-build.22 |
| Harbor                                   | 2.1.2-build.5  |

Do you want to install a new TKGI instance or upgrade an existing one?

**INSTALL**      **UPGRADE**

[View a larger version of this image](#)

2. Enter the IP address of the old version of the Tanzu Kubernetes Grid Integrated Edition Management Console VM in the **Endpoint** text box.
3. Enter the username and password for the old version of the management console VM and click **Connect**.

This upgrade will migrate your current instance to this new release of the existing resources.

Provide new credentials to existing Tanzu Kubernetes Grid Integrated Edition Management Console

|                                          |                  |                   |
|------------------------------------------|------------------|-------------------|
| Endpoint<br>Existing Server 10.185.29.35 | Username<br>root | Password<br>..... |
|------------------------------------------|------------------|-------------------|

**CONNECT**

4. Under Resources, verify the list of components that will be upgraded and click **Next**.

| Resources       |                  |                |                          |
|-----------------|------------------|----------------|--------------------------|
| Resource        | Previous Version | This Version   | Description              |
| harbor          | 1.8.4-build.2    | 1.9.3-build.2  | harbor component upgrade |
| windowsStemcell |                  |                |                          |
| opsman          | 2.6.11           | 2.7.3          | opsman component upgrade |
| pks             | 1.5.1-build.8    | 1.6.0-build.17 | pks component upgrade    |

**NEXT**    **CANCEL**

- If any sections of the configuration wizard are marked in red, expand and reconfigure them.

Sections might appear in red because they are in an error state, or because they relate to new configuration parameters that were not present in the previous version. For information about how to configure each section, see [Deploy Tanzu Kubernetes Grid Integrated Edition by Using the Configuration Wizard](#). For information about new parameters that have been added, see the release notes for the version to which you are upgrading.

- When all of the sections of the configuration wizard are green, click **Generate Configuration** to see the generated YAML file.
- (Optional) Specify an FQDN address for the Ops Manager VM by editing the YAML directly in the YAML editor.



**WARNING:** You cannot change the Ops Manager FQDN of Tanzu Kubernetes Grid Integrated Edition once it has already deployed.

To specify an FQDN address for the Ops Manager VM, update the YAML as follows:

- Locate the `opsman_fqdn:` entry in the YAML file.
- Update the `opsman_fqdn:` entry with the Ops Manager VM FQDN:

```
opsman_fqdn: "myopsman.example.com"
```

- Make sure that the FQDN is mapped to the following IP address:
  - For NSX-T deployments map it to the first address in the floating IP range.
  - For vSphere without NSX-T deployments, map it to the first address in the deployment network, excluding the gateway, deployment DNS, and reserved IP range.

If you start the upgrade and you have not mapped the FQDN to an IP address, the deployment fails with an error. If this happens, configure the mapping as above, return to the YAML editor, and start the upgrade again.

- Optionally click **Export YAML** to save a copy of the YAML file for future use.  
This is recommended. The manifest is exported as the file `PksConfiguration.yaml`.
- Click **Apply Configuration** then **Continue** to upgrade Tanzu Kubernetes Grid Integrated Edition.
- On the VMware Tanzu Kubernetes Grid Integrated Edition Upgrade page, follow the progress of the upgrade.

## Next Steps

You can now access the upgraded Tanzu Kubernetes Grid Integrated Edition control plane and continue deploying Kubernetes clusters. Any new clusters that you deploy from the upgraded Tanzu Kubernetes Grid Integrated Edition control plane will use the new version of Kubernetes.



**Important:** Existing clusters are not upgraded during the upgrade of Tanzu Kubernetes Grid Integrated Edition Management Console. You must manually upgrade any existing clusters.

For information about how you can use Tanzu Kubernetes Grid Integrated Edition Management Console to monitor and manage your upgraded deployment, see [Monitor and Manage Tanzu Kubernetes Grid Integrated Edition in the Management Console](#).

You can decommission the old version of Tanzu Kubernetes Grid Integrated Edition by deleting the previous version of the Tanzu Kubernetes Grid Integrated Edition Management Console VM from the vSphere inventory.

## Upgrading Tanzu Kubernetes Grid Integrated Edition with Ops Manager

This section describes how to upgrade the VMware Tanzu Kubernetes Grid Integrated Edition tile. See the following topics:

- [Upgrade Preparation Checklist for Tanzu Kubernetes Grid Integrated Edition](#)
- [Upgrade Order for Tanzu Kubernetes Grid Integrated Edition Environments on vSphere](#)
- [Upgrading Tanzu Kubernetes Grid Integrated Edition \(Antrea and Flannel Networking\)](#)
- [Upgrading Tanzu Kubernetes Grid Integrated Edition \(NSX-T Networking\)](#)
- [Maintaining Workload Uptime](#)
- [Configuring the Upgrade Pipeline](#)

For information about upgrading the Tanzu Kubernetes Grid Integrated Edition Management Console, see [Upgrade Tanzu Kubernetes Grid Integrated Edition Management Console](#).

## Upgrade Preparation Checklist for Tanzu Kubernetes Grid Integrated Edition

This topic serves as a checklist for preparing to upgrade VMware Tanzu Kubernetes Grid Integrated Edition from v1.15 to v1.16.

## Overview

This topic describes the steps that you must follow before beginning your TKGI upgrade.



**Warning:** Failure to follow these instructions might jeopardize your existing deployment data and cause the TKGI upgrade to fail.

To prepare for a TKGI Upgrade:

1. [Back Up Your TKGI Deployment](#)
2. [Review What Happens During TKGI Upgrades](#)
3. [Review Changes in TKGI](#)
4. [Determine Upgrade Order \(vSphere Only\)](#)
5. [Set User Expectations and Restrict Cluster Access](#)
6. [Update Default Scanner](#)
7. If you are upgrading a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
8. [Upgrade All Clusters to v1.15](#)
9. [Migrate from PSP to PSA Controller](#)
10. [Verify Your Clusters Support Upgrading](#)
11. [Verify Health of Kubernetes Environment](#)
12. [Verify Your Environment Configuration](#)
13. [Clean Up or Fix Failed Kubernetes Clusters](#)
14. [Verify Kubernetes Clusters Have Unique External Hostnames](#)
15. [Verify TKGI Proxy Configuration](#)
16. [Check PodDisruptionBudget Value](#)
17. [\(Optional\) Configure Node Drain Behavior](#)

After completing the steps in this topic, continue to [Upgrading Tanzu Kubernetes Grid Integrated Edition \(Antrea and Flannel Networking\)](#) or [Upgrading Tanzu Kubernetes Grid Integrated Edition \(NSX-T Networking\)](#).

## Back Up Your Tanzu Kubernetes Grid Integrated Edition Deployment

VMware recommends backing up your Tanzu Kubernetes Grid Integrated Edition deployment and workloads before upgrading. To back up Tanzu Kubernetes Grid Integrated Edition, see [Backing Up and Restoring Tanzu Kubernetes Grid Integrated Edition](#).

## Review What Happens During Tanzu Kubernetes Grid Integrated Edition Upgrades

If you have not already done so, review [About Tanzu Kubernetes Grid Integrated Edition Upgrades](#).

Plan your upgrade based on your workload capacity and uptime requirements.

## Review Changes in Tanzu Kubernetes Grid Integrated Edition v1.16

Review the [Release Notes](#) for Tanzu Kubernetes Grid Integrated Edition v1.16.

## Determine Upgrade Order (vSphere Only)

To determine the upgrade order for your Tanzu Kubernetes Grid Integrated Edition environment, review [Upgrade Order for Tanzu Kubernetes Grid Integrated Edition Environments on vSphere](#).

## Set User Expectations and Restrict Cluster Access

Coordinate the Tanzu Kubernetes Grid Integrated Edition upgrade with cluster admins and users.

During the upgrade:

- Their workloads will remain active and accessible.
- They will be unable to perform cluster management functions, including creating, resizing, updating, and deleting clusters.
- They will be unable to log in to TKGI or use the TKGI CLI and other TKGI control plane services.



**Note:** Cluster admins should not start any cluster management tasks right before an upgrade. Wait for cluster operations to complete before upgrading.

## Upgrade All Clusters to v1.15

Tanzu Kubernetes Grid Integrated Edition v1.16 does not support clusters running versions of TKGI earlier than v1.15.

Before you upgrade from Tanzu Kubernetes Grid Integrated Edition v1.15 to v1.16, you must upgrade all of your TKGI-provisioned clusters to v1.15.

To upgrade TKGI-provisioned clusters:

1. Check the version of your clusters:

```
tkgi clusters
```

2. If one or more of your clusters are running a version of TKGI earlier than v1.15:

1. Verify these clusters support being upgraded to TKGI v1.15. For more information, see [Verify Your Clusters Support Upgrading](#) below.
2. Upgrade the clusters to TKGI v1.15. For instructions, see the [Upgrading Clusters](#) topic in the TKGI v1.15 documentation.

## Migrate from PSP to PSA Controller

Pod Security Policy (PSP) security configuration must be migrated to Pod Security Admission (PSA) before upgrading to TKGI v1.16. For more information, see [Migrate from PSP to PSA Controller](#) in *Pod Security Admission in TKGI*.



**Note:** Kubernetes v1.25 does not serve the `policy/v1beta1 PodSecurityPolicy` API.

You must replace `PodSecurityPolicy` configurations with PSA before upgrading to TKGI v1.16.

## Verify Your Clusters Support Upgrading

It is critical that you confirm that a cluster's resource usage is within the recommended maximum limits before upgrading the cluster.

VMware Tanzu Kubernetes Grid Integrated Edition upgrades a cluster by upgrading control plane and worker nodes individually. The upgrade processes a control plane node by redistributing the node's workload, stopping the node, upgrading it and restoring its workload. This redistribution of a node's workloads increases the resource usage on the remaining nodes during the upgrade process.

If a Kubernetes cluster control plane VM is operating too close to capacity, the upgrade can fail.



**Warning:** Downtime is required to repair a cluster failure resulting from upgrading an overloaded Kubernetes cluster control plane VM.

To prevent workload downtime during a cluster upgrade, complete the following before upgrading a cluster:

1. Ensure none of the control plane VMs being upgraded will become overloaded during the cluster upgrade. See [Control Plane Node VM Size](#) for more information.
2. Review the cluster's workload resource usage.
3. Scale up the cluster if it is near capacity on its existing infrastructure.
  1. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
  2. Scale up your cluster by running the command below or create a new cluster using a larger plan. For more information, see [Changing Cluster Configurations](#).

```
tkgi update-cluster CLUSTER-NAME --num-nodes NUMBER-OF-WORKER-NODES
```

Where:

- `CLUSTER-NAME` is the name of your cluster.
- `NUMBER-OF-WORKER-NODES` is the number of worker nodes that you want to set for the cluster.



**Note:** VMware recommends that you avoid using the `tkgi resize` command to perform resizing operations.

4. Run the cluster's workloads on at least three worker VMs using multiple replicas of your workloads spread across those VMs. For more information, see [Maintaining Workload Uptime](#).

## Verify Health of Kubernetes Environment

Verify that your Kubernetes environment is healthy. To verify the health of your Kubernetes environment, see [Verifying Deployment Health](#).

## Verify Your Environment Configuration

If you are upgrading Tanzu Kubernetes Grid Integrated Edition, verify the configuration of your environment supports the TKGI version you are installing:

- [Verify Your vSphere with NSX-T Configuration](#)
- [Verify Your Antrea Environment Configuration](#)

If you are using Flannel networking, this verification step is unnecessary.

### Verify Your vSphere with NSX-T Configuration

If you are upgrading Tanzu Kubernetes Grid Integrated Edition for environments using vSphere with NSX-T, perform the following steps:

1. Verify that the vSphere datastores have enough space.
2. Verify that the vSphere hosts have enough memory.
3. Verify that there are no alarms in vSphere.
4. Verify that the vSphere hosts are in a good state.
5. Verify that NSX Edge is configured for high availability.



**Note:** Workloads in your Kubernetes cluster are unavailable while the NSX Edge nodes run the upgrade unless you configure NSX Edge for high availability. For more information, see the [Configure NSX Edge for High Availability \(HA\)](#) section of *Preparing NSX-T Before Deploying Tanzu Kubernetes Grid Integrated Edition*.

### Verify Your Antrea Environment Configuration

If you are upgrading Tanzu Kubernetes Grid Integrated Edition in an environment using Antrea networking, perform the following steps:

1. Verify the 6081 UDP port is open on all worker node VMs.
2. Verify the 8091 TCP port is open on all control plane node VMs.
3. Verify your environment configuration meets the Antrea networking requirements. For more information, see [Network Requirements](#) in the Antrea GitHub repository.



**Note:** Port 6081 must be open on all of the worker node VMs and port 8091 must be open on all control plane node VMs in the clusters you create in an Antrea networking environment.

## Clean Up or Fix Failed Kubernetes Clusters

Clean up or fix any previous failed attempts to create TKGI clusters with the TKGI Command Line Interface (TKGI CLI) by performing the following steps:

1. View your deployed clusters by running the following command:

```
tkgi clusters
```

If the `Status` of any cluster displays as `FAILED`, continue to the next step. If no cluster displays as `FAILED`, no action is required. Continue to the next section.

2. To troubleshoot and fix failed clusters, perform the procedure in [Cluster Creation Fails](#).
3. To clean up failed BOSH deployments related to failed clusters, perform the procedure in [Cannot Re-Create a Cluster that Failed to Deploy](#).
4. After fixing and cleaning up any failed clusters, view your deployed clusters again by running `tkgi clusters`.

For more information about troubleshooting and fixing failed clusters, see the [Knowledge Base](#).

## Verify Kubernetes Clusters Have Unique External Hostnames

Verify that existing Kubernetes clusters have unique external hostnames by checking for multiple Kubernetes clusters with the same external hostname. Perform the following steps:

1. Log in to the TKGI CLI. For more information, see [Logging in to Tanzu Kubernetes Grid Integrated Edition](#). You must log in with an account that has the UAA scope of `pks.clusters.admin`. For more information about UAA scopes, see [Managing Tanzu Kubernetes Grid Integrated Edition Users with UAA](#).
2. View your deployed TKGI clusters by running the following command:

```
tkgi clusters
```

3. For each deployed cluster, run `tkgi cluster CLUSTER-NAME` to view the details of the cluster. For example:

```
$ tkgi cluster my-cluster
```

Examine the output to verify that the `Kubernetes Master Host` is unique for each cluster.

## Verify TKGI Proxy Configuration

Verify your current TKGI proxy configuration by performing the following steps:

1. Check whether an existing proxy is enabled:
  1. Log in to Ops Manager.
  2. Click the **VMware Tanzu Kubernetes Grid Integrated Edition** tile.
  3. Click **Networking**.
  4. If **HTTP/HTTPS Proxy is Disabled**, no action is required. Continue to the next

section. If **HTTP/HTTPS Proxy** is **Enabled**, continue to the next step.

2. If the existing **No Proxy** field contains any of the following values, or you plan to add any of the following values, contact Support:

- `localhost`
- Hostnames containing dashes, such as `my-host.mydomain.com`

## Check PodDisruptionBudget Value

Tanzu Kubernetes Grid Integrated Edition upgrades can run without ever completing if any Kubernetes app has a `PodDisruptionBudget` with `maxUnavailable` set to `0`.

To ensure that no apps have a `PodDisruptionBudget` with `maxUnavailable` set to `0`:

1. Run the following `kubectl` command to verify the `PodDisruptionBudget` as the cluster administrator:

```
kubectl get poddisruptionbudgets --all-namespaces
```

2. Examine the output to verify that no app displays `0` in the `MAX UNAVAILABLE` column.

## (Optional) Configure Node Drain Behavior

During the Tanzu Kubernetes Grid Integrated Edition upgrade process, worker nodes are cordoned and drained. Workloads can prevent worker nodes from draining and cause the upgrade to fail or hang.

To prevent hanging cluster upgrades, you can configure default node drain behavior using the following methods:

- [Configure with the TKGI Tile](#)
- [Configure with the TKGI CLI](#)

The new default behavior takes effect during the next upgrade, not immediately after configuring the behavior.

### Configure with the TKGI Tile

To configure node drain behavior in the Tanzu Kubernetes Grid Integrated Edition tile, see [Worker Node Hangs Indefinitely](#) in *Troubleshooting*.

### Configure with the TKGI CLI

To configure default node drain behavior with the TKGI CLI:

1. View the current node drain behavior by running the following command:

```
tkgi cluster CLUSTER-NAME --details
```

Where `CLUSTER-NAME` is the name of your cluster.

For example:

```
$ tkgi cluster my-cluster --details

Name: my-cluster
Plan Name: small
UUID: f55ed6c4-c0a7-451d-b735-56c89fdb2ad7
Last Action: CREATE
Last Action State: succeeded
Last Action Description: Instance provisioning completed
Kubernetes Master Host: my-cluster.tkgi.local
Kubernetes Master Port: 8443
Worker Nodes: 3
Kubernetes Master IP(s): 10.196.219.88
Network Profile Name:
Kubernetes Settings Details:
 Set by Cluster:
 Kubelet Node Drain timeout (mins) (kubelet-drain-timeout):
 10
 Kubelet Node Drain grace-period (mins) (kubelet-drain-grace-period):
 10
 Kubelet Node Drain force (kubelet-drain-force):
 true
 Set by Plan:
 Kubelet Node Drain force-node (kubelet-drain-force-node):
 true
 Kubelet Node Drain ignore-daemonsets (kubelet-drain-ignore-daemonse
ts): true
 Kubelet Node Drain delete-local-data (kubelet-drain-delete-local-da
ta): true
```

2. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
3. Configure the default node drain behavior by running the following command:

```
tkgi update-cluster CLUSTER-NAME FLAG
```

Where:

- ◊ **CLUSTER-NAME** is the name of your cluster.
- ◊ **FLAG** is an action flag for updating the node drain behavior.

For example:

```
$ tkgi update-cluster my-cluster -kubelet-drain-timeout 1 -kubelet-dra
in-grace-period 5
Update summary for cluster my-cluster:

Kubelet Drain Timeout: 1

Kubelet Drain Grace Period: 5

Are you sure you want to continue? (y/n): y

Use 'tkgi cluster my-cluster' to monitor the state of your cluster
```

For a list of the available action flags for setting node drain behavior, see [tkgi update-cluster](#)

in *TKGI CLI*.

## Upgrade Order for Tanzu Kubernetes Grid Integrated Edition Environments on vSphere

This topic provides upgrade scenarios for Tanzu Kubernetes Grid Integrated Edition (TKGI) environments that are upgraded from v1.15 to v1.16 on vSphere.

### Overview

When you upgrade TKGI on vSphere, you might also upgrade vSphere and, if you are using it, NSX-T.

TKGI, NSX-T, and vSphere upgrades depend on each other. Some combinations also require upgrading Ops Manager or TKGI-provisioned Kubernetes clusters.

For any combination of upgrades that you perform, you must follow the upgrade order described in this topic.

- If your environment is on vSphere with NSX-T networking, see [TKGI on vSphere with NSX-T Networking](#) below.
- If your environment is on vSphere with Antrea or Flannel networking, see [TKGI on vSphere \(Antrea and Flannel Networking\)](#) below.

### TKGI on vSphere with NSX-T Networking

When upgrading a TKGI environment on vSphere with NSX-T networking, you can choose to upgrade any of the following:

- TKGI only, optionally including Kubernetes clusters
- TKGI, Kubernetes clusters, and NSX-T
- TKGI, Kubernetes clusters, NSX-T, and vSphere

For more information, see below:

| To upgrade these components... | Use this order...                                                                                                                                                                    | For more information, see...                           |
|--------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------|
| TKGI                           | <ol style="list-style-type: none"> <li>1. Upgrade Ops Manager if necessary.</li> <li>2. Upgrade TKGI.</li> <li>3. (Recommended) Upgrade Kubernetes clusters.</li> </ol>              | <a href="#">Upgrading to TKGI v1.16</a>                |
| TKGI and NSX-T                 | <ol style="list-style-type: none"> <li>1. Upgrade NSX-T.</li> <li>2. Upgrade Ops Manager if necessary.</li> <li>3. Upgrade TKGI.</li> <li>4. Upgrade Kubernetes clusters.</li> </ol> | <a href="#">Upgrading to TKGI v1.16 and NSX-T v3.2</a> |



**Warning:** Refer to the [Release Notes](#) for current version support, known issues, and other important information.

For a list of NSX-T and vSphere versions compatible with Tanzu Kubernetes Grid Integrated Edition v1.16, see:

- [Product Snapshot](#) in *Release Notes*
- [VMware Product Interoperability Matrices](#)

## Scenario 1: Upgrading to TKGI v1.16

In this upgrade scenario, you upgrade Tanzu Kubernetes Grid Integrated Edition from v1.15 to v1.16 and do not upgrade your NSX-T or vSphere infrastructure.

The upgrade scenario includes the following steps:

1. Upgrade Ops Manager to v2.10.53 or later. These are the recommended Ops Manager versions for Tanzu Kubernetes Grid Integrated Edition v1.16.0. To verify Ops Manager compatibility with other v1.16 versions, see [VMware Tanzu Network](#).
2. Upgrade Tanzu Kubernetes Grid Integrated Edition from v1.15 to v1.16.
3. If you are upgrading a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
4. (Recommended) Upgrade all Kubernetes clusters to Tanzu Kubernetes Grid Integrated Edition v1.16. This upgrades the NCP version of your clusters.

See the table below for version information and instructions for this upgrade scenario:

| Component   | Pre-upgrade version | Post-upgrade version | Instructions                                                                                |
|-------------|---------------------|----------------------|---------------------------------------------------------------------------------------------|
| TKGI        | v1.15               | v1.16                | See <a href="#">Upgrading Tanzu Kubernetes Grid Integrated Edition (NSX-T Networking)</a> . |
| Ops Manager | v2.10.46            | v2.10.53 or v3.0.4   | n/a                                                                                         |

## Scenario 2: Upgrading to TKGI v1.16 and NSX-T v3.2



**Warning:** Refer to the [Release Notes](#) for current version support, known issues, and other important information.

In this upgrade scenario, you upgrade Tanzu Kubernetes Grid Integrated Edition from v1.15 to v1.16 and NSX-T from v3.2.1, to v3.2.2 or later.

The upgrade scenario includes the following steps:

1. Upgrade NSX-T from v3.2.1 or later to v3.2.2 or later.
2. Upgrade Ops Manager to v2.10.53 or later. These are the recommended Ops Manager versions for Tanzu Kubernetes Grid Integrated Edition v1.16.0. To verify Ops Manager compatibility with other v1.16 versions, see [VMware Tanzu Network](#).
3. Upgrade Tanzu Kubernetes Grid Integrated Edition from v1.15 to v1.16.

4. If you are upgrading a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
5. Upgrade all Kubernetes clusters to Tanzu Kubernetes Grid Integrated Edition v1.16. This upgrades the NCP version of your clusters.

See the table below for version information and instructions for this upgrade scenario:

| Component   | Pre-upgrade version | Post-upgrade version | Instructions                                                                               |
|-------------|---------------------|----------------------|--------------------------------------------------------------------------------------------|
| TKGI        | v1.15               | v1.16                | <a href="#">See Upgrading Tanzu Kubernetes Grid Integrated Edition (NSX-T Networking).</a> |
| Ops Manager | v2.10.46            | v2.10.53 or v3.0.4   | n/a                                                                                        |
| NSX-T       | v3.2.1 or later     | v3.2.2 or later      |                                                                                            |
| NCP         | v4.0.0.0            | v4.1.0               | n/a                                                                                        |

## TKGI on vSphere (Antrea and Flannel Networking)

When upgrading a Tanzu Kubernetes Grid Integrated Edition environment on vSphere with Antrea or Flannel networking, you can choose to upgrade any of the following:

- TKGI only, optionally including Kubernetes clusters
- TKGI, Kubernetes clusters, and vSphere

For more information, see below:

| To upgrade these components... | Use this order...                                                                                                                                                       | For more information, see...            |
|--------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------|
| TKGI                           | <ol style="list-style-type: none"> <li>1. Upgrade Ops Manager if necessary.</li> <li>2. Upgrade TKGI.</li> <li>3. (Recommended) Upgrade Kubernetes clusters.</li> </ol> | <a href="#">Upgrading to TKGI v1.16</a> |

For a list of vSphere versions compatible with Tanzu Kubernetes Grid Integrated Edition v1.16, see [VMware Product Interoperability Matrices](#).

## Scenario 1: Upgrading to TKGI v1.16

In this upgrade scenario, you upgrade Tanzu Kubernetes Grid Integrated Edition from v1.15 to v1.16 and do not upgrade your vSphere infrastructure.

The upgrade scenario includes the following steps:

1. Upgrade Ops Manager to v2.10.53 or later. These are the recommended Ops Manager versions for Tanzu Kubernetes Grid Integrated Edition v1.16.0. To verify Ops Manager compatibility with other v1.16 versions, see [VMware Tanzu Network](#).
2. Upgrade Tanzu Kubernetes Grid Integrated Edition from v1.15 to v1.16.
3. If you are upgrading a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.

- (Recommended) Upgrade all Kubernetes clusters to Tanzu Kubernetes Grid Integrated Edition v1.16. This upgrades the NCP version of your clusters.

See the table below for version information and instructions for this upgrade scenario:

| Component   | Pre-upgrade version | Post-upgrade version | Instructions                                                                                             |
|-------------|---------------------|----------------------|----------------------------------------------------------------------------------------------------------|
| TKGI        | v1.15               | v1.16                | See <a href="#">Upgrading Tanzu Kubernetes Grid Integrated Edition (Antrea and Flannel Networking)</a> . |
| Ops Manager | v2.10.46            | v2.10.53 or v3.0.4   | n/a                                                                                                      |
| vSphere     | v7.0                | v7.0                 | n/a                                                                                                      |

## Upgrading Tanzu Kubernetes Grid Integrated Edition (Antrea and Flannel Networking)

This topic explains how to upgrade VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) in Antrea and Flannel Networking environments from v1.15 to v1.16 on vSphere, Google Cloud Platform (GCP), Amazon Web Services (AWS), and Azure.

For instructions on upgrading Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T networking, see [Upgrading Tanzu Kubernetes Grid Integrated Edition \(NSX-T Networking\)](#).



**Warning:** Do not manually upgrade your Kubernetes version. Tanzu Kubernetes Grid Integrated Edition includes the compatible Kubernetes version.

## Overview

Before you upgrade, follow the procedures in [Prepare to Upgrade](#) below to plan and prepare your upgrade.

After you complete the preparation steps, continue to the procedures in [Perform the Upgrade](#) below. These steps guide you through the process of upgrading Ops Manager and the Tanzu Kubernetes Grid Integrated Edition tile, importing a new stemcell, and applying the changes to your deployment.

After you complete the upgrade, follow the procedures in [After the Upgrade](#) below to verify that your upgraded Tanzu Kubernetes Grid Integrated Edition deployment is running properly.

## Prepare to Upgrade

If you have not already, complete all of the steps in [Upgrade Preparation Checklist for Tanzu Kubernetes Grid Integrated Edition](#).



**Note:** Kubernetes v1.25 does not serve the `policy/v1beta1 PodSecurityPolicy` API. You must replace `PodSecurityPolicy` configurations with PSA before upgrading to TKGI v1.16.

## Perform the Upgrade

This section describes the steps required to upgrade to Tanzu Kubernetes Grid Integrated Edition v1.16:

1. [Upgrade Ops Manager](#)
2. [Download and Import Tanzu Kubernetes Grid Integrated Edition v1.16](#)
3. [Download and Import Stemcells](#)
4. [Modify Plan CNI Configuration](#)
5. [Verify Errand Configuration](#)
6. [Verify Other Configurations](#)
7. [Apply Changes to the Tanzu Kubernetes Grid Integrated Edition Tile](#)

### Upgrade Ops Manager

Each version of Tanzu Kubernetes Grid Integrated Edition is compatible with multiple versions of Ops Manager.



**Warning:** If you use an automated pipeline to upgrade TKGI, see [Configure Automated Ops Manager and Ubuntu Jammy Stemcell Downloading](#) in *Configuring the Upgrade Pipeline*.

To determine Ops Manager compatibility and, if necessary, upgrade Ops Manager:

1. See [VMware Tanzu Network](#) to determine if your Ops Manager version is compatible with Tanzu Kubernetes Grid Integrated Edition v1.16.
2. If your Ops Manager version is not compatible with Tanzu Kubernetes Grid Integrated Edition v1.16, follow the steps below.
3. Upgrade Ops Manager. For instructions, see [Import Installation to Ops Manager v2.10 VM](#) or [Import Installation to Ops Manager v3.0 VM](#) in *Upgrading Ops Manager* in the Ops Manager documentation.
4. Verify that the Tanzu Kubernetes Grid Integrated Edition control plane remains functional by performing the following steps:
  1. Add more workloads and create an additional cluster. For more information, see [About Cluster Upgrades](#) in *Maintaining Workload Uptime* and [Creating Clusters](#).
  2. Monitor the Tanzu Kubernetes Grid Integrated Edition control plane in the **Tanzu Kubernetes Grid Integrated Edition** tile > **Status** tab. Review the load and resource usage data for the TKGI API and TKGI Database VMs. If any levels are at capacity, scale up the VMs.

### Download and Import Tanzu Kubernetes Grid Integrated Edition v1.16

When you upgrade Tanzu Kubernetes Grid Integrated Edition, your configuration settings typically

migrate to the new version automatically. To download and import a Tanzu Kubernetes Grid Integrated Edition version:

1. Download the desired version of the product from [VMware Tanzu Network](#).
2. Navigate to the Ops Manager Installation Dashboard and click **Import a Product** to upload the product file.
3. Under the **Import a Product** button, click + next to **Tanzu Kubernetes Grid Integrated Edition**. This adds the tile to your staging area.

## Download and Import Stemcells

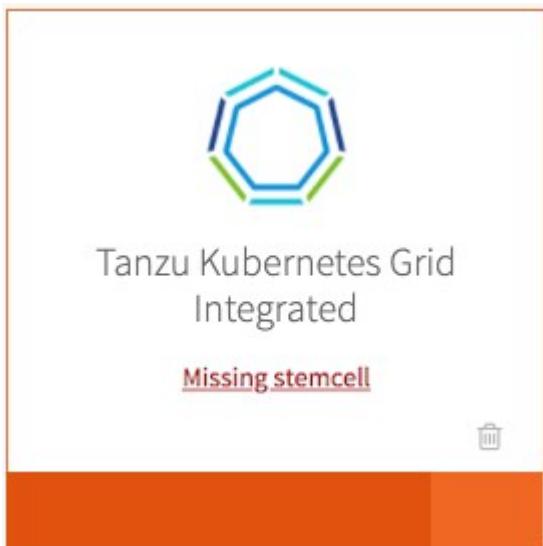
TKGI requires a Ubuntu Jammy stemcell. A stemcell for Windows 2019 is also required if you intend to create Windows worker-based clusters. For information about Windows stemcells, see [Configuring Windows Worker-Based Clusters](#).



**Warning:** If you use an automated pipeline to upgrade TKGI, see [Configure Automated Ops Manager and Ubuntu Jammy Stemcell Downloading](#) in *Configuring the Upgrade Pipeline*.

If Ops Manager does not have the Ubuntu Jammy stemcell required for Tanzu Kubernetes Grid Integrated Edition v1.16, the Tanzu Kubernetes Grid Integrated Edition tile displays the message **Missing stemcell**. To download and import a new Ubuntu Jammy stemcell, follow the steps below:

1. On the Tanzu Kubernetes Grid Integrated Edition tile, click the **Missing stemcell** link.



2. In the **Stemcell Library**, locate the **Tanzu Kubernetes Grid Integrated Edition** tile and note the required stemcell version.
3. Navigate to the [Stemcells \(Ubuntu Jammy\)](#) page on VMware Tanzu Network and download the required stemcell version for your IaaS.
4. Return to the **Installation Dashboard** in Ops Manager and click **Stemcell Library**.
5. On the **Stemcell Library** page, click **Import Stemcell** and select the stemcell file you downloaded from VMware Tanzu Network.
6. Select the Tanzu Kubernetes Grid Integrated Edition tile and click **Apply Stemcell to**

### Products.

7. Verify that Ops Manager successfully applied the stemcell. The stemcell version you imported and applied appears in the **Staged** column for Tanzu Kubernetes Grid Integrated Edition.
8. Return to the **Installation Dashboard**.

## Modify Container Network Interface Configuration

Tanzu Kubernetes Grid Integrated Edition supports using the Antrea Container Network Interface (CNI) as the CNI for new TKGI-provisioned clusters.

To configure Tanzu Kubernetes Grid Integrated Edition to use Antrea as the CNI for new clusters:

1. In the **Installation Dashboard**, click **Networking**.
2. Under **Container Networking Interface**, select **Antrea**.
3. Confirm the remaining Container Networking Interface settings.
4. Click **Save**.

For more information about Tanzu Kubernetes Grid Integrated Edition support for Antrea and Flannel CNIs, see [About Switching from the Flannel CNI to the Antrea CNI](#) in *About Tanzu Kubernetes Grid Integrated Edition Upgrades*.

## Verify Errand Configuration

To verify your **Errands** pane is correctly configured, do the following:

1. In the **Tanzu Kubernetes Grid Integrated Edition** tile, click **Errands**.
2. Under **Post-Deploy Errands**:
  - ◊ Review the **Upgrade all clusters** errand:
    - If you want to upgrade the Tanzu Kubernetes Grid Integrated Edition tile and all your existing Kubernetes clusters simultaneously, confirm that **Upgrade all clusters errand** is set to **Default (On)**. The errand upgrades all clusters. Upgrading Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes clusters can temporarily interrupt the service as described in [Service Interruptions](#).
    - If you want to upgrade the Tanzu Kubernetes Grid Integrated Edition tile only and then upgrade your existing Kubernetes clusters separately, deactivate **Upgrade all clusters errand**. For more information, see [Upgrading Clusters](#).



**Warning:** Deactivating the **Upgrade all clusters errand** causes the TKGI version tagged in your Kubernetes clusters to fall behind the Tanzu Kubernetes Grid Integrated Edition tile version. If you deactivate the **Upgrade all clusters errand** when upgrading the Tanzu Kubernetes Grid Integrated Edition tile, you must upgrade all your Kubernetes clusters before the next Tanzu Kubernetes Grid Integrated Edition upgrade.

- ◊ Configure the **Run smoke tests** errand:
  - Set the **Run smoke tests** errand to **On**. The errand uses the Tanzu Kubernetes Grid Integrated Edition Command Line Interface (TKGI CLI) to create a Kubernetes cluster and then delete it. If the creation or deletion fails, the errand fails and the installation of the Tanzu Kubernetes Grid Integrated Edition tile is aborted.

3. Click **Save**.

## Verify Other Configurations

To confirm your other **Tanzu Kubernetes Grid Integrated Edition** tile panes are correctly configured, do the following:

1. Review the **Assign AZs and Networks** pane.



**Note:** When you upgrade Tanzu Kubernetes Grid Integrated Edition, you must place singleton jobs in the AZ you selected when you first installed the Tanzu Kubernetes Grid Integrated Edition tile. You cannot move singleton jobs to another AZ.

2. Review the other configuration panes.
3. Make changes where necessary.



**WARNING:** Do not change the number of control plane/etcdb nodes for any plan that was used to create currently-running clusters. Tanzu Kubernetes Grid Integrated Edition does not support changing the number of control plane/etcdb nodes for plans with existing clusters.

4. Click **Save** on any panes where you make changes.

## Apply Changes to the Tanzu Kubernetes Grid Integrated Edition Tile

To complete the upgrade of the Tanzu Kubernetes Grid Integrated Edition tile:

1. Return to the **Installation Dashboard** in Ops Manager.
2. Click **Review Pending Changes**. For more information about this Ops Manager page, see [Reviewing Pending Product Changes](#).
3. Click **Apply Changes**.
4. (Optional) To monitor the progress of the **Upgrade all clusters errand** using the BOSH CLI, do the following:
  1. Log in to the BOSH Director by running `bosh -e MY-ENVIRONMENT log-in` from a VM that can access your Tanzu Kubernetes Grid Integrated Edition deployment. For more information, see [Using BOSH Diagnostic Commands in Tanzu Kubernetes Grid Integrated Edition](#).
  2. Run `bosh -e MY-ENVIRONMENT tasks`.

3. Locate the task number for the errand in the # column of the BOSH output.
4. Run `bosh task TASK-NUMBER`, replacing `TASK-NUMBER` with the task number you located in the previous step.

## After the Upgrade

After you complete the upgrade to Tanzu Kubernetes Grid Integrated Edition v1.16, complete the following verifications and upgrades:

- [Upgrade the TKGI and Kubernetes CLIs](#)
- [Verify the Upgrade](#)

### Upgrade the TKGI and Kubernetes CLIs

Upgrade the TKGI and Kubernetes CLIs on any local machine where you run commands that interact with your upgraded version of Tanzu Kubernetes Grid Integrated Edition.

To upgrade the CLIs, download and re-install the TKGI and Kubernetes CLI distributions that are provided with Tanzu Kubernetes Grid Integrated Edition on VMware Tanzu Network.

For more information about installing the CLIs, see the following topics:

- [Installing the TKGI CLI](#)
- [Installing the Kubernetes CLI](#)

### Verify the Upgrade

After you apply changes to the Tanzu Kubernetes Grid Integrated Edition tile and the upgrade is complete, do the following:

1. Verify that your Kubernetes environment is healthy. To verify the health of your Kubernetes environment, see [Verifying Deployment Health](#).  
For any cluster upgrade that fails, you can use the BOSH ID of the upgrade task for debugging. To retrieve the BOSH task ID, see [Retrieve Cluster Upgrade Task ID](#) in [Verifying Deployment Health](#).
2. Verify that the Tanzu Kubernetes Grid Integrated Edition control plane remains functional by performing the following steps:
  1. Add more workloads and create an additional cluster. For more information, see [About Cluster Upgrades](#) in [Maintaining Workload Uptime](#) and [Creating Clusters](#).
  2. Monitor the Tanzu Kubernetes Grid Integrated Edition control plane in the **Tanzu Kubernetes Grid Integrated Edition** tile > **Status** tab. Review the load and resource usage data for the TKGI API and TKGI Database VMs. If any levels are at capacity, scale up the VMs.

## Upgrading Tanzu Kubernetes Grid Integrated Edition (VMware NSX Networking)

This topic explains how to upgrade Tanzu Kubernetes Grid Integrated Edition (TKGI) from v1.15 to v1.16 on vSphere with NSX-T networking.

For instructions on upgrading TKGI with Flannel networking, see [Upgrading Tanzu Kubernetes Grid Integrated Edition \(Antrea and Flannel Networking\)](#).



**Warning:** Do not manually upgrade your Kubernetes version. TKGI includes the compatible Kubernetes version.

## Overview

Before you upgrade, follow the procedures in [Prepare to Upgrade](#) below to plan and prepare your upgrade.

After you complete the preparation steps, continue to the procedures in [Perform the Upgrade](#) below. These steps guide you through the process of upgrading Ops Manager and the TKGI tile, importing a new stemcell, and applying the changes to your deployment.

After you complete the upgrade, follow the procedures in [After the Upgrade](#) below to verify that your upgraded TKGI deployment is running properly and to optionally upgrade NSX-T and vSphere.

## Prerequisites

To see a list of NSX-T v3.2 versions compatible with TKGI v1.16, consult [Product Snapshot](#) in *Release Notes* for TKGI v1.16.

## Prepare to Upgrade

To prepare for upgrading Tanzu Kubernetes Grid Integrated Edition from TKGI v1.15 to TKGI v1.16:

- Complete all of the steps in [Upgrade Preparation Checklist for Tanzu Kubernetes Grid Integrated Edition](#).
- Review the upgrade procedures in [Upgrade Order for Tanzu Kubernetes Grid Integrated Edition Environments on vSphere](#).

## Perform the Upgrade

This section describes the steps required to upgrade to TKGI v1.16:

1. [Upgrade NSX-T Data Center to v3.2.2](#)
2. [Upgrade Ops Manager](#)
3. [Download and Import TKGI v1.16](#)
4. [Download and Import Stemcells](#)
5. [Upgrade the TKGI Tile](#)

### Upgrade NSX-T Data Center to v3.2.2

Tanzu Kubernetes Grid Integrated Edition v1.16 supports running on NSX-T v3.2.2 or later.

To upgrade NSX-T to NSX-T v3.2.2 or later:

1. Confirm that you are upgrading NSX-T to a version compatible with TKGI v1.16. For a list of

NSX-T v3.2 versions compatible with TKGI v1.16, see [Product Snapshot](#) in *Release Notes* for TKGI v1.16.



**Warning:** Refer to the [Release Notes](#) for current version support, known issues, and other important information.

2. Confirm that your vSphere v7.0 installation is on the supported version and patch for NSX-T v3.2.
  1. Refer to the [VMware Product Interoperability Matrices](#).
  2. If necessary, upgrade to the required vSphere version or patch before proceeding with the upgrade of NSX-T.
3. Upload the NSX-T upgrade bundle using the NSX-T Manager and proceed with the upgrade process by following the instructions in the UI.

| Category         | Status              | From Version           | Target Version     | Pre Check Status    |
|------------------|---------------------|------------------------|--------------------|---------------------|
| Edges            | Upgrade Not Started | 2.5.1.0.0.15314297 (8) | 3.0.0.0.0.15946012 | No checks performed |
| Hosts            | Upgrade Not Started | 2.5.1.0.0.15314289 (4) | 3.0.0.0.0.15945993 | No checks performed |
| Management Nodes | Upgrade Not Started | 2.5.1.0.0.15314292 (3) | 3.0.0.0.0.15946739 | No checks performed |

**Upload Upgrade Bundle**

Upload Local MUB file  
 Upload from remote location  
 Enter URL: ! This field is required for Upgrade Bundle.

**Do not power-off or reboot the nodes when upgrade is in progress. They may be rebooted automatically as part of the upgrade process.**

[Activate Window](#)  
[Get to Settings](#) to [NEXT](#)

For more information, refer to the [Upgrading NSX-T Data Center](#) documentation.

4. If you made architectural changes to your NSX-T environment that affect TKGI, such as adding or changing a [VIP address](#), or a [load balancer](#) for the NSX-T Management Cluster, update the BOSH Director and TKGI tiles with the new or updated IP addresses:
  1. In the BOSH Director tile > **vCenter Configuration** pane, update **NSX Address** and **NSX CA Cert**.
  2. In the TKGI tile > **Networking** pane, update **NSX Manager hostname** and **NSX Manager CA Cert**.
  3. After making changes to the BOSH Director or TKGI tiles:
    1. On the **Installation Dashboard** in Ops Manager, click **Review Pending Changes**.
    2. Expand the **Errands** list for TKGI.

3. Ensure that the **Upgrade all clusters errand** is selected.
4. Click **Apply Changes**.

## Upgrade Ops Manager

Each version of TKGI is compatible with multiple versions of Ops Manager. See [VMware Tanzu Network](#) to determine if your Ops Manager version is compatible with TKGI v1.16.



**Warning:** If you use an automated pipeline to upgrade TKGI, see [Configure Automated Ops Manager and Ubuntu Jammy Stemcell Downloading](#) in *Configuring the Upgrade Pipeline*.

To upgrade Ops Manager:

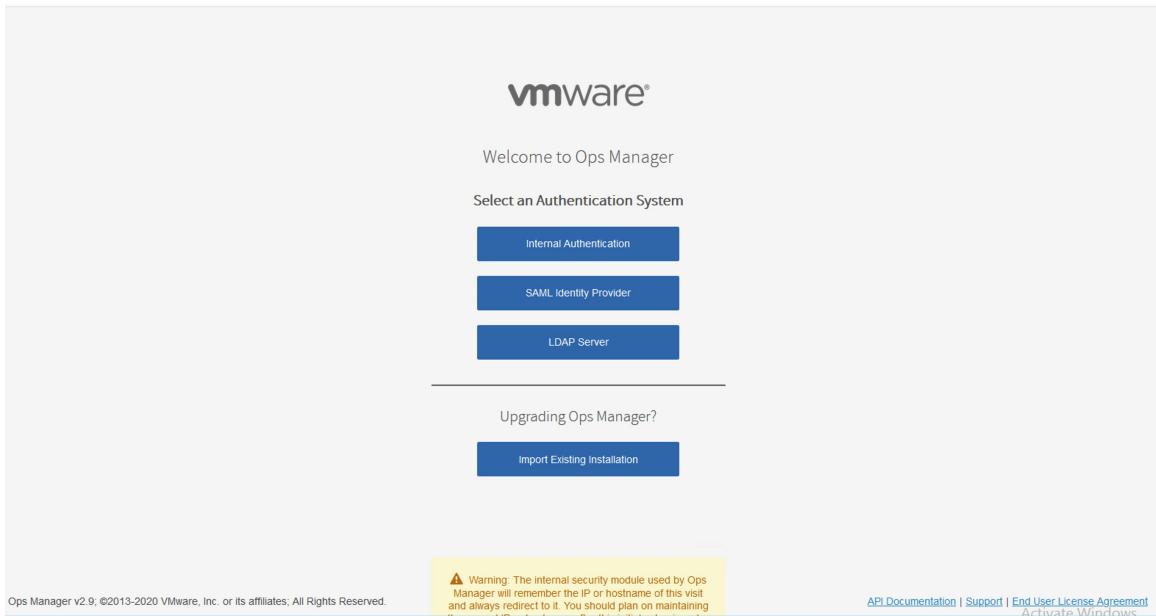
1. Log in to Ops Manager.
2. Click your username in the top right corner and navigate to **Settings > Export Installation Settings**.
3. Click **Export Installation Settings**.
  - ◊ Ops Manager exports an encrypted archive of your current installation configuration.
  - ◊ Later, you import this configuration into to your upgraded Ops Manager.

The screenshot shows the 'Settings' page with the 'Export Installation Settings' section highlighted. The 'EXPORT INSTALLATION SETTINGS' button is visible. A warning message states: 'Upgrading Ops Manager can block certain product upgrades. Verify an upgrade path exists for your installed products by reading [Upgrading Ops Manager](#).'. Below the main section, there is a list of other settings: Change Decryption Passphrase, Internal Authentication Settings, SAML Settings, LDAP Settings, SSL Certificate, VMware Tanzu Network Settings, Proxy Settings, Custom Banner, Export Installation Settings (which is the current page), Syslog, and Advanced Options.

4. Log in to vCenter Server using the vSphere Client.
5. Shut down the Ops Manager VM.
6. Deploy the upgraded Ops Manager VM by following the first two steps of [Deploying Ops Manager with NSX-T for TKGI](#):
  1. Step 1: Generate SSH Key Pair

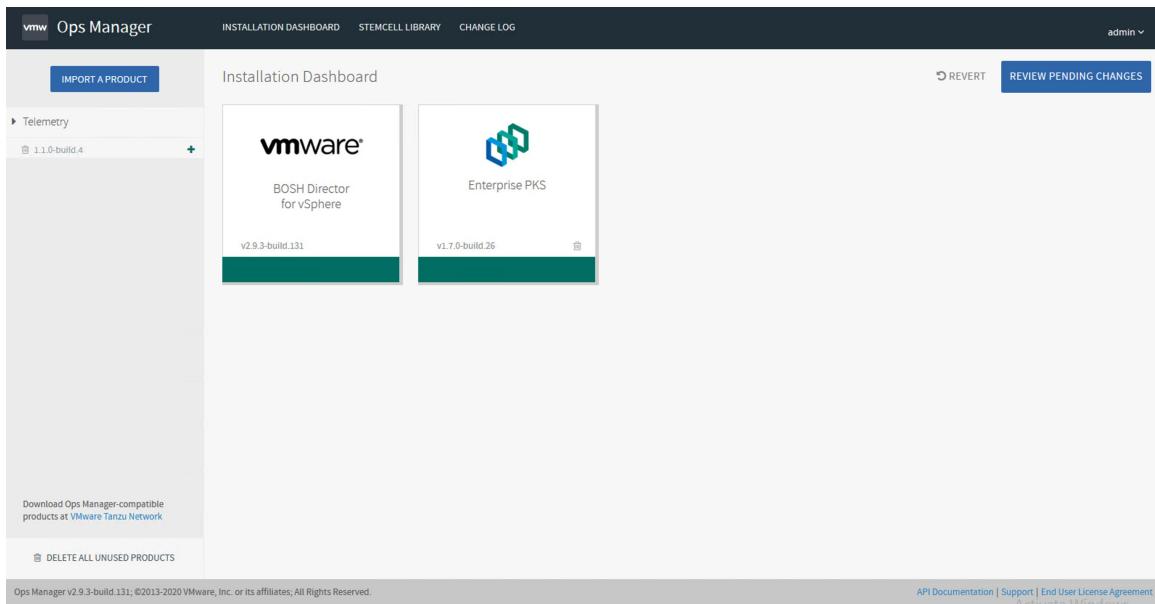
## 2. Step 2: Deploy Ops Manager for Tanzu Kubernetes Grid Integrated Edition

7. Using a browser, navigate to the newly-deployed Ops Manager web interface.
8. On the welcome page, select **Import Existing Installation**.



9. Browse to and select the installation configuration archive you exported.
10. Log in to Ops Manager
11. Click **Apply Changes**.

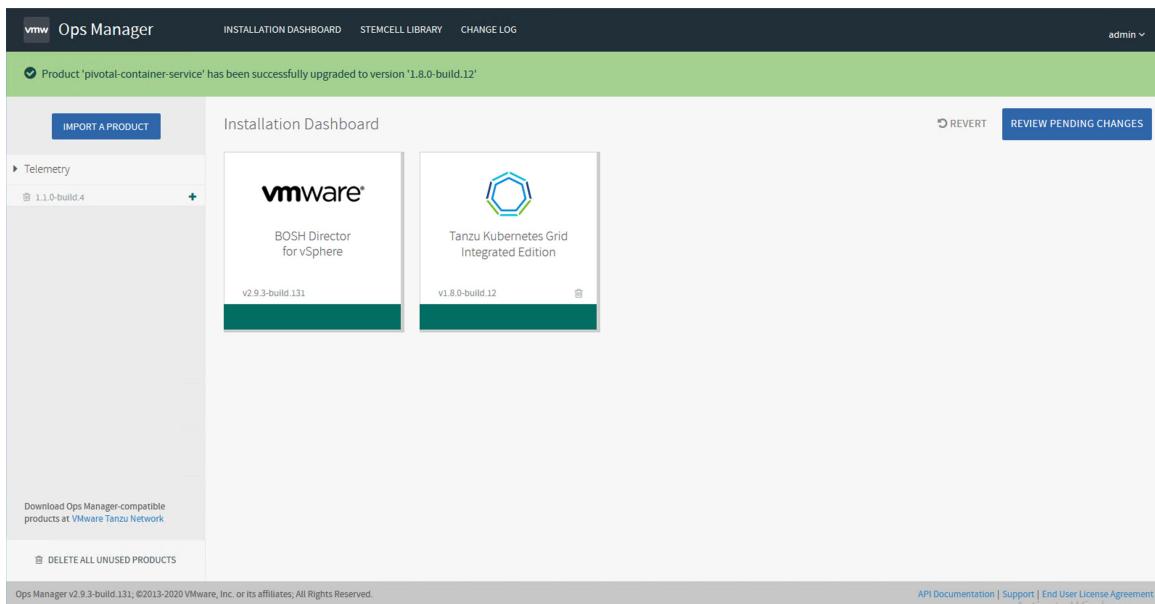
12. Verify that the BOSH Director for vSphere tile shows the upgrade version.



## Download and Import TKGI v1.16

When you upgrade TKGI, your configuration settings typically migrate to the new version automatically. To download and import a TKGI version:

1. Download the target version of the product from [VMware Tanzu Network](#).
2. Import the target version of the TKGI tile to the Ops Manager Installation Dashboard.



3. Click **Review Pending Changes**.
4. Expand the **Errands** dropdown and activate or deactivate **Upgrade all clusters errand**
  - See [Deciding Between Full and Two-Phase Upgrade](#) to decide whether to upgrade TKGI-provisioned Kubernetes clusters along with TKGI, or upgrade them later.
  - VMware recommends that you upgrade Kubernetes clusters along with TKGI if possible.
  - Activate the **Upgrade all clusters errand** to upgrade clusters along with TKGI.



**Warning:** Deactivating the **Upgrade all clusters errand** causes the TKGI version tagged in your Kubernetes clusters to fall behind the TKGI tile version. If you deactivate the **Upgrade all clusters errand** when upgrading the TKGI tile, you must upgrade all your Kubernetes clusters before the next TKGI upgrade.

- Set the **Run smoke tests** errand to **On**. The errand uses the TKGI CLI to create a Kubernetes cluster and then delete it. If the creation or deletion fails, the errand fails and the installation of the TKGI tile is aborted.

## Download and Import Stemcells

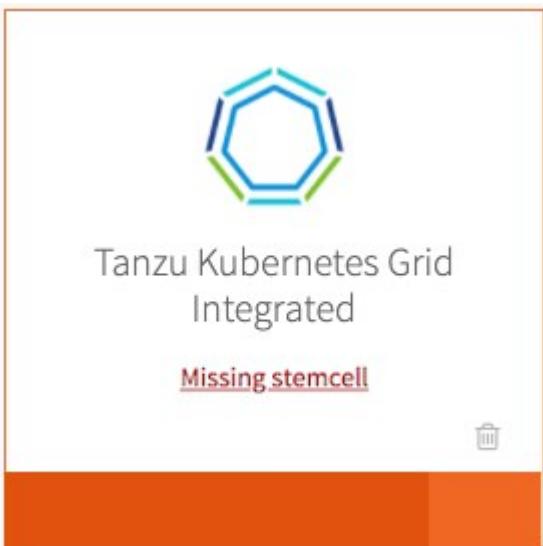
TKGI requires a Ubuntu Jammy stemcell. A stemcell for Windows 2019 is also required if you intend to create Windows worker-based clusters. For information about Windows stemcells, see [Configuring Windows Worker-Based Clusters](#).



**Warning:** If you use an automated pipeline to upgrade TKGI, see [Configure Automated Ops Manager and Ubuntu Jammy Stemcell Downloading](#) in *Configuring the Upgrade Pipeline*.

If Ops Manager does not have the Ubuntu Jammy stemcell required for TKGI v1.16, the TKGI tile displays the message **Missing stemcell**. To download and import a new Ubuntu Jammy stemcell, follow the steps below:

- On the TKGI tile, click the **Missing stemcell** link.



- In the **Stemcell Library**, locate the **TKGI** tile and note the required stemcell version.
- Navigate to the [Stemcells \(Ubuntu Jammy\)](#) page on VMware Tanzu Network and download the required stemcell version for your IaaS.
- Return to the **Installation Dashboard** in Ops Manager and click **Stemcell Library**.
- On the **Stemcell Library** page, click **Import Stemcell** and select the stemcell file you downloaded from VMware Tanzu Network.
- Select the TKGI tile and click **Apply Stemcell to Products**.

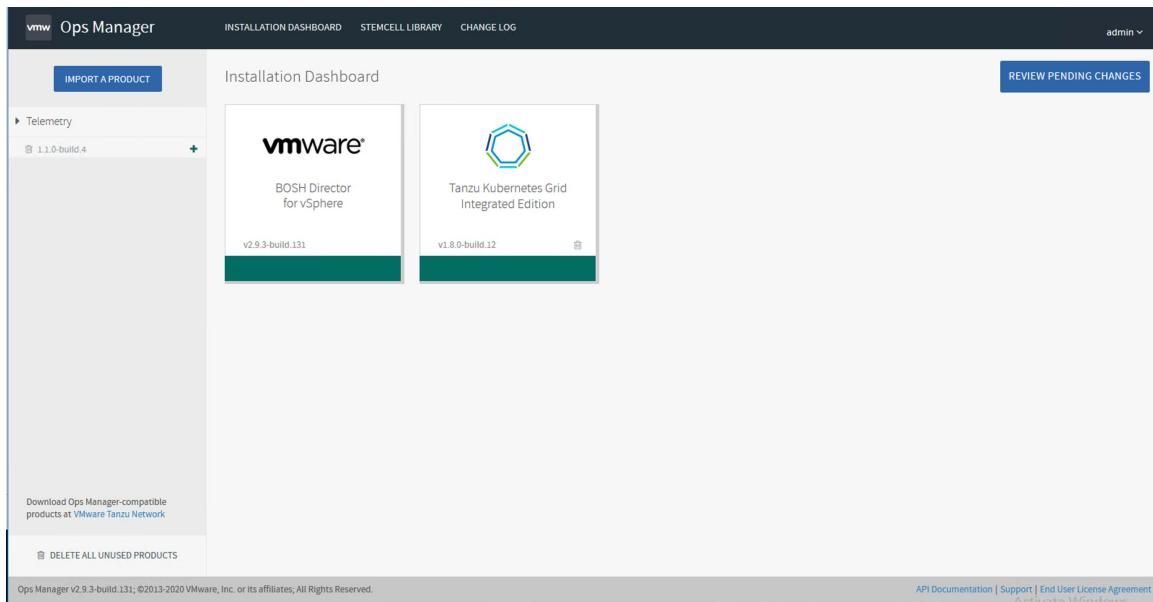
7. Verify that Ops Manager successfully applied the stemcell. The stemcell version you imported and applied appears in the **Staged** column for TKGI.
8. Return to the **Installation Dashboard**.

## Upgrade the TKGI Tile

To complete the upgrade of the TKGI tile:

1. Return to the **Installation Dashboard** in Ops Manager.
2. Click **Review Pending Changes**. For more information about this Ops Manager page, see [Reviewing Pending Product Changes](#).
3. Click **Apply Changes**.

4. (Optional) If you activated the **Upgrade all clusters errand**, you can use the BOSH CLI to monitor its progress:
  1. Log in to the BOSH Director by running `bosh -e MY-ENVIRONMENT log-in` from a VM that can access your TKGI deployment. For more information, see [Using BOSH Diagnostic Commands in Tanzu Kubernetes Grid Integrated Edition](#).
  2. Run `bosh -e MY-ENVIRONMENT tasks`.
  3. Locate the task number for the errand in the `#` column of the BOSH output.
  4. Run `bosh task TASK-NUMBER`, replacing `TASK-NUMBER` with the task number you located in the previous step.
5. Verify that the TKGI tile shows the target version.



## After the Upgrade

After you complete the upgrade to TKGI v1.16, complete the following verifications and upgrades:

1. [Upgrade the TKGI and Kubernetes CLIs](#)
2. [Upgrade Kubernetes Clusters if Needed](#)
3. [Verify TKGI Upgrade](#)
4. [Upgrade NSX-T Data Center to v3.2.2](#)

## Upgrade the TKGI and Kubernetes CLIs

Upgrade the TKGI and Kubernetes CLIs on any local machine where you run commands that interact with your upgraded version of TKGI.

To upgrade the CLIs, download and re-install the TKGI and Kubernetes CLI distributions that are provided with TKGI on VMware Tanzu Network.

For more information about installing the CLIs, see the following topics:

- [Installing the TKGI CLI](#)
- [Installing the Kubernetes CLI](#)

## Upgrade Kubernetes Clusters If Needed

If you upgraded TKGI with the **Upgrade all clusters errand** deactivated, the next step is to upgrade the Kubernetes clusters individually using the TKGI CLI.

1. Log in to the TKGI environment using the [TKGI CLI](#).
2. Run the command `tkgi clusters` to list all Kubernetes clusters with their current versions and status:

| TKGI Version | Name                 | k8s Version | Plan Name | UUID                 |
|--------------|----------------------|-------------|-----------|----------------------|
| 1.15.0       | tkgi-cluster-1-small | 1.24.1      | small     | 0bea03c8-af47-48e8-b |

|                              |        |        |                      |                  |
|------------------------------|--------|--------|----------------------|------------------|
| 249-814c0bc407b9             |        |        |                      |                  |
| 1.15.0 tkgi-cluster-2-medium | 1.24.1 | medium | 5d9f4501-70cb-460b-9 |                  |
| d78-0afbc074cb8c             |        |        |                      |                  |
| 1.15.0 tkgi-cluster-3-large  | 1.24.1 | large  | b448117a-bb6f-49de-b | c9b-452588bd44ef |

3. Upgrade each cluster one-by-one using the command `tkgi upgrade-cluster CLUSTER-NAME`.
  - ◊ You do not have to wait for each upgrade to complete before upgrading the next one.
  - ◊ The advantage of running each upgrade separately is that it makes troubleshooting easier. BOSH assigns a unique task ID to each cluster upgrade.
4. When the cluster upgrades are complete, run the command `tkgi clusters` and verify that they list the target version:

| TKGI Version     | Name                  | k8s Version | Plan Name | UUID                 |
|------------------|-----------------------|-------------|-----------|----------------------|
|                  | Status                | Action      |           |                      |
| 1.16.0           | tkgi-cluster-1-small  | 1.25.4      | small     | 0bea03c8-af47-48e8-b |
| 249-814c0bc407b9 | succeeded             | UPGRADE     |           |                      |
| 1.16.0           | tkgi-cluster-2-medium | 1.25.4      | medium    | 5d9f4501-70cb-460b-9 |
| d78-0afbc074cb8c | succeeded             | UPGRADE     |           |                      |
| 1.16.0           | tkgi-cluster-3-large  | 1.25.4      | large     | b448117a-bb6f-49de-b |
| c9b-452588bd44ef | succeeded             | UPGRADE     |           |                      |

## Verify TKGI Upgrade

1. To verify successful upgrade, create a test cluster:

```
tkgi create-cluster tkgi-cluster-4-test --external-hostname tkgi-cluster-test - -plan medium --num-nodes 3
```



**Note:** Use only lowercase characters when naming your cluster if you manage your clusters with Tanzu Mission Control (TMC). Clusters with names that include an uppercase character cannot be attached to TMC.

2. Run `tkgi clusters` to verify that the new cluster is created with the appropriate version of TKGI and Kubernetes:

| \$ tkgi clusters |                       |             |           |                      |  |
|------------------|-----------------------|-------------|-----------|----------------------|--|
| TKGI Version     | Name                  | k8s Version | Plan Name | UUID                 |  |
|                  | Status                | Action      |           |                      |  |
| 1.16.0           | tkgi-cluster-4-test   | 1.25.4      | medium    | 5d9f4501-70cb-460b-9 |  |
| d78-0afbc074cb8c | succeeded             | CREATE      |           |                      |  |
| 1.16.0           | tkgi-cluster-1-small  | 1.25.4      | small     | 0bea03c8-af47-48e8-b |  |
| 249-814c0bc407b9 | succeeded             | UPGRADE     |           |                      |  |
| 1.16.0           | tkgi-cluster-2-medium | 1.25.4      | medium    | 5d9f4501-70cb-460b-9 |  |
| d78-0afbc074cb8c | succeeded             | UPGRADE     |           |                      |  |
| 1.16.0           | tkgi-cluster-3-large  | 1.25.4      | large     | b448117a-bb6f-49de-b |  |
| c9b-452588bd44ef | succeeded             | UPGRADE     |           |                      |  |

## Maintaining Workload Uptime

This topic describes how you can maintain workload uptime for Kubernetes clusters deployed with VMware Tanzu Kubernetes Grid Integrated Edition.

To maintain workload uptime, configure the following settings in your deployment manifest:

1. Configure [workload replicas](#) to handle traffic during rolling upgrades.
2. Define an [anti-affinity rule](#) to evenly distribute workloads across the cluster.

To increase uptime, you can also refer to the documentation for the services that run on your clusters, and configure your workload based on the recommendations of the software vendor.

## About Cluster Upgrades

The Tanzu Kubernetes Grid Integrated Edition tile contains an errand that upgrades all Kubernetes clusters. Upgrades run on a single VM at a time:

- While a control plane VM is upgraded, the VM's workloads are distributed to the cluster's remaining control plane VMs.
- While a worker VM is upgraded, the workload on that VM goes down. The cluster's additional worker VMs continue to run replicas of your workload, maintaining the uptime of your workload.



**Note:** Ensure that your pods are bound to a *ReplicaSet* or *Deployment*.

Naked pods are not rescheduled in the event of a node failure. For more information, see [Configuration Best Practices](#) in the Kubernetes documentation.

Upgrading a cluster with only a single control plane or worker VM results in a workload outage.

To prevent workload downtime during a cluster upgrade, VMware recommends the following:

- Ensure none of the control plane VMs being upgraded will become overloaded during the cluster upgrade. See [Control Plane Node VM Size](#) for more information.
- Run your workload on at least three worker VMs and using multiple replicas of your workloads spread across those VMs. You must edit your manifest to define the replica set and configure an anti-affinity rule to ensure that the replicas run on separate worker nodes.

## Set Workload Replicas

Set the number of workload replicas to handle traffic during rolling upgrades. To replicate your workload on additional worker VMs, deploy the workload using a replica set.

Edit the `spec.replicas` value in your deployment manifest:

```
kind: Deployment
metadata:
 # ...
spec:
 replicas: 3
```

```
template:
 metadata:
 labels:
 app: APP-NAME
```

See the following table for more information about this section of the manifest:

| Key-Value Pair                     | Description                                                                                         |
|------------------------------------|-----------------------------------------------------------------------------------------------------|
| <pre>spec:   replicas:     3</pre> | Set this value to at least 3 to have at least three instances of your workload running at any time. |
| <pre>app: APP-NAME</pre>           | Use this app name when you define the anti-affinity rule later in the spec.                         |

## Define an Anti-Affinity Rule

To distribute your workload across multiple worker VMs, you must use anti-affinity rules. If you do not define an anti-affinity rule, the replicated pods can be assigned to the same worker node. See the [Kubernetes documentation](#) for more information about anti-affinity rules.

To define an anti-affinity rule, add the `spec.template.spec.affinity` section to your deployment manifest:

```
kind: Deployment
metadata:
 # ...
spec:
 replicas: 3
 template:
 metadata:
 labels:
 app: APP-NAME
 spec:
 containers:
 - name: MY-APP
 image: MY-IMAGE
 ports:
 - containerPort: 12345
 affinity:
 podAntiAffinity:
 requiredDuringSchedulingIgnoredDuringExecution:
 - labelSelector:
 matchExpressions:
 - key: "app"
 operator: In
 values:
 - APP-NAME
 topologyKey: "kubernetes.io/hostname"
```

See the following table for more information:

| Key-Value Pair                                                               | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>podAntiAffinity:   requiredDuringSchedulingIgnoredDuringExecution</pre> | <ul style="list-style-type: none"> <li>When you set <code>podAntiAffinity</code> to the <code>requiredDuringSchedulingIgnoredDuringExecution</code> value, the pod is eligible to be scheduled only on worker nodes that are not running a replica of this pod. If the requirement cannot be met, scheduling fails.</li> <li>Alternatively, you can set <code>podAntiAffinity</code> to the <code>preferredDuringSchedulingIgnoredDuringExecution</code> value. With this rule, the scheduler tries to schedule pod replicas on different worker nodes. If it is not possible, the scheduler assigns more than one pod to the same worker node.</li> </ul> |
| <pre>matchExpressions: - key: "app"</pre>                                    | This value matches <code>spec.template.metadata.labels.app</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <pre>values: - APP-NAME</pre>                                                | This value matches the <code>APP-NAME</code> you defined earlier in the spec.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

## Multi-AZ Worker

Kubernetes evenly spreads pods in a replication controller over multiple Availability Zones (AZs). For more granular control over scheduling pods, add an [Anti-Affinity Rule](#) to the deployment spec by replacing `"kubernetes.io/hostname"` with `"failure-domain.beta.kubernetes.io/zone"`.

For more information about scheduling pods, see [Advanced Scheduling in Kubernetes](#) on the Kubernetes Blog.

## PersistentVolumes

If an AZ goes down, PersistentVolumes (PVs) and their data also go down and cannot be automatically re-attached. To preserve your PV data in the event of a fallen AZ, your persistent workload needs to have a failover mechanism in place.

Depending on the underlying storage type, PVs are either completely free of zonal information or can have multiple AZ labels attached. Both options enable a PV to travel between AZs.

To ensure the uptime of your PVs during a cluster upgrade, VMware recommends that you have at least two nodes per AZ. By configuring your workload as suggested, Kubernetes reschedules pods in the other node of the same AZ while BOSH is performing the upgrade.

For information about configuring PVs in Tanzu Kubernetes Grid Integrated Edition, see [Configuring and Using PersistentVolumes](#).

For information about the supported storage topologies for Tanzu Kubernetes Grid Integrated Edition on vSphere, see [PersistentVolume Storage Options on vSphere](#).

## Configuring the Upgrade Pipeline

This topic describes how to set up a Concourse pipeline to perform automatic upgrades of a VMware Tanzu Kubernetes Grid Integrated Edition installation.

When you configure the upgrade pipeline, the pipeline upgrades your installation when a new Tanzu Kubernetes Grid Integrated Edition release becomes available on VMware Tanzu Network.

By default, the pipeline upgrades when a new major patch version is available.

For more information about configuring and using Concourse for continuous integration (CI), see the [Concourse documentation](#).

## Download the Upgrade Pipeline

Perform the following steps:

1. From a browser, log in to [VMware Tanzu Network](#).
2. Navigate to the **Platform Automation Tools** product page to download the upgrade-tile pipeline.



**Note:** If you cannot access Platform Automation Tools on VMware Tanzu Network, contact Support.

3. (Optional) Edit `params.yml` to configure the pipeline.
  - ◊ For example, edit the `product_version_regex` value to follow minor version updates.
4. Set the pipeline using the `fly` CLI for Concourse. See the [upgrade-tile pipeline documentation](#) for more information.

## Configure Automated Ops Manager and Ubuntu Jammy Stemcell Downloading

If you use an automated pipeline to upgrade TKGI, you must configure your pipeline to download only Ops Manager and Ubuntu Jammy Stemcell versions that are compatible with your version of TKGI.



**Warning:** VMware recommends that you review the Tanzu Network metadata for your version of TKGI and confirm Ops Manager and stemcell version compatibility before using the Tanzu Network APIs to update Ops Manager and Ubuntu Jammy Stemcells in your automated pipeline.

To configure your automated TKGI upgrade pipeline:

1. To retrieve the Ops Manager and Ubuntu Jammy Stemcell versions compatible with your TKGI version:

```
curl -X GET https://network.tanzu.vmware.com/api/v2/products/pivotal-container-service/releases/RELEASE-ID/dependencies
```

Where `RELEASE-ID` is the VMware Tanzu Network ID for your TKGI version. You can see the `RELEASE-ID` in the VMware Tanzu Network URL for your TKGI version.

For example:

The following example returns the versions of TKGI dependencies that are compatible with TKGI v1.16:

```
curl -X GET https://network.tanzu.vmware.com/api/v2/products/pivotal-container-service/releases/1250933/dependencies
{
 "dependencies": [
 {
 "release": {
 "id": 1249995,
 "version": "3.0.4",
 "product": {
 "id": 208,
 "slug": "ops-manager",
 "name": "VMware Tanzu Operations Manager"
 },
 "_links": {
 "self": {
 "href": "https://network.tanzu.vmware.com/api/v2/products/ops-manager/releases/1249995"
 }
 }
 },
 {
 "release": {
 "id": 1249953,
 "version": "1.83",
 "product": {
 "id": 206,
 "slug": "stemcells-ubuntu-jammy",
 "name": "Pivotal Stemcells (Ubuntu Jammy)"
 },
 "_links": {
 "self": {
 "href": "https://network.tanzu.vmware.com/api/v2/products/stemcells-ubuntu-jammy/releases/1249953"
 }
 }
 }
 }
],
 "_links": {
 "self": {
 "href": "https://network.tanzu.vmware.com/api/v2/products/pivotal-container-service/releases/1250933/dependencies"
 }
 }
 }
}
```

The example returned metadata above indicates that TKGI v1.16 is compatible with only Ops Manager versions v3.0.4 and Ubuntu Jammy Stemcell version v1.83.

2. Configure your automated pipeline to upgrade to only a component version specified in the returned metadata for your version of TKGI.

For more information on retrieving TKGI dependencies, see [API reference](#) in the VMware Tanzu

Network documentation.

# Managing Tanzu Kubernetes Grid Integrated Edition

This section describes how to manage VMware Tanzu Kubernetes Grid Integrated Edition.

See the following topics:

- Monitor and Manage Tanzu Kubernetes Grid Integrated Edition in the Management Console
- Managing Tanzu Kubernetes Grid Integrated Edition Users
- Managing Kubernetes Cluster Options
- Adding Infrastructure Password Changes to the Tanzu Kubernetes Grid Integrated Edition Tile
- Rotating Tanzu Kubernetes Grid Integrated Edition Control Plane Certificates
- Shutting Down and Restarting Tanzu Kubernetes Grid Integrated Edition
- Deleting Tanzu Kubernetes Grid Integrated Edition

## Monitor and Manage Tanzu Kubernetes Grid Integrated Edition in the Management Console

After you have deployed VMware Tanzu Kubernetes Grid Integrated Edition on vSphere, you can use VMware Tanzu Kubernetes Grid Integrated Edition Management Console to perform the following operations:

- View the [overall status](#) of your deployment.
- View the [deployment metadata](#) and [status](#) of each of the components of your deployment.
- Edit the configuration of your deployment, either in the configuration wizard or by editing the YAML file. For information about reconfiguring your deployment, see [Reconfigure Your Tanzu Kubernetes Grid Integrated Edition Deployment](#).
- Upgrade your deployment to a new version. For information about upgrading deployments, see [Upgrade Tanzu Kubernetes Grid Integrated Edition Management Console](#).
- Patch the individual components of your deployment. For information about patching components, see [Patch Tanzu Kubernetes Grid Integrated Edition Management Console Components](#).
- [Delete Your Tanzu Kubernetes Grid Integrated Edition Deployment](#).

For information about how to deploy the management console and install Tanzu Kubernetes Grid Integrated Edition, see [Install on vSphere with the Management Console](#).

## Obtain General Status Information

1. Go to the **TKG Integrated Edition** view of the management console.
  2. Select the **Summary** tab for your Tanzu Kubernetes Grid Integrated Edition instance.
- You see general information about your deployment, including the status and version of each component, as well as the names and addresses of the VMs that run those services.

| Component                                | Status              | Version                                  | IP Address                              |
|------------------------------------------|---------------------|------------------------------------------|-----------------------------------------|
| TANZU KUBERNETES GRID INTEGRATED EDITION | ACTIVE              | 1.5.0                                    | vm-4fb44987-52d5-4elb-bf2b-b44dcc58fb4b |
|                                          | VM                  |                                          | vm-4fb44987-52d5-4elb-bf2b-b44dcc58fb4b |
|                                          | IP Address          | 30.0.0.12                                |                                         |
|                                          | Kubernetes Version  |                                          |                                         |
|                                          | Kubernetes Clusters | 5                                        |                                         |
| NETWORK                                  | Network Stack       | NSX-T Data Center                        |                                         |
|                                          | Version             | 2.4                                      |                                         |
|                                          | NSX Manager         | https://11.11.11.11                      |                                         |
|                                          | Pods IP Block       | 564d6439-4abb-e39c-1a2f-d2524e3cc3e1     |                                         |
|                                          | Management Network  | 5006d98a-352f-134f-df6b-33e7f8d5de65     |                                         |
| OPS MANAGER                              | Status              | ACTIVE                                   |                                         |
|                                          | Version             | 2.6                                      |                                         |
|                                          | VM                  | opsman-3x6j2MxQK                         |                                         |
|                                          | IP Address          | 192.168.160.100                          |                                         |
| BOSH                                     | Status              | ACTIVE                                   |                                         |
|                                          | VM                  | vm-50bf8b8c-d17c-4fe4-b2f5-ae607abbff459 |                                         |
|                                          | IP Address          | 30.0.0.11                                |                                         |
| HARBOR                                   | Status              | ACTIVE                                   |                                         |
|                                          | Version             | 1.8.1                                    |                                         |
|                                          | VM                  |                                          |                                         |
|                                          | IP Address          | 192.168.160.103                          |                                         |
| WINDOWS STEMCELL                         | Status              | NOT INSTALLED                            | -                                       |
|                                          | Version             |                                          |                                         |

[View a larger version of this image](#)

## Obtain Deployment Metadata

The deployment metadata provides credentials, certificates, and other metadata about your Tanzu Kubernetes Grid Integrated Edition deployment.

1. Expand **Configuration** and select **Deployment Metadata**.

The screenshot shows the VMware Enterprise PKS Management Console interface. On the left, there's a navigation sidebar with sections like ADMINISTRATION (Identity Management, Quotas, Network Profiles, Configuration), PKS Configuration, PKS Instance Upgrade, and PKS Component Patch. The main content area is titled 'Deployment Metadata' and contains a table with columns 'Description' and 'Value'. The table lists various deployment parameters such as BOSH CA Certificate, BOSH CLI invocation from console appliance, BOSH CLI invocation from Ops Manager, Enterprise PKS TLS, Enterprise PKS UAA Client, Enterprise PKS Services Admin UAA Client, Enterprise PKS UAA Management Admin Client Credential, Enterprise PKS UAA Client Credentials, Enterprise PKS UAA Encryption Passphrase, and Enterprise PKS VM Credentials. Each row has a clipboard icon at the end.

[View a larger version of this image](#)

2. Select the clipboard icon at the end of each row to copy the relevant value.

For example, copy the Ops Manager password so that you can log in to the instance of Ops Manager that is running in your deployment.

## View Component Deployment Status

You can see the status of the individual components of your deployment.

1. Expand **Configuration** and go to the **TKGI Configuration** view of the management console.
2. Select **Deployment Status** to see the status of the components.

The screenshot shows the VMware Tanzu Kubernetes Grid Integrated Edition Management Console. The left sidebar includes sections for TKG Integrated Edition (Quotas, Network Profiles), ADMINISTRATION (Identity Management, Configuration, Deployment Metadata, TKGI Configuration, TKGI Instance Upgrade, TKGI Component Patch), and a navigation bar with 'root' and 'DARK' options. The main area is titled 'TKGI Configuration' with tabs for 'WIZARD' and 'DEPLOYMENT STATUS'. It displays the 'Current configuration status of VMware Tanzu Kubernetes Grid Integrated Edition'. A message states 'This installation was successfully deployed and all components are up and running.' Below this is a table showing component status:

| Operations  | State     | Status    |
|-------------|-----------|-----------|
| NSX         | ✓ Success | Completed |
| Ops Manager | ✓ Success | Completed |
| BOSH        | ✓ Success | Completed |
| PKS         | ✓ Success | Completed |
| Harbor      | ✓ Success | Completed |

A 'CONTINUE' button is at the bottom.

[View a larger version of this image](#)

3. Click the Download Logs button to download the log bundle for your Tanzu Kubernetes Grid Integrated Edition deployment.

## Reconfigure Your Tanzu Kubernetes Grid Integrated Edition Deployment

After deployment, you can reconfigure your Tanzu Kubernetes Grid Integrated Edition installation in VMware Tanzu Kubernetes Grid Integrated Edition Management Console, either by using the wizard or by importing an updated YAML file.

## Reconfigure Your Tanzu Kubernetes Grid Integrated Edition Deployment in the Wizard

The procedure to reconfigure Tanzu Kubernetes Grid Integrated Edition deployments in the wizard is as follows.

1. Expand **Configuration** and go to the **TKGI Configuration** view of the management console.
2. Select **Wizard** to be taken to the configuration wizard for Tanzu Kubernetes Grid Integrated Edition.

|                          |                                                                                  |
|--------------------------|----------------------------------------------------------------------------------|
| > 1. vCenter Account     | VC: 192.168.111.36, Username: administrator@vsphere.local                        |
| > 2. Networking          | NSX Manager: 192.168.111.41, Username: admin                                     |
| > 3. Identity            | Identity Management Source: Local user database                                  |
| > 4. Availability Zones  | Configured Availability Zones: az-1, az-2, az-3                                  |
| > 5. Resources & Storage | Ephemeral: Datastore 2; Permanent: Datastore 4; Kubernetes: Datastore 2          |
| > 6. Plans               | test, medium, large                                                              |
| > 7. Integrations        | Wavefront, Syslog enabled                                                        |
| > 8. Harbor              | Harbor registry is configured and will be installed                              |
| > 9. CEIP & Telemetry    | VMware's Customer Experience Improvement Program and Pivotal's Telemetry Program |

**GENERATE CONFIGURATION**    **EXIT**    **IMPORT YAML**

[View a larger version of this image](#)

3. Expand the different sections of the wizard and change the configurations as necessary.

For information about which configuration options you can change, see [Which Options Can I Reconfigure?](#) below.

For the options that you can modify, refer to [Deploy Tanzu Kubernetes Grid Integrated Edition by Using the Configuration Wizard](#) for instructions about how to fill in each section.

4. When you have finished reconfiguring, click **Generate Configuration**.
5. Optionally export the `PksConfiguration.yaml` file to save a copy of your configuration.
6. Click **Apply Configuration** and **Continue** to complete the reconfiguration of Tanzu Kubernetes Grid Integrated Edition.

## Reconfigure Your Tanzu Kubernetes Grid Integrated Edition Deployment by Importing a YAML File

The procedure to reconfigure Tanzu Kubernetes Grid Integrated Edition deployments by importing

an updated YAML file is as follows.

1. Expand **Configuration** and go to the **TKGI Configuration** view of the management console.
2. Select **Wizard** to be taken to the configuration wizard for Tanzu Kubernetes Grid Integrated Edition.
3. Scroll to the bottom of the screen and click **Import YAML**.

The screenshot shows the 'TKGI Configuration' wizard interface. At the top, there are two tabs: 'WIZARD' (which is selected) and 'DEPLOYMENT STATUS'. Below the tabs is a list of configuration items with their details:

- > 1. vCenter Account VC: 192.168.111.36, Username: administrator@vsphere.local
- > 2. Networking NSX Manager: 192.168.111.41, Username: admin
- > 3. Identity Identity Management Source: Local user database
- > 4. Availability Zones Configured Availability Zones: az-1, az-2, az-3
- > 5. Resources & Storage Ephemeral: Datastore 2; Permanent: Datastore 4; Kubernetes: Datastore 2
- > 6. Plans test, medium, large
- > 7. Integrations Wavefront, Syslog enabled
- > 8. Harbor Harbor registry is configured and will be installed
- > 9. CEIP & Telemetry VMware's Customer Experience Improvement Program and Pivotal's Telemetry Program

At the bottom of the configuration list are three buttons: 'GENERATE CONFIGURATION' (blue), 'EXIT' (light blue), and 'IMPORT YAML' (red box).

4. Drag the YAML file into the Import Configuration File window, or click **Browse** to navigate to it.
5. In the Configuration File editor, modify the contents of the YAML file appropriately for the new instance of Tanzu Kubernetes Grid Integrated Edition that you want to deploy.

For information about which configuration options you can change, see [Which Options Can I Reconfigure?](#) below.

If the YAML was generated by an instance of the management console that is running in a different vSphere environment, update the passwords for NSX Manager, vCenter Server, and Harbor. For more information see [Deploy Tanzu Kubernetes Grid Integrated Edition by Importing a YAML Configuration File](#).

To abandon this YAML and start again, click **Import** to upload the YAML again or to import a new one.

You can also click the **Edit in Wizard** button, to open the imported configuration in the wizard.

6. Click **Apply Configuration** and **Continue** to complete the reconfiguration of Tanzu Kubernetes Grid Integrated Edition.

## Which Options Can I Reconfigure?

After the initial deployment of Tanzu Kubernetes Grid Integrated Edition, there are certain options that you cannot reconfigure. In particular, you cannot make significant infrastructure changes.

The table below lists what you can and cannot modify on an existing Tanzu Kubernetes Grid Integrated Edition deployment.

| Section                                                  | Cannot Modify                                                                                                                                                                           | Can Modify                                                                                                                                             |
|----------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| vCenter Account                                          | <ul style="list-style-type: none"> <li>vCenter Server instance</li> <li>Datacenter</li> </ul>                                                                                           | <ul style="list-style-type: none"> <li>Username</li> <li>Password</li> <li>vCenter Server address, if it changes</li> </ul>                            |
| Networking: Container Networking Interface               | You cannot change between <b>NSX-T Data Center (Automated NAT Deployment)</b> , <b>NSX-T Data Center (Bring Your Own Topology)</b> , and <b>vSphere without NSX-T</b> .                 | None.                                                                                                                                                  |
| Networking: NSX Manager Details                          | You cannot change the deployment to a different NSX Manager instance.                                                                                                                   | <ul style="list-style-type: none"> <li>Username</li> <li>Password</li> <li>NSX Manager address, if it changes</li> <li>NSX Manager CA Cert*</li> </ul> |
| Networking: NSX-T Data Center (Automated NAT Deployment) | <ul style="list-style-type: none"> <li>Tier0 Active-Active Mode</li> <li>Deployment CIDR</li> <li>Deployment Network Reserved IP Range</li> <li>Usable Range of Floating IPs</li> </ul> | All other options.                                                                                                                                     |
| Networking: NSX-T Data Center (Bring Your Own Topology)  | <ul style="list-style-type: none"> <li>Network for TKGI Management Plane</li> <li>Floating IP Pool ID</li> <li>Deployment Network Reserved IP Range</li> <li>NAT mode</li> </ul>        | All other options.                                                                                                                                     |
| Networking: vSphere Without NSX-T                        | <ul style="list-style-type: none"> <li>Deployment Network Reserved IP Range</li> <li>Service Network Reserved IP Range</li> </ul>                                                       | All other options.                                                                                                                                     |
| Identity                                                 | None.                                                                                                                                                                                   | All options**.                                                                                                                                         |
| Availability Zones                                       | <ul style="list-style-type: none"> <li>Management availability zone</li> </ul>                                                                                                          | All other options, including adding and deleting availability zones.                                                                                   |
| Plans                                                    | None.                                                                                                                                                                                   | All options, including adding and deleting plans.                                                                                                      |
| Integrations                                             | None.                                                                                                                                                                                   | All options.                                                                                                                                           |

|                    |                                                                                                |                    |
|--------------------|------------------------------------------------------------------------------------------------|--------------------|
| Harbor             | <ul style="list-style-type: none"> <li>• Harbor FQDN</li> <li>• Authentication mode</li> </ul> | All other options. |
| CEIP and Telemetry | None.                                                                                          | All options.       |

\* If you update the **NSX Manager CA Cert** field, TKGI MC will update BOSH and TKGI to the new NSX-T CA certificate. For clusters that you have already created, you must update those clusters to use the new certificate. You must manually update the NSX-T CA certificate for the related NCP configurations.

\*\* If you previously used a local user database and change to **AD/LDAP** or **SAML**, the local user database is deleted.

## Patch Tanzu Kubernetes Grid Integrated Edition Management Console Components

You can use VMware Tanzu Kubernetes Grid Integrated Edition Management Console on vSphere to update some of the components of your deployment individually when a new minor version of those components is available.

1. In Tanzu Kubernetes Grid Integrated Edition Management Console, go to **Configuration > TKGI Component Patch** to view the list of components that are ready for patching.
2. Obtain the patch installers.
  - ◊ In air-gapped environments, download the patch installer from <https://downloads.vmware.com/> to a local location. Click the **Import Patch** button to upload the installer to the management console.
  - ◊ In environments with access to the internet, click the **Download** button next to the relevant components to import the patch installers directly.

The screenshot shows the VMware Enterprise PKS Management Console interface. On the left, there's a sidebar with navigation links: 'Enterprise PKS' (selected), 'ADMINISTRATION' (with 'Identity Management', 'Deployment Metadata', and 'Configuration' sub-links), 'PKS Configuration', 'PKS Instance Upgrade', and 'PKS Component Patch'. The main content area is titled 'VMware Enterprise PKS Component Patches'. It displays a table with one row for 'gear2'. The table columns are 'COMPONENT', 'CURRENT VERSION', 'NEW VERSION', and 'DESCRIPTION'. The 'COMPONENT' column shows 'gear2'. The 'CURRENT VERSION' column shows '1.6.0-rev.2'. The 'NEW VERSION' column shows '1.6.0-rev.3'. The 'DESCRIPTION' column states 'Enterprise PKS Management Console Patch up-to-date.' To the right of the table is a blue 'DOWNLOAD' button. At the top of the page, there's a banner stating 'There are new patches to the VMware Enterprise PKS components' with a 'MORE INFO' link.

- When the patch imports are complete, select **Install Patch** to patch a component.

The screenshot shows the 'VMware Enterprise PKS Component Patches' table. The table has one row for 'gear2'. The columns are 'COMPONENT', 'CURRENT VERSION', 'NEW VERSION', and 'DESCRIPTION'. 'COMPONENT' is 'gear2'. 'CURRENT VERSION' is '1.6.1-rev.1'. 'NEW VERSION' is '3.0.0-rev.1'. 'DESCRIPTION' is 'VMware Enterprise PKS Management Console Service'. To the right of the table is a blue 'INSTALL PATCH' button. Below the table, a note says 'For more information please refer to the Help: ⓘ'.

If you are patching Tanzu Kubernetes Grid Integrated Edition Management Console itself, you will be automatically logged out during the patching process.

- Log back in to the management console.
- Click the help icon ⓘ in the top banner and select **About** to check that the version of Tanzu Kubernetes Grid Integrated Edition Management Console has been updated.
- Click **TKG Integrated Edition** to check the versions of the installed components.

## Delete Your Tanzu Kubernetes Grid Integrated Edition Deployment

If you no longer require an Tanzu Kubernetes Grid Integrated Edition deployment on vSphere, you can use VMware Tanzu Kubernetes Grid Integrated Edition Management Console to delete it. The deletion process removes all objects from the vSphere inventory, and cleans up the objects in the Tanzu Kubernetes Grid Integrated Edition Management Console VM related to your deployment.

If Tanzu Kubernetes Grid Integrated Edition fails to deploy correctly, always use Tanzu Kubernetes

Grid Integrated Edition Management Console to cleanly remove the failed deployment.



**Note:** You must use the TKGI CLI to delete any existing clusters and nodes before you can use the management console to delete your Tanzu Kubernetes Grid Integrated Edition deployment.

1. Go to the **TKG Integrated Edition** view of the management console.
2. Click the **Action** drop-down menu and select **Delete Tanzu Kubernetes Grid Integrated Edition Deployment**
3. Click **Delete** to confirm the deletion of the deployment.

## Managing Tanzu Kubernetes Grid Integrated Edition Users

This section describes how to use either VMware Tanzu Kubernetes Grid Integrated Edition Management Console or Ops Manager to manage users.

See the following topics:

- [Identity Management in the Management Console](#)
- [Identity Management in Ops Manager](#)

## Identity Management in the Management Console

On vSphere, you can add individual users or user groups to Tanzu Kubernetes Grid Integrated Edition Management Console. You can assign roles to individual users or to groups. If you assign a role to a group, all of the users in that group have that role.

For information about the roles that you can assign, see [UAA Scopes for Tanzu Kubernetes Grid Integrated Edition Users](#).

For information about the tasks that Cluster Managers can perform, see [Tanzu Kubernetes Grid Integrated Edition Architecture](#). The TKGI Administrator role allows users to manage the Tanzu Kubernetes Grid Integrated Edition infrastructure.

## Add Individual Users

The procedure to add individual users to Tanzu Kubernetes Grid Integrated Edition Management Console is as follows.



**Note:** This release of Tanzu Kubernetes Grid Integrated Edition Management Console does not support assigning roles to individual LDAP or SAML users. To assign roles to LDAP or SAML users, use user groups.

1. Go to the **Identity Management** view of the management console.
2. Select the **Users** tab.

The screenshot shows the 'Identity Management - Users' page. The left sidebar has sections for TKG Integrated Edition, Quotas, Network Profiles, Administration, and Configuration (Deployment Metadata, TKGI Configuration, TKGI Instance Upgrade, TKGI Component Patch). The main area shows a table of users:

| Name          | Role                                        |
|---------------|---------------------------------------------|
| Eisa Lin      | Unassigned                                  |
| Fang Shui Lin | pks.clusters.manage pks.clusters.admin.read |
| Fei Xu Sun    | pks.clusters.manage                         |
| Mia Li        | pks.clusters.admin                          |
| Mike          | pks.clusters.admin.read                     |
| Han Meimei    | pks.clusters.manage                         |
| Li Lei        | pks.clusters.manage                         |
| Xiao ming     | pks.clusters.admin                          |
| ling ling     | pks.cluster.manage                          |
| sun zheng     | pks.cluster.admin.read                      |

At the bottom, there are pagination controls: 'Users per page' (10), '1-10 of 11 Users', and navigation buttons (K, <, 1, / 2, >, K).

[View a larger version of this image](#)

3. Click **Add User**.
4. Enter a user name and enter and verify a password to create a new user account.

The password that you set must meet the following criteria:

- ◊ Minimum eight characters
- ◊ Maximum 20 characters
- ◊ Minimum one lowercase character
- ◊ Minimum one numeric character
- ◊ Minimum one special character
- ◊ Minimum one uppercase character
- ◊ Maximum 3 identical adjacent characters

## ♂ Add User

Add one or more UAA users to your installation specifying their username, password and associated role.

|          |          |                 |
|----------|----------|-----------------|
| Username | Password | Verify Password |
| newuser  | .....    | .....           |

Assign Role pks.clusters.manage

Unassigned  
pks.clusters.manage  
pks.clusters.admin  
pks.clusters.admin.read

**SAVE**

5. Assign a role to the user.

- ◊ **pks.clusters.manage**: Accounts with this scope can create and access their own clusters.

- ◊ `pks.clusters.admin`: Accounts with this scope can create and access all clusters.
  - ◊ `pks.clusters.admin.read`: Accounts with this scope can access any information about all clusters except for cluster credentials.
6. Click **Save**.
  7. If you do not assign a role to a user when you create or add the account, or to change a user's role, select the user in the **Users** tab, and select **Assign Role**.

## Add User Groups

The procedure to add user groups to Tanzu Kubernetes Grid Integrated Edition Management Console is as follows.

1. Go to the **Identity Management** view of the management console.
2. Select the **Groups** tab.

| Name   | Role            |
|--------|-----------------|
| group1 | Cluster Manager |
| group2 | Cluster Manager |
| group3 | Cluster Manager |

[View a larger version of this image](#)

3. Click **Add Group**.
4. Enter an existing LDAP or SAML user group.
  - ◊ **LDAP**: Enter the distinguished name of an existing LDAP group under the configured group search base, for example `cn=admins,ou=engineering,dc=username,dc=local`.
  - ◊ **SAML**: Enter the name of your SAML identity provider group.
5. Assign a role to the group.
  - ◊ `pks.clusters.manage`: Accounts with this scope can create and access their own clusters.
  - ◊ `pks.clusters.admin`: Accounts with this scope can create and access all clusters.
  - ◊ `pks.clusters.admin.read`: Accounts with this scope can access any information about all clusters except for cluster credentials.

## Add Group

Add one or more groups to your installation specifying the group name and associated role.

Group 

newgroup

Assign Role  pks.clusters.manage

pks.clusters.manage

**pks.clusters.admin**

pks.clusters.admin.read

**SAVE**

6. Click **Save**.



**Note:** You must assign a role to a group when you add it. You cannot assign, change, or revoke a group role after you have added the group.

## Remove Individual Users

The procedure to remove individual users from Tanzu Kubernetes Grid Integrated Edition Management Console is as follows.

1. Go to the **Identity Management** view of the management console.
2. Select the **Users** tab.
3. Select a user.
4. Click **Remove User**.

## Remove User Groups

The procedure to remove individual users from Tanzu Kubernetes Grid Integrated Edition Management Console is as follows.

1. Go to the **Identity Management** view of the management console.
2. Select the **Groups** tab.
3. Select a group.
4. Click **Remove Group**.

## Next Steps

- [Assign Resource Quotas to Users](#)
- [Working with Network Profiles](#)
- [Create Clusters in the Management Console](#)

# Managing Tanzu Kubernetes Grid Integrated Edition Users in Ops Manager

This section describes how to connect UAA to external user stores when using Ops Manager to configure VMware Tanzu Kubernetes Grid Integrated Edition and how to manage users with UAA.

See the following topics:

- [Connecting Tanzu Kubernetes Grid Integrated Edition to an LDAP Server](#)
- [Configuring Okta as a SAML Identity Provider](#)
- [Configuring Azure Active Directory as a SAML Identity Provider](#)
- [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)
- [Managing Tanzu Kubernetes Grid Integrated Edition Users with UAA](#)
- [UAA Scopes for Tanzu Kubernetes Grid Integrated Edition Users](#)
- [OIDC Provider for Kubernetes Clusters](#)

## Connecting Tanzu Kubernetes Grid Integrated Edition to an LDAP Server

This topic describes how to connect VMware Tanzu Kubernetes Grid Integrated Edition to an external LDAP server.

### Overview

User Account and Authentication (UAA), the identity management service for Tanzu Kubernetes Grid Integrated Edition, can authenticate users either through its internal user account store or external authentication mechanisms such as an LDAP server or a SAML identity provider.

To enable an internal user account store for UAA, you select **Internal UAA** in the **Tanzu Kubernetes Grid Integrated Edition** tile > **UAA**.

If you want to connect Tanzu Kubernetes Grid Integrated Edition to an external LDAP server, you must integrate the UAA server with your LDAP server by following the instructions in [Integrate UAA with an LDAP Server](#) below. This enables UAA to delegate authentication to your LDAP user store.

### Integrate UAA with an LDAP Server

To integrate UAA with one or more LDAP servers:

1. In **Tanzu Kubernetes Grid Integrated Edition** > **UAA**, under **Configure your UAA user account store with either internal or external authentication mechanisms**, select **LDAP Server**.

Configure your UAA user account store with either internal or external authentication mechanisms \*

- Internal UAA
- LDAP Server

Server URL \*

`ldaps://example.com`

LDAP Credentials \*

Username

Password

2. Under **Server URL**, enter the URLs that point to your LDAP server. For example, `ldaps://example.com`. If you have multiple LDAP servers, separate their URLs with spaces. Each URL must include one of the following protocols:

- `ldap://`: Enter this protocol if your LDAP server uses an unencrypted connection.
- `ldaps://`: Enter this protocol if your LDAP server uses SSL for an encrypted connection. To support an encrypted connection, the LDAP server must hold a trusted certificate or you must import a trusted certificate to the JVM truststore.

3. Under **LDAP Credentials**, enter the LDAP Distinguished Name (DN) and password for binding to the LDAP server. For example, `cn=administrator,ou=Users,dc=example,dc=com`. If the bind user belongs to a different search base, you must use the full DN.



**Note:** VMware recommends that you provide LDAP credentials that grant read-only permissions on the LDAP search base and the LDAP group search base.

4. Under **User Search Base**, enter the location in the LDAP directory tree where LDAP user search begins. For example, a domain named `cloud.example.com` might use `ou=Users,dc=example,dc=com` as its LDAP user search base.

User Search Base \*

`ou=Users,dc=example,dc=com`

User Search Filter \*

`cn={0}`

Group Search Base

`ou=Groups,dc=example,dc=com`

Group Search Filter \*

member={0}

Server SSL Cert

First Name Attribute

Last Name Attribute

Email Attribute \*

mail

Email Domain(s)

LDAP Referrals\*

Automatically follow any referrals

External Groups Whitelist

\*

Group Max Search Depth

1

- Under **User Search Filter**, enter a string to use for LDAP user search criteria. The search criteria allows LDAP to perform more effective and efficient searches. For example, the standard LDAP search filter `cn=Smith` returns all objects with a common name equal to `Smith`.

In the LDAP search filter string that you use to configure Tanzu Kubernetes Grid Integrated Edition, use `{0}` instead of the username. For example, use `cn={0}` to return all LDAP objects with the same common name as the username. In addition to `cn`, other common attributes are `mail`, `uid`, and for Active Directory, `sAMAccountName`.



**Note:** For information about testing and troubleshooting your LDAP search filters, see [Configuring LDAP Integration with Pivotal Cloud Foundry](#).

6. Under **Group Search Base**, enter the location in the LDAP directory tree where the LDAP group search begins. For example, a domain named `cloud.example.com` might use `ou=Groups,dc=example,dc=com` as its LDAP group search base. You must configure **Group Search Base** if you want to map an external LDAP group to a role in Tanzu Kubernetes Grid Integrated Edition or a Kubernetes group.



**Note:** To map the groups under this search base to roles in Tanzu Kubernetes Grid Integrated Edition, follow the instructions in [Grant Tanzu Kubernetes Grid Integrated Edition Access to an External LDAP Group](#).

7. Under **Group Search Filter**, enter a string that defines LDAP group search criteria. The default value is `member={0}`.
8. Under **Server SSL Cert**, paste in the root certificate from your CA certificate or your self-signed certificate.
9. Under **First Name Attribute**, enter the attribute name in your LDAP directory that contains user first names. For example, `cn`.
10. Under **Last Name Attribute**, enter the attribute name in your LDAP directory that contains user last names. For example, `sn`.
11. Under **Email Attribute**, enter the attribute name in your LDAP directory that contains user email addresses. For example, `mail`.
12. Under **Email Domain(s)**, enter a comma-separated list of the email domains for external users who can receive invitations to Apps Manager.
13. Under **LDAP Referrals**, choose how UAA handles LDAP server referrals to other user stores. UAA can follow the external referrals, ignore them without returning errors, or generate an error for each external referral and abort the authentication.
14. Under **External Groups Whitelist**, enter a comma-separated list of group patterns that need to be populated in the user's `id_token`. For more information about accepted patterns, see the description of `config.externalGroupsWhitelist` in the [OAuth/OIDC Identity Provider Documentation](#).



**Note:** When sent as a Bearer token in the Authentication header, wide pattern queries for users who are members of multiple groups can cause the size of the `id_token` to extend beyond what is supported by web servers.

15. Under **Group Max Search Depth**, enter the LDAP group search depth. Allowed values are

between `1` and `10`. The default value is `1`, which limits queering under the `searchBase` to one subtree. Values greater than `1` activate nested group searching. If the `searchBase` in your LDAP groups includes more than one subtree, for example:

`ou=Groups,ou=Client,ou=DE,dc=fs01,dc=vwf,dc=vwfs-ad`, increase the **Group Max Search Depth** value to support searching all subtrees in your groups.



**Note:** Increasing the LDAP group search depth impacts performance.

16. Click **Save**.

## Complete Your Tile Configuration

- If you do not need to configure any other settings in the Tanzu Kubernetes Grid Integrated Edition tile, return to the Ops Manager Installation Dashboard and click **Review Pending Changes > Apply Changes**.
- If you need to configure any other settings in the Tanzu Kubernetes Grid Integrated Edition tile, return to the *Installing Tanzu Kubernetes Grid Integrated Edition* topic for your IaaS and follow the instructions for the pane you want to configure:
  - [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#)
  - [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T](#)
  - [Installing Tanzu Kubernetes Grid Integrated Edition on GCP](#)
  - [Installing Tanzu Kubernetes Grid Integrated Edition on AWS](#)
  - [Installing Tanzu Kubernetes Grid Integrated Edition on Azure](#)

## Next Steps

For information about creating Tanzu Kubernetes Grid Integrated Edition roles and managing Kubernetes cluster access, see:

- [Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users](#) for your IaaS
- [Managing Cluster Access and Permissions](#)

## Configuring Okta as a SAML Identity Provider

This topic explains how to configure single sign-on (SSO) between Okta and VMware Tanzu Kubernetes Grid Integrated Edition.

## Prerequisites

To configure Okta to designate Tanzu Kubernetes Grid Integrated Edition as a service provider, you must have the following:

- An Okta Single-Sign On admin account
- An app with SAML 2.0 enabled in Okta

## Configure SAML in Okta

To configure Okta as a SAML identity provider for Tanzu Kubernetes Grid Integrated Edition, do the following:

1. Log in to Okta as an admin.
2. Navigate to your app and click **Sign On**.
3. Under **Settings**, click **Edit**, and select **SAML 2.0**.

The screenshot shows the Okta Admin Console with the 'Sign On' tab selected. A red box highlights the 'SAML 2.0' configuration section. Below it, a yellow sidebar provides setup instructions and metadata availability information. At the bottom, there are credential details for application username format and password reveal options.

General    Sign On    Mobile    Import    People    Groups

Settings    Edit

**SIGN ON METHODS**

The sign-on method determines how a user signs into and manages their credentials for an application. Some sign-on methods require additional configuration in the 3rd party application.

SAML 2.0

Default Relay State

**SAML 2.0** is not configured until you complete the setup instructions.  
[View Setup Instructions](#)

Identity Provider metadata is available if this application supports dynamic configuration.

**CREDENTIALS DETAILS**

|                             |                                                                                   |
|-----------------------------|-----------------------------------------------------------------------------------|
| Application username format | Okta username                                                                     |
| Password reveal             | <input type="checkbox"/> Allow users to securely see their password (Recommended) |

4. Click the **General** tab.
5. Under **SAML Settings**, click the **Edit** button followed by the **Next** button.

### A SAML Settings

**GENERAL**

**Single sign on URL** ?   Use this for Recipient URL and Destination URL

**Audience URI (SP Entity ID)** ?

**Default RelayState** ?   
If no value is set, a blank RelayState is sent

**Name ID format** ?

**Application username** ?

[Show Advanced Settings](#)

---

**ATTRIBUTE STATEMENTS (OPTIONAL)** [LEARN MORE](#)

| Name                 | Name format (optional)       | Value                                                  |
|----------------------|------------------------------|--------------------------------------------------------|
| <input type="text"/> | Unspecified <small>▼</small> | <input type="text"/> <small>▼</small> <small>x</small> |
| <input type="text"/> | Unspecified <small>▼</small> | <input type="text"/> <small>▼</small> <small>x</small> |
| <input type="text"/> | Unspecified <small>▼</small> | <input type="text"/> <small>▼</small> <small>x</small> |

[Add Another](#)

---

**GROUP ATTRIBUTE STATEMENTS (OPTIONAL)**

| Name                 | Name format (optional)       | Filter                                                                                           |
|----------------------|------------------------------|--------------------------------------------------------------------------------------------------|
| <input type="text"/> | Unspecified <small>▼</small> | <input style="width: 100px;" type="text" value="Starts with"/> <small>▼</small> <small>x</small> |

[Add Another](#)

6. Configure the fields as follows:

| Field                     | Instructions                                                                                                                                                                       |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Single sign on URL</b> | Enter <code>https://TKGI-API:8443/saml/SSO/alias/TKGI-API:8443</code> .<br>For example:<br><code>https://api.tkgi.example.com:8443/saml/SSO/alias/api.tkgi.example.com:8443</code> |

|                                                       |                                                                                                                                                                                                                                        |
|-------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Use this for Recipient URL and Destination URL</b> | Ensure this checkbox is enabled.                                                                                                                                                                                                       |
| <b>Audience URI (SP Entity ID)</b>                    | Enter <code>TKGI-API:8443</code> .<br>For example: <code>api.tkgi.example.com:8443</code>                                                                                                                                              |
| <b>Name ID format</b>                                 | Select a name identifier format. By default, Tanzu Kubernetes Grid Integrated Edition uses <code>EmailAddress</code> .                                                                                                                 |
| <b>Attribute Statements</b>                           | Enter any attribute statements that you want to map to users in the ID token. In Tanzu Kubernetes Grid Integrated Edition you can define first name, last name, and email attributes.                                                  |
| <b>Group Attribute Statements</b>                     | Enter any group attribute statements that you want to map to users in the ID token. In Okta, these are groups that users belong to. You can use filters to define which groups are passed to Tanzu Kubernetes Grid Integrated Edition. |



**Note:** VMware recommends using the default settings for the fields that are not referenced in the above table.

7. Click the **Next** button followed by the **Finish** button.
8. (Optional) If you want to enable multi-factor authentication (MFA), you can add a SSO policy rule to your app. To enable MFA, do the procedure in [Add Sign On policies for applications](#) in the Okta documentation.
9. Click **Identity Provider metadata** to download the metadata, or copy and save the link address of the **Identity Provider metadata**.

**SIGN ON METHODS**

The sign-on method determines how a user signs into and manages their credentials for an application. Some sign-on methods require additional configuration in the 3rd party application.

**SAML 2.0**

Default Relay State

**SAML 2.0** is not configured until you complete the setup instructions.  
[View Setup Instructions](#)

**Identity Provider metadata** is available if this application supports dynamic configuration.

**CREDENTIALS DETAILS**

|                             |                                                                                   |
|-----------------------------|-----------------------------------------------------------------------------------|
| Application username format | Okta username                                                                     |
| Password reveal             | <input type="checkbox"/> Allow users to securely see their password (Recommended) |

10. Use the Okta metadata you retrieved in the above step to configure SAML in the Tanzu Kubernetes Grid Integrated Edition tile. See [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#).

## Configuring Azure Active Directory as a SAML Identity Provider

This topic explains how to configure single sign-on (SSO) between Azure Active Directory (Azure AD) and VMware Tanzu Kubernetes Grid Integrated Edition.

### Prerequisites

To configure Azure AD to designate Tanzu Kubernetes Grid Integrated Edition as a service provider, you must have an Azure AD Global Administrator account.

### Configure SAML in Azure AD

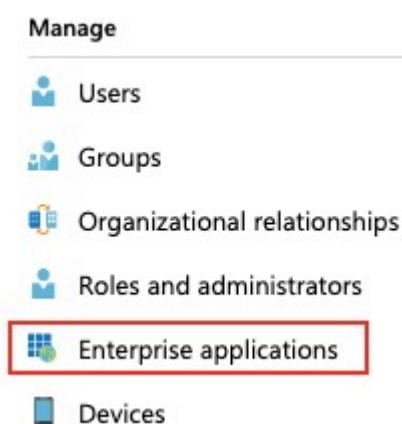
To configure Azure AD as a SAML identity provider for Tanzu Kubernetes Grid Integrated Edition, do the following:

1. Log in to Azure AD as a Global Administrator.

2. Navigate to **Azure Active Directory**.
3. Under **Create**, click **Enterprise application**.



4. Under **Add your own app**, select **Non-gallery application**. Enter a **Name** and click **Add**.
5. Navigate to **Azure Active Directory > Enterprise applications**.



6. Click your app and then click **Single sign-on**.



7. Under **Select a single sign-on method**, select **SAML**.

**SAML**

Rich and secure authentication to applications using the SAML (Security Assertion Markup Language) protocol.

8. Under **Set up Single Sign-On with SAML**, click the pencil icon for **Basic SAML Configuration**.

### Set up Single Sign-On with SAML

Read the [configuration guide](#) for help integrating Test.

**1**

#### Basic SAML Configuration



|                                            |                 |
|--------------------------------------------|-----------------|
| Identifier (Entity ID)                     | <b>Required</b> |
| Reply URL (Assertion Consumer Service URL) | <b>Required</b> |
| Sign on URL                                | <i>Optional</i> |
| Relay State                                | <i>Optional</i> |
| Logout Url                                 | <i>Optional</i> |

9. Configure the following fields:

| Field                         | Instructions                                                                                                                                                                       |
|-------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Identifier (Entity ID)</b> | Enter <code>TKGI-API:8443</code> .<br>For example:<br><code>api.tkgi.example.com:8443</code>                                                                                       |
| <b>Reply URL</b>              | Enter <code>https://TKGI-API:8443/saml/SSO/alias/TKGI-API:8443</code> .<br>For example:<br><code>https://api.tkgi.example.com:8443/saml/SSO/alias/api.tkgi.example.com:8443</code> |
| <b>Sign on URL</b>            | Enter <code>https://TKGI-API:8443/saml/SSO/alias/TKGI-API:8443</code> .<br>For example:<br><code>https://api.tkgi.example.com:8443/saml/SSO/alias/api.tkgi.example.com:8443</code> |



**Note:** VMware recommends that you use the default settings for the fields that are not referenced in the above table.

10. Click the pencil icon for **User Attributes & Claims**.

**2**

| User Attributes & Claims |                        |
|--------------------------|------------------------|
| Givenname                | user.givenname         |
| Surname                  | user.surname           |
| Emailaddress             | user.mail              |
| Name                     | user.userprincipalname |
| Unique User Identifier   | user.userprincipalname |

11. Configure your user attributes and claims by doing the procedures in [How to: Customize claims issued in the SAML token for enterprise applications](#) in the Microsoft Azure documentation. By default, Tanzu Kubernetes Grid Integrated Edition uses the [EmailAddress](#) name identifier format.
12. Configure your group attributes and claims by doing the procedures in the [Configure group claims for SAML applications using SSO configuration](#) section of [Configure group claims for applications with Azure Active Directory \(Public Preview\)](#) in the Microsoft Azure documentation.
13. Under **SAML Signing Certificate**, copy and save the link address for **App Federation Metadata Url** or download **Federation Metadata XML**. You use the Azure AD metadata to configure SAML in the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#).

**3**

| SAML Signing Certificate    |                                                                                                                                                                                                         |
|-----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Status                      | Active                                                                                                                                                                                                  |
| Thumbprint                  | 41800B385144B1426B29D22B17BE35C443D3AA38                                                                                                                                                                |
| Expiration                  | 7/22/2022, 3:16:44 PM                                                                                                                                                                                   |
| Notification Email          |                                                                                                                                                                                                         |
| App Federation Metadata Url | <a href="https://login.microsoftonline.com/c5e8ffb0-eef...">https://login.microsoftonline.com/c5e8ffb0-eef...</a>  |
| Certificate (Base64)        | <a href="#">Download</a>                                                                                                                                                                                |
| Certificate (Raw)           | <a href="#">Download</a>                                                                                                                                                                                |
| Federation Metadata XML     | <a href="#">Download</a>                                                                                                                                                                                |

## Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider

This topic describes how to connect VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) to a SAML identity provider (IdP).

### Overview

User Account and Authentication (UAA), the identity management service for Tanzu Kubernetes Grid Integrated Edition, can authenticate users either through its internal user account store or external authentication mechanisms such as an LDAP server or a SAML IdP.

To connect Tanzu Kubernetes Grid Integrated Edition to a SAML IdP:

1. Configure your SAML IdP. For more information, see [Configure a SAML IdP](#) below.
2. Integrate the UAA server with your SAML IdP by enabling UAA to delegate authentication to your SAML IdP. For more information, see [Integrate UAA with the SAML IdP](#) below. This

enables UAA to delegate authentication to your SAML IdP.

## Configure a SAML IdP

You must configure a SAML IdP to designate Tanzu Kubernetes Grid Integrated Edition as a service provider (SP) before configuring the SAML IdP in the Tanzu Kubernetes Grid Integrated Edition tile.

See the table below for information about industry-standard SAML IdPs and how to integrate them with Tanzu Kubernetes Grid Integrated Edition:

| Solution Name          | Integration Guide                                                              |
|------------------------|--------------------------------------------------------------------------------|
| Okta Single Sign-On    | <a href="#">Configuring Okta as a SAML Identity Provider</a>                   |
| Azure Active Directory | <a href="#">Configuring Azure Active Directory as a SAML Identity Provider</a> |

## Integrate UAA with the SAML IdP

To integrate UAA with your SAML IdP:

1. In the Tanzu Kubernetes Grid Integrated Edition tile, click **UAA**.
2. Under **Configure your UAA user account store with either internal or external authentication mechanisms**, select **SAML Identity Provider**.

Configure your UAA user account store with either internal or external authentication mechanisms \*

- Internal UAA
- LDAP Server
- SAML Identity Provider

Provider Name \*

Default Identity Provider

Enable tkgi cli automatic approval

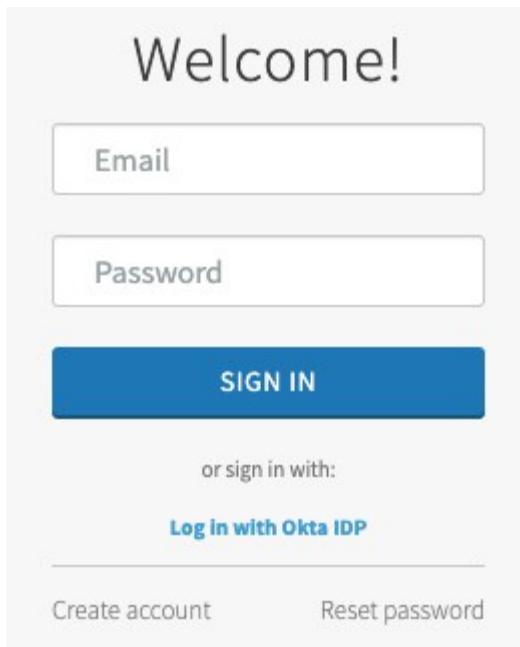
Enable cluster client tkgi cli automatic approval

Display Name \*

Provider Metadata (if you would rather provide an SSO endpoint URL, skip to the next field)

(OR) Provider Metadata URL

3. For **Provider Name**, enter a unique name you create for the IdP. This name can include only alphanumeric characters, +, \_, and -. You must not change this name after deployment because all external users use it to link to the provider.
4. For **Display Name**, enter a display name for your provider. This display name appears as a link on your Ops Manager login page, which you can access at <https://TKGI-API:8443/login>.



or sign in with:

[Log in with Okta IDP](#)

[Create account](#)    [Reset password](#)

5. To directly authenticate users with the configured external identity provider, enable **Default Identity Provider**.
6. To automatically bypass displaying the scope approval screen when logging in to the TKGI CLI, enable **Enable tkgi cli automatic approval**.
7. To automatically bypass displaying the scope approval screen for the `tkgi get-credentials` cli command, enable **Enable cluster client tkgi cli automatic approval**.
8. Retrieve the metadata from your IdP. You recorded your IdP metadata when you configured your IdP to designate Tanzu Kubernetes Grid Integrated Edition as a SP. See [Prerequisites](#) above.
9. Enter your IdP metadata into either the **Provider Metadata** or the **Provider Metadata URL** fields:
  - ◊ If your IdP exposes a metadata URL, enter it in **Provider Metadata URL**.
  - ◊ If your IdP does not expose a metadata URL, paste the XML you retrieved into **Provider Metadata**.

 **Note:** VMware recommends that you use the Provider Metadata URL rather than Provider Metadata because the metadata can change. You need to select only one of the above configurations. If you configure both, your IdP defaults to the **(OR) Provider Metadata URL**.

10. For **Name ID Format**, select the name identifier format for your SAML IdP. This translates to `username` in Tanzu Kubernetes Grid Integrated Edition. The default is `Email Address`.

Name ID Format\*

Email Address

First Name Attribute

Last Name Attribute

Email Attribute

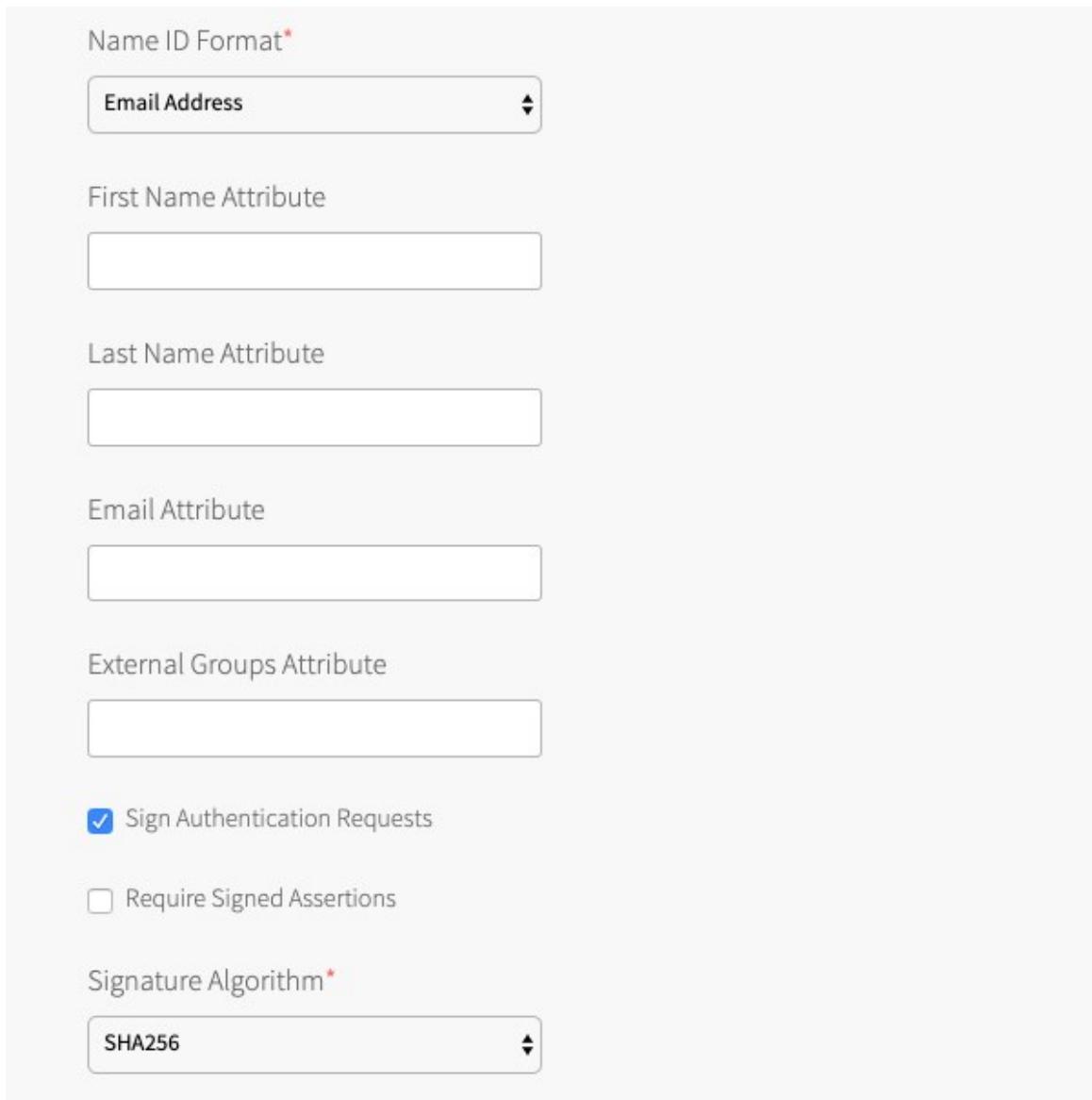
External Groups Attribute

Sign Authentication Requests

Require Signed Assertions

Signature Algorithm\*

SHA256



11. For **First Name Attribute** and **Last Name Attribute**, enter the attribute names in your SAML database that correspond to the first and last names in each user record. This field is case sensitive.
12. For **Email Attribute**, enter the attribute name in your SAML assertion that corresponds to the email address in each user record, for example, `EmailID`. This field is case sensitive.
13. For **External Groups Attribute**, enter the attribute name in your SAML database for your user groups. This field is case sensitive. To map the groups from the SAML assertion to admin roles in TKGI, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to an External SAML Group](#) in *Managing Tanzu Kubernetes Grid Integrated Edition Users with UAA*.
14. By default, all SAML authentication requests from Tanzu Kubernetes Grid Integrated Edition are signed. To change this, deactivate **Sign Authentication Requests** and configure your IdP to verify SAML authentication requests.
15. To validate the signature for the incoming SAML assertions, enable **Required Signed Assertions** and configure your IdP to send signed SAML assertions.
16. For **Signature Algorithm**, choose an algorithm from the dropdown to use for signed requests

and assertions. The default value is [SHA256](#).

17. Click **Save**.

## Complete Your Tile Configuration

- If you do not need to configure any other settings in the Tanzu Kubernetes Grid Integrated Edition tile, return to the Ops Manager Installation Dashboard and click **Review Pending Changes > Apply Changes**.
- If you need to configure any other settings in the Tanzu Kubernetes Grid Integrated Edition tile, return to the *Installing Tanzu Kubernetes Grid Integrated Edition* topic for your IaaS and follow the instructions for the pane you want to configure:
  - [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#)
  - [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T](#)
  - [Installing Tanzu Kubernetes Grid Integrated Edition on GCP](#)
  - [Installing Tanzu Kubernetes Grid Integrated Edition on AWS](#)
  - [Installing Tanzu Kubernetes Grid Integrated Edition on Azure](#)

## Next Steps

For information about creating Tanzu Kubernetes Grid Integrated Edition roles and managing Kubernetes cluster access, see:

- [Setting Up Tanzu Kubernetes Grid Integrated Edition Admin Users](#) for your IaaS
- [Managing Cluster Access and Permissions](#)

## Managing Tanzu Kubernetes Grid Integrated Edition Users with UAA

This topic describes how to manage users in VMware Tanzu Kubernetes Grid Integrated Edition with User Account and Authentication (UAA).

### Overview

UAA is the identity management service for Tanzu Kubernetes Grid Integrated Edition. Tanzu Kubernetes Grid Integrated Edition includes a UAA server, which is hosted on the TKGI API VM.

To interact with the UAA server, you can use the UAA Command Line Interface (UAAC). You can either run UAAC commands from the Ops Manager VM or install UAAC on your local workstation.

### UAA Scopes for Tanzu Kubernetes Grid Integrated Edition Users

By assigning UAA scopes, you grant users the ability to create, manage, and audit Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition.

A UAA admin user can assign the following UAA scopes to Tanzu Kubernetes Grid Integrated Edition users:

- `pks.clusters.admin`: Accounts with this scope can create and access all clusters.
- `pks.clusters.manage`: Accounts with this scope can create and access their own clusters.
- `pks.clusters.admin.read`: Accounts with this scope can access any information about all clusters except for cluster credentials.

You can assign these scopes to individual users, external identity provider groups, or clients for automation purposes.

For more information about UAA scopes in Tanzu Kubernetes Grid Integrated Edition, see [UAU Scopes](#).

## Prerequisites

Before managing users for Tanzu Kubernetes Grid Integrated Edition, you must connect to the TKGI API VM. To connect to the TKGI API VM, you need one of the following:

- SSH access to the Ops Manager VM
- A machine that can connect to your TKGI API VM

For instructions on how to connect to the TKGI control plane, see [Connect to the TKGI API VM](#) for your IaaS.

## Log In as a UAA Admin

Before creating TKGI users, you must log in to the UAA server as a UAA admin. To log in to the UAA server, do the following:

1. Retrieve the UAA management admin client secret:
  1. In a web browser, navigate to the Ops Manager **Installation Dashboard** and click the **Tanzu Kubernetes Grid Integrated Edition** tile.
  2. Click the **Credentials** tab.
  3. Click **Link to Credential** next to **Pks Uaa Management Admin Client** and copy the value of `secret`.
2. Target your UAA server by running the following command:

```
uaac target https://TKGI-API:8443 --ca-cert CERTIFICATE-PATH
```

Where:

- `TKGI-API` is the domain name of your TKGI API server. You entered this domain name in the **Tanzu Kubernetes Grid Integrated Edition** tile > **TKGI API > API Hostname (FQDN)**.
- `CERTIFICATE-PATH` is the path to your Ops Manager root CA certificate. Provide this certificate to validate the TKGI API certificate with SSL.
  - If you are logged in to the Ops Manager VM, specify `/var/tempster/workspaces/default/root_ca_certificate` as the path. This is the default location of the root certificate on the Ops Manager VM.

- If you downloaded the Ops Manager root CA certificate to your machine, specify the path where you stored the certificate.

For example:

```
$ uaac target api.tkgi.example.com:8443 -ca-cert /var/tempest/wo
rkspaces/default/root_ca_certificate
```



**Note:** If you receive an `Unknown key: Max-Age = 86400` warning message, you can ignore it because it has no impact.

3. Authenticate with UAA by running the following command:

```
uaac token client get admin -s ADMIN-CLIENT-SECRET
```

Where `ADMIN-CLIENT-SECRET` is your UAA management admin client secret that you retrieved in a previous step. The client username is `admin`.

## Grant Tanzu Kubernetes Grid Integrated Edition Access to an Individual User

To create a new UAA user with Tanzu Kubernetes Grid Integrated Edition access, do the following:

1. If you are not logged in as the UAA admin, perform the steps in [Log In as a UAA Admin](#).
2. Create a new user by running the following command:

```
uaac user add USERNAME --emails USER-EMAIL -p USER-PASSWORD
```

For example:

```
$ uaac user add cody --emails cody@example.com -p password
```



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must add `--origin SAML-ORIGIN` to the above command. `SAML-ORIGIN` is the domain name for your SAML identity provider. To find `SAML-ORIGIN`, click on the TKGI tile, select **Settings > UAA > SAML**, and locate the `Provider Name`. For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#).

3. Assign a TKGI cluster scope to the new user by running the following command:

```
uaac member add UAA-SCOPE USERNAME
```

Where:

- ◊ **UAA-SCOPE** is one of the UAA scopes described in [UAA Scopes for Tanzu Kubernetes Grid Integrated Edition Users](#).
- ◊ **USERNAME** is the user that you created in the previous step.

For example:

```
$ uaac member add pks.clusters.admin cody
```

After you assign this scope, the user can create and manage Kubernetes clusters. For more information, see [Managing Kubernetes Clusters and Workloads](#).

## Grant Tanzu Kubernetes Grid Integrated Edition Access to an External Group

Connecting Tanzu Kubernetes Grid Integrated Edition to an external LDAP or SAML user store enables the UAA server to delegate authentication to existing enterprise user stores.



**Note:** When integrating UAA with an external identity provider, authentication within UAA becomes chained. UAA first attempts to authenticate with user credentials against the UAA user store before the external identity provider. For more information about integrating LDAP, see [Chained Authentication in the User Account and Authentication LDAP Integration GitHub documentation](#).

For more information about the process used by the UAA server when it attempts to authenticate a user through LDAP, see the [Configuring LDAP Integration with Pivotal Cloud Foundry Knowledge Base article](#).

To grant Tanzu Kubernetes Grid Integrated Edition access to an external identity provider group, do one the following procedures:

- [Grant Tanzu Kubernetes Grid Integrated Edition Access to an External LDAP Group](#)
- [Grant Tanzu Kubernetes Grid Integrated Edition Access to an External SAML Group](#)

## Grant Tanzu Kubernetes Grid Integrated Edition Access to an External LDAP Group

To grant Tanzu Kubernetes Grid Integrated Edition access to an external LDAP group, do the following:

1. If you are not logged in as the UAA admin, do the steps in [Log In as a UAA Admin](#).
2. Assign a TKGI cluster scope to all users in an LDAP group by running the following command:

```
uaac group map --name UAA-SCOPE GROUP-DISTINGUISHED-NAME
```

Where:

- ◊ **UAA-SCOPE** is one of the UAA scopes described in [UAA Scopes for Tanzu Kubernetes Grid Integrated Edition Users](#).

- ◆ `GROUP-DISTINGUISHED-NAME` is the LDAP Distinguished Name (DN) for the group.

For example:

```
$ uaac group map -name pks.clusters.manage cn=operators,ou=groups,dc=example,dc=com
```

For more information about LDAP DNs, see the [LDAP DNs and RDNs](#) in the LDAP documentation.

## Grant Tanzu Kubernetes Grid Integrated Edition Access to an External SAML Group

To grant Tanzu Kubernetes Grid Integrated Edition access to an external SAML group, do the following:

1. If you are not logged in as the UAA admin, do the steps in [Log In as a UAA Admin](#).
2. Assign a TKGI cluster scope to all users in a SAML group by running the following command:

```
uaac group map --name UAA-SCOPE SAML-GROUP --origin SAML-ORIGIN
```

Where:

- ◆ `UAA-SCOPE` is one of the UAA scopes described in [UAA Scopes for Tanzu Kubernetes Grid Integrated Edition Users](#).
- ◆ `SAML-GROUP` is name of your SAML identity provider group.
- ◆ `SAML-ORIGIN` is the domain name for your SAML identity provider. To find `SAML-ORIGIN`, click on the TKGI tile, select **Settings > UAA > SAML**, and locate the `Provider Name`.

For example:

```
$ uaac group map -name pks.clusters.manage tkgi-devs -origin my-sso.example.com
```

## Grant Tanzu Kubernetes Grid Integrated Edition Access to a Client

To grant Tanzu Kubernetes Grid Integrated Edition access to a client for a script or service automation, do the following:

1. If you are not logged in as the UAA admin, perform the steps in [Log In as a UAA Admin](#).
2. Create a client with the desired scopes by running the following command:

```
uaac client add CLIENT-NAME -s CLIENT-SECRET \
--authorized_grant_types client_credentials \
--authorities UAA-SCOPES
```

Where:

- ◆ `CLIENT-NAME` and `CLIENT-SECRET` are the client credentials.

- **UAA-SCOPES** is one or more of the UAA scopes described in [UAA Scopes for Tanzu Kubernetes Grid Integrated Edition Users](#), separated by a comma. For example:

```
$ uaac client add automated-client
-s randomly-generated-secret -authorized_grant_types client_credentials
-authorities pks.clusters.admin,pks.clusters.manage
```

## UAA Scopes for Tanzu Kubernetes Grid Integrated Edition Users

This topic describes User Account and Authentication (UAA) scopes that a UAA admin can assign to VMware Tanzu Kubernetes Grid Integrated Edition users.

### Overview

UAA is the identity management service for Tanzu Kubernetes Grid Integrated Edition. By assigning UAA scopes, you grant users the ability to create, manage, and audit Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition.

A UAA admin user can assign the following UAA scopes to Tanzu Kubernetes Grid Integrated Edition users:

- **pks.clusters.admin**: Accounts with this scope can create and access all clusters.
- **pks.clusters.manage**: Accounts with this scope can create and access their own clusters.
- **pks.clusters.admin.read**: Accounts with this scope can access any information about all clusters except for cluster credentials.

You can assign these scopes to individual users, external identity provider groups, or clients for automation purposes.

### UAA Scopes

Each UAA scope grants Tanzu Kubernetes Grid Integrated Edition users a set of permissions for creating, managing, and auditing Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes clusters. For information about the permissions, see the table below.

| Operation                                    | <b>pks.clusters.<br/>admin</b>                                 | <b>pks.clusters.<br/>manage</b>                                           | <b>pks.clusters.<br/>admin.read</b>                    |
|----------------------------------------------|----------------------------------------------------------------|---------------------------------------------------------------------------|--------------------------------------------------------|
| Create, update, resize, and delete a cluster | <b>Yes.</b> Can create, modify, and delete all clusters.       | <b>Yes.</b> Can create, modify, and delete only their own clusters.       | <b>No.</b> Cannot create, modify, and delete clusters. |
| Get cluster credentials                      | <b>Yes.</b> Can retrieve cluster credentials for all clusters. | <b>Yes.</b> Can retrieve cluster credentials only for their own clusters. | <b>No.</b> Cannot retrieve cluster credentials.        |
| Upgrade clusters                             | <b>Yes.</b> Can upgrade all clusters.                          | <b>Yes.</b> Can upgrade only their own clusters.                          | <b>No.</b> Cannot upgrade clusters.                    |

|                                                     |                                                                     |                                                                                |                                                          |
|-----------------------------------------------------|---------------------------------------------------------------------|--------------------------------------------------------------------------------|----------------------------------------------------------|
| List clusters                                       | <b>Yes.</b> Can list all clusters.                                  | <b>Yes.</b> Can list only their own clusters.                                  | <b>Yes.</b> Can list all clusters.                       |
| View cluster details                                | <b>Yes.</b> Can view cluster details for all clusters.              | <b>Yes.</b> Can view cluster details only for their own clusters.              | <b>Yes.</b> Can view cluster details for all clusters.   |
| Create and delete a compute profile                 | <b>Yes.</b> Can create and delete compute profiles.                 | <b>No.</b> Cannot create and delete compute profiles.                          | <b>No.</b> Cannot create and delete compute profiles.    |
| Create and delete a network profile                 | <b>Yes.</b> Can create and delete network profiles.                 | <b>No.</b> Cannot create and delete network profiles.                          | <b>No.</b> Cannot create and delete network profiles.    |
| Create and delete a Kubernetes profile              | <b>Yes.</b> Can create, modify, and delete all Kubernetes profiles. | <b>Yes.</b> Can create, modify, and delete only their own Kubernetes profiles. | <b>No.</b> Cannot create and delete Kubernetes profiles. |
| Create, update, and delete a quota                  | <b>Yes.</b> Can create, update, and delete quotas.                  | <b>No.</b> Cannot create, update, and delete quotas.                           | <b>No.</b> Cannot create, update, and delete quotas.     |
| List Tanzu Kubernetes Grid Integrated Edition plans | <b>Yes.</b> Can list all available plans.                           | <b>Yes.</b> Can list all available plans.                                      | <b>Yes.</b> Can list all available plans.                |

To assign UAA scopes in Tanzu Kubernetes Grid Integrated Edition, follow the instructions in [Managing Tanzu Kubernetes Grid Integrated Edition Users with UAA](#).

## OIDC Provider for Kubernetes Clusters

This topic describes the global default OpenID Connect (OIDC) provider setting that you can use for Kubernetes clusters in VMware Tanzu Kubernetes Grid Integrated Edition and how to override it for individual clusters.

### Overview

Configuring an OIDC provider for TKGI-provisioned clusters enables Kubernetes to verify end-user identities based on the authentication performed by UAA or a custom OIDC provider.

You can use the following methods to configure an OIDC provider in Tanzu Kubernetes Grid Integrated Edition:

- Configure UAA as the default OIDC provider in the **Tanzu Kubernetes Grid Integrated Edition** tile > **UAA**. For more information, see [UAA as the Default OIDC Provider](#) below.
- Configure a custom OIDC provider by applying a Kubernetes profile to one or more TKGI-provisioned clusters. For more information, see [Custom OIDC Provider](#) below.

### UAA as the Default OIDC Provider

The **Tanzu Kubernetes Grid Integrated Edition** tile > **UAA** > **Configure created clusters to use UAA as the OIDC provider** is a global setting for TKGI-provisioned clusters, described in the table below:

| Option | Description |
|--------|-------------|
|--------|-------------|

---

|                    |                                                                                                                                                                                                                                                                                                                                                                                                          |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Activate</b>    | If you enable UAA as the OIDC provider, Kubernetes verifies end-user identities based on authentication executed by UAA as follows:                                                                                                                                                                                                                                                                      |
|                    | <ul style="list-style-type: none"> <li>• If you select <b>Internal UAA</b>, Kubernetes authenticates users against the internal UAA authentication mechanism.</li> <li>• If you select <b>LDAP Server</b>, Kubernetes authenticates users against the LDAP server.</li> <li>• If you select <b>SAML Identity Provider</b>, Kubernetes authenticates users against the SAML identity provider.</li> </ul> |
| <b>Deactivated</b> | If you do not activate UAA as the OIDC provider, Kubernetes authenticates users against its internal user management system.                                                                                                                                                                                                                                                                             |

---

When you activate UAA as your OIDC provider, existing TKGI-provisioned clusters are upgraded to use OIDC. This invalidates your kubeconfig files. You must regenerate the files for all existing clusters.

## Custom OIDC Provider

You can configure one or more Kubernetes clusters to use a custom OIDC provider by creating and applying a Kubernetes profile to the clusters. This overrides the global **Configure created clusters to use UAA as the OIDC provider** setting in the **Tanzu Kubernetes Grid Integrated Edition** tile > **UAA**.

For instructions, see [Adding an OIDC Provider](#).

## After You Configure Your OIDC Provider

If you want to give Kubernetes end users, such as developers, access to TKGI-provisioned clusters after you configure your OIDC provider, you must create Kubernetes role bindings for them.

For instructions, see [Managing Cluster Access and Permissions](#).

## Managing Kubernetes Cluster Options

This section describes how an Tanzu Kubernetes Grid Integrated Edition administrator can create and manage options for the Kubernetes clusters that Tanzu Kubernetes Grid Integrated Edition users provision.

See the following topics:

- [CPU and Memory Quotas](#)
- [Network Profiles \(NSX-T Only\)](#)
- [Compute Profiles and Host Groups \(vSphere Only\)](#)

## Limit Resource Usage

This section describes how to define and use resource quotas for Kubernetes clusters in VMware Tanzu Kubernetes Grid Integrated Edition.

See the following topics:

- [Assign Resource Quotas to Users in the Management Console](#)

- Managing Resource Usage with Quotas
- Viewing Usage Quotas

## Assign Resource Quotas to Users in the Management Console

This topic describes how to assign VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) resource quotas to users.



**Warning:** This feature is a beta component and is intended for evaluation and test purposes only. Do not use this feature in a production environment. Product support and future availability are not guaranteed for beta components.

### Overview

After you have added users to your TKGI deployment on vSphere and assigned the cluster manager or admin role to them, you can optionally assign resource quotas to them.

By setting quotas, you can limit the amount of compute power and memory that those users can consume. You can also set a limit on the number of clusters that they can deploy.

### Assign a Resource Quota to a User

To assign a resource quota to a user:

1. Go to the **Quotas** view of the management console.

| User Name | Number of vCPUs | Total Memory(G) | Number of Clusters |
|-----------|-----------------|-----------------|--------------------|
|           |                 |                 |                    |

Users per page: 10 | 1 - 10 of 0 Users

[View a larger version of this image](#)

2. Click **Add Quota** and select the users to whom to apply the quota.

## + Add Quota

You can add quotas for users who has cluster management role based on the number of vCPUs, total memory or number of clusters.

| <input type="checkbox"/>            | Name          |
|-------------------------------------|---------------|
| <input checked="" type="checkbox"/> | Fang Shui Lin |
| <input checked="" type="checkbox"/> | Fei Xu Sun    |
| <input type="checkbox"/>            | Han Meimei    |
| <input type="checkbox"/>            | Li Lei        |

2      1 - 4 of 4 Users

Only users who have the cluster manager role are included in the list.

3. Enter maximums for the number of virtual CPUs and the amount of memory that the selected users can consume, and the number of clusters they can create.

|                    |        |                                               |
|--------------------|--------|-----------------------------------------------|
| Number of vCPUs    | 10     | <input type="checkbox"/> Unlimited            |
| Total Memory       | 100 GB | <input type="checkbox"/> Unlimited            |
| Number of Clusters | -1     | <input checked="" type="checkbox"/> Unlimited |

CANCEL
SAVE

4. To grant users unrestricted access to CPU or memory resources or an unlimited number of clusters, enable the **Unlimited** toggle for the unrestricted resource.
5. Click **Save**.

## Modify a User Resource Quota

To update the resources assigned to a user:

1. Select the quota.
2. Click **Edit**.

⊖ Quotas

**ADD QUOTA** **EDIT** **DELETE**

| <input type="checkbox"/>            | User Name     | Number of vCPUs | Total Memory(G) | Number of Clusters                          |
|-------------------------------------|---------------|-----------------|-----------------|---------------------------------------------|
| <input checked="" type="checkbox"/> | Fang Shui Lin | 10              | 100             | Unlimited                                   |
| <input type="checkbox"/>            | Fei Xu Sun    | 10              | 100             | Unlimited                                   |
| <input checked="" type="checkbox"/> | 1             |                 |                 | Users per page <b>10</b>   1 - 2 of 2 Users |

3. Modify the maximums for the number of virtual CPUs and the amount of memory that the selected users can consume, and the number of clusters they can create.
4. To grant users unrestricted access to CPU or memory resources or an unlimited number of clusters, enable the **Unlimited** toggle for the unrestricted resource.
5. Click **Save**.

## Delete a User Resource Quota

To delete a resources quota:

1. Select the quota.
2. Click **Delete**.

⊖ Quotas

**ADD QUOTA** **EDIT** **DELETE**

| <input type="checkbox"/>            | User Name     | Number of vCPUs | Total Memory(G) | Number of Clusters                          |
|-------------------------------------|---------------|-----------------|-----------------|---------------------------------------------|
| <input checked="" type="checkbox"/> | Fang Shui Lin | 10              | 100             | Unlimited                                   |
| <input type="checkbox"/>            | Fei Xu Sun    | 10              | 100             | Unlimited                                   |
| <input checked="" type="checkbox"/> | 1             |                 |                 | Users per page <b>10</b>   1 - 2 of 2 Users |



**Note:** Deleting a quota only deletes the quota. It does not delete the user account.

## Next Steps

- [Working with Network Profiles in the Management Console](#)
- [Create Clusters in the Management Console](#)

## Managing Resource Usage with Quotas



**Warning:** This feature is a beta component and is intended for evaluation and test purposes only. Do not use this feature in a production environment. Product support and future availability are not guaranteed for beta components.

This topic describes how to restrict and review the usage of VMware Tanzu Kubernetes Grid Integrated Edition resources by Tanzu Kubernetes Grid Integrated Edition users.

## Overview

As an Tanzu Kubernetes Grid Integrated Edition administrator, you can set a limit on each user's total resource allocation within Tanzu Kubernetes Grid Integrated Edition.

You manage resources in Tanzu Kubernetes Grid Integrated Edition by defining quotas for individual users with the TKGI API.

The `quotas` API endpoint allows you to restrict the total amount of memory and number of CPUs that a user can allocate in total across their deployed clusters.

In addition, you can limit the total number of clusters a user can provision within Tanzu Kubernetes Grid Integrated Edition.

To review overall resource usage and for individual users, you access the TKGI API `usages` endpoint.



**Note:** Quota settings affect only non-admin user accounts. A quota applied to an admin user account is ignored.

## Set up Your API Access Token

The curl commands in this topic use an access token environment variable to authenticate into the TKGI API.

1. To export your access token into an environment variable, run the following command:

```
tkgi login -a TKGI-API -u USER-ID -p 'PASSWORD' -k; \
export YOUR-ACCESS-TOKEN=$(bosh int ~/.pks/creds.yml --path /access_token)
```

Where:

- ◊ `TKGI-API` is the FQDN of your TKGI API endpoint. For example, `api.tkgi.example.com`.
- ◊ `USER-ID` is your Tanzu Kubernetes Grid Integrated Edition user ID.
- ◊ `PASSWORD` is your Tanzu Kubernetes Grid Integrated Edition password.
- ◊ `YOUR-ACCESS-TOKEN` is the name of your access token environment variable.

For example:

```
$ tkgi login -a tkgi.my.lab -u alana -p 'psswrdaabc123...!' -k;
export my_token=$(bosh int ~/.pks/creds.yml -path /access_token)
```



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

## Manage Quotas

This section describes how to add, modify and delete user quotas.

### Add a Quota

To enforce a quota on a specific user, run the following command:

```
curl -k -X POST \
-H "Authorization: Bearer $YOUR-ACCESS-TOKEN" \
-H "Content-Type: application/json" \
-d \
'{
 "owner": "USER-ID",
 "limit": {
 "cpu": MAX-CPU,
 "memory": MAX-MEM,
 "cluster": MAX-CLUSTER
 }
}' \
https://TKGI-API:9021/v1/quotas
```

Where:

- `YOUR-ACCESS-TOKEN` is your access token environment variable.
- `USER-ID` is the user account ID to enforce the quota restriction on.
- `MAX-CPU` is the maximum total amount of CPU resources that the user can allocate to containers and pods. If `MAX-CPU` is set to `0`, the user cannot create clusters.
- `MAX-MEM` is the maximum total amount of memory, in gigabytes, that the user can allocate to containers and pods. If `MAX-MEM` is set to `0`, the user cannot create clusters.
- `MAX-CLUSTER` is the maximum number of clusters that the user can provision. This value must be greater than or equal to `1`.
- `TKGI-API` is the FQDN of your TKGI API server.

For example:

```
$ user=exampleuser $ tkgi login -a tkgi.my.lab -u $user -p 'psswrda...!' \
-k; export TOKEN=$(bosh int ~/.pks/creds.yml -path /access_token) $ curl -k \
-X POST

-H "Authorization: Bearer $TOKEN"

-H "Content-Type: application/json"

-d

'{ "owner": "cody", "limit": { "cpu": 4, "memory": 5, "cluster": 10 } }'

https://example.com:9021/v1/quotas
```

## Modify an Existing Quota

To modify a specific user's existing quota, run the following command:

```
curl -k -X PATCH \
-H "Authorization: Bearer $YOUR-ACCESS-TOKEN" \
-H "Content-Type: application/json" \
-d \
'{
 "owner": "USER-ID",
 "limit": {
 "cpu": MAX-CPU,
 "memory": MAX-MEM,
 "cluster": MAX-CLUSTER
 }
}' \
https://TKGI-API:9021/v1/quotas/USER-ID
```

Where:

- `YOUR-ACCESS-TOKEN` is your access token environment variable.
- `USER-ID` is the user account ID to enforce the quota restriction on.
- `MAX-CPU` is the maximum total amount of CPU resources that the user can allocate to containers and pods. If `MAX-CPU` is set to `0`, the user cannot create clusters.
- `MAX-MEM` is the maximum total amount of memory, in gigabytes, that the user can allocate to containers and pods. If `MAX-MEM` is set to `0`, the user cannot create clusters.
- `MAX-CLUSTER` is the maximum number of clusters that the user can provision. This value must be greater than or equal to `1`.
- `TKGI-API` is the FQDN of your TKGI API server. For example, `api.tkgi.example.com`.

For example:

```
$ user@exampleuser $ tkgi login -a tkgi.my.lab -u $user -p 'psswrda...!' \
-k; export TOKEN=$(bosh int ~/.pks/creds.yml -path /access_token) $ curl -k \
-X PATCH

-H "Authorization: Bearer $TOKEN"

-H "Content-Type: application/json"

-d

'{
 "owner": "cody",
 "limit": {
 "cpu": 2,
 "memory": 3,
 "cluster": 6
 }
}' \
https://example.com:9021/v1/quotas/$user
```

## Delete a Quota

To delete a specific user's existing quota, run the following command:

```
curl -k -X DELETE -H "Authorization: Bearer $YOUR-ACCESS-TOKEN" \
https://TKGI-API:9021/v1/quotas/USER-ID
```

Where:

- `YOUR-ACCESS-TOKEN` is your access token environment variable.
- `TKGI-API` is the FQDN of your TKGI API server.
- `USER-ID` is the user account ID to enforce the quota restriction on.

For example:

```
$ user=exampleuser $ tkgi login -a tkgi.my.lab -u $user -p 'psswrda...!' \
-k; export TOKEN=$(bosh int ~/.pks/creds.yml -path /access_token) $ curl -k \
-X DELETE -H "Authorization: Bearer $TOKEN"

https://example.com:9021/v1/quotas/$user { "body": "The quota owner named: \"e
xampleuser\" not found." }
```

## View Quotas

The TKGI API `quotas` endpoint reports on resource usage quotas in the JSON format.

### View Quotas for a Single User

To list the resource quota restrictions currently applied to a single user, run the following command:

```
curl -k -H "Authorization: Bearer $YOUR-ACCESS-TOKEN" \
https://TKGI-API:9021/v1/quotas/USER-ID
```

Where:

- `YOUR-ACCESS-TOKEN` is your access token environment variable.
- `TKGI-API` is the FQDN of your TKGI API server.
- `USER-ID` is the user account ID to report on.

For example:

```
$ user=exampleuser $ tkgi login -a tkgi.my.lab -u $user -p 'psswrda...!' \
-k; export TOKEN=$(bosh int ~/.pks/creds.yml -path /access_token) $ curl -k \
-H "Authorization: Bearer $TOKEN"

https://example.com:9021/v1/quotas/$user { "owner": "cody", "limit": { "cpu": 2
, "memory": 1.0, "cluster": 6 } }
```

### View All Quotas

To list all current resource and cluster quota restrictions, run the following command:

```
curl -k -H "Authorization: Bearer $YOUR-ACCESS-TOKEN" \
https://TKGI-API:9021/v1/quotas
```

Where:

- **YOUR-ACCESS-TOKEN** is your access token environment variable.
- **TKGI-API** is the FQDN of your TKGI API server.

For example:

```
$ user=exampleuser $ tkgi login -a tkgi.my.lab -u $user -p 'psswrdaabc123...!' -k; export TOKEN=$(bosh int ~/.pks/creds.yml -path /access_token) $ curl -k -H "Authorization: Bearer $TOKEN"

https://example.com:9021/v1/quotas [{ "owner": "cody", "limit": { "cpu": 2, "memory": 1.0, "cluster": 6 } }]
```

## Error Message When User Exceeds Cluster Quota

If a user has exceeded their set cluster creation quota, then the following error message appears when the user attempts to create a cluster.

```
Error: You do not have enough privileges to perform this action. Please contact the TKGI administrator.
```

## View Usage

The TKGI API **usages** endpoint returns resource usage per user in the JSON format.

### View Resource Usage by User

To list the current resource usage of a single user, run the following command:

```
curl -k -H "Authorization: Bearer $YOUR-ACCESS-TOKEN" https://TKGI-API:9021/v1/usages/USER-ID
```

Where:

- **YOUR-ACCESS-TOKEN** is your access token environment variable.
- **TKGI-API** is the FQDN of your TKGI API server.
- **USER-ID** is the user account ID whose resource utilization you want to view.

### View All Resource Usage

To list the current resource utilization for all users and clusters, run the following command:

```
curl -k -H "Authorization: Bearer $YOUR-ACCESS-TOKEN" \
https://TKGI-API:9021/v1/usages
```

Where:

- **YOUR-ACCESS-TOKEN** is your access token environment variable.
- **TKGI-API** is the FQDN of your TKGI API server.

For example:

```
$ user=exampleuser $ tkgi login -a tkgi.my.lab -u $user -p 'psswrdabc123...!' -k; export TOKEN=$(bosh int ~/.pks/creds.yml --path /access_token) $ curl -k -H "Authorization: Bearer $TOKEN"

https://example.com:9021/v1/usages [{ "owner": "cody", "totals": { "cpu": 20, "memory": 52, "cluster": 2 }, "clusters": [{ "name": "vsp1", "cpu": 12, "memory": 36 }] }
```

## Viewing Usage Quotas



**Warning:** This feature is a beta component and is intended for evaluation and test purposes only. Do not use this feature in a production environment. Product support and future availability are not guaranteed for beta components.

This topic describes how to review your resource usage and quotas in VMware Tanzu Kubernetes Grid Integrated Edition using the Tanzu Kubernetes Grid Integrated Edition API.

### Overview

Your Tanzu Kubernetes Grid Integrated Edition administrator might set a limit on the number of clusters you can provision and the resources, such as amount of memory and number of CPUs, that are allocated in total to any clusters you create and workloads you deploy.

The resource quota limitations are based on the total allocated size of the VM instances you create, not their actual utilization.

By using the TKGI API, you can check the resource and cluster limitations that the administrator has assigned to you as well as review your current usage.

## Set up Your API Access Token

The curl commands in this topic use an access token environment variable to authenticate to the TKGI API endpoints.

- To export your access token into an environment variable, run the following command:

```
tkgi login -a TKGI-API -u USER-ID -p 'PASSWORD' -k; \
export YOUR-ACCESS-TOKEN=$(bosh int ~/.pks/creds.yml --path /access_token)
```

Where:

- ◊ **TKGI-API** is the FQDN of your TKGI API endpoint. For example, `api.tkgi.example.com`.
- ◊ **USER-ID** is your Tanzu Kubernetes Grid Integrated Edition user ID.
- ◊ **PASSWORD** is your Tanzu Kubernetes Grid Integrated Edition password.
- ◊ **YOUR-ACCESS-TOKEN** is the name of your access token environment variable.

For example:

```
$ tkgi login -a tkgi.my.lab -u alana -p 'psswrda...!' -k;
export my_token=$(bosh int ~/.pks/creds.yml -path /access_token)
```



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

## View Quotas

The TKGI API `quotas` endpoint returns your resource usage and cluster quota in the JSON format.

To view your resource and cluster quota, run the following command:

```
curl -k -H "Authorization: Bearer $YOUR-ACCESS-TOKEN" \
https://TKGI-API:9021/v1/quotas/USER-ID
```

Where:

- `YOUR-ACCESS-TOKEN` is your access token environment variable.
- `TKGI-API` is the FQDN of your TKGI API endpoint. For example, `api.tkgi.example.com`.
- `USER-ID` is your Tanzu Kubernetes Grid Integrated Edition user ID.

For example:

```
$ user=exampleuser $ tkgi login -a tkgi.my.lab -u $user -p 'psswrda...!' -k; export TOKEN=$(bosh int ~/.pks/creds.yml -path /access_token) $ curl -k -H "Authorization: Bearer $TOKEN"

https://example.com:9021/v1/quotas/$user { "owner": "cody", "limit": { "cpu": 2, "memory": 1.0, "cluster": 6 } }
```

## View Usage

The TKGI API `usages` endpoint reports your actual resource usage in the JSON format.

To view your current allocated resource usage, run the following command:

```
curl -k -H "Authorization: Bearer $YOUR-ACCESS-TOKEN" \
https://TKGI-API:9021/v1/usages/USER-ID
```

Where:

- `YOUR-ACCESS-TOKEN` is your access token environment variable.
- `TKGI-API` is the FQDN of your TKGI API endpoint. For example, `api.tkgi.example.com`.

- `USER-ID` is your Tanzu Kubernetes Grid Integrated Edition user ID.

For example:

```
$ user@exampleuser $ tkgi login -a tkgi.my.lab -u $user -p 'psswrda...!' -k; export TOKEN=$(bosh int ~/.pks/creds.yml -path /access_token) $ curl -k -H "Authorization: Bearerer $TOKEN"

https://example.com:9021/v1/usages [{ "owner": "cody", "totals": { "cpu": 2, "memory": 1, "cluster": 2 }, }]
```

## Error Message When You Exceed Cluster Quota

If you exceed your set cluster creation quota, then the following error message appears when you attempt to create a cluster.

Error: You do not have enough privileges to perform this action. Please contact the TKGI administrator.

## Network Profiles (VMware NSX Only)

This section describes how to define and use network profiles for VMware Tanzu Kubernetes Grid Integrated Edition clusters deployed on NSX-T with vSphere.

See the following topics:

- [Working with Network Profiles in the Management Console](#)
- [Working with Network Profiles in Ops Manager](#)
- **Network Profile Use Cases**
  - ◊ [Size a Load Balancer](#)
  - ◊ [Customize Pod Networks](#)
  - ◊ [Customize Node Networks](#)
  - ◊ [Customize Floating IP Pools](#)
  - ◊ [Configure Bootstrap NSGroups](#)
  - ◊ [Configure Edge Router Selection](#)
  - ◊ [Specify Nodes DNS Servers](#)
  - ◊ [Configure DNS for Pre-Provisioned IPs](#)
  - ◊ [Configure the TCP Layer 4 Load Balancer](#)
  - ◊ [Configure the HTTP/S Layer 7 Ingress Controller](#)
  - ◊ [Define DFW Section Markers](#)
  - ◊ [Configure NCP Logging](#)
  - ◊ [Shared and Dedicated Tier-1 Router Topologies](#)

# Creating and Managing Network Profiles in the Management Console

You can add, view and remove network profiles using the Tanzu Kubernetes Grid Integrated Edition Management Console on vSphere.

## Using Network Profiles

Network profiles let you customize the NSX-T infrastructure networking and the runtime NCP networking for Kubernetes clusters provisioned by Tanzu Kubernetes Grid Integrated Edition. For example, using a network profile you can change the size of the control plane load balancer, add an additional subnet for nodes, and enable the use of a third party ingress controller. For a complete list of use cases, see [Network Profile Use Cases](#).

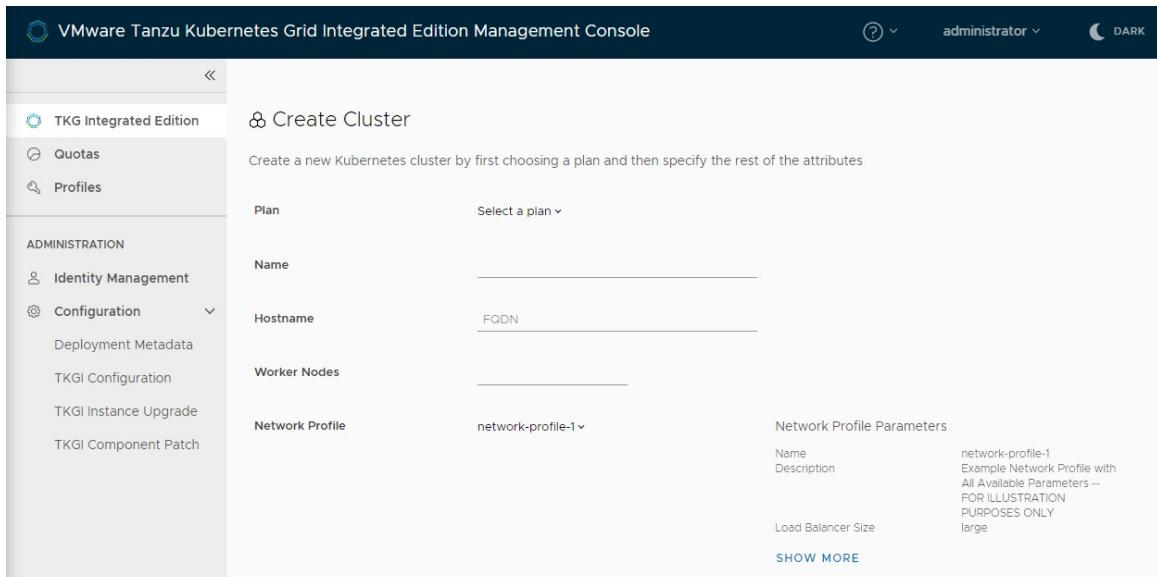
## Requirements for Network Profiles

Network profiles are supported in NSX mode only; there is no support for vSphere without NSX-T mode. In addition, only management console `root` and `pks.clusters.admin` users can create, view, and delete network profiles. Cluster managers can use a network profile when creating a cluster, either using the management console or the TKGI CLI.

## Create Cluster with Network Profile

Use the Tanzu Kubernetes Grid Integrated Edition Management Console to create a cluster with an existing network profile.

1. Select **TKG Integrated Edition > Clusters**, and select **Create Cluster**.
2. Use the **Network Profile** drop-down menu to select the network profile to use.
3. Click **Show More** to view the profile.



[View a larger version of this image](#)

## Define Network Profile

Use the Tanzu Kubernetes Grid Integrated Edition Management Console to define a network profile.



**NOTE:** You must be at the console home page to view the **Network Profiles** tab.

1. Select **Profiles** and select the **Network** tab.
2. Click **Create Profile**.
3. Enter a **Name** for the profile.
4. Enter a suitable **Description** for the profile.
5. Optionally you can set up Parameters for **Advanced Network** or **Container Network**.
6. Configure the new profile as needed, or use the default values.
7. Click **Save**.

The screenshot shows the 'New Network Profile' creation interface. The 'Name' field is set to 'my-network-profile'. The 'Description' field contains the text 'Custom network profile for load balancers'. In the 'Optional Parameters' section, under the 'Advanced Network' tab, the following settings are configured:

- Load Balancer Size: Large
- Pod IP Block IDs: IP Block ip-blk-1
- Pod Subnet Prefix: 16
- Pod Routable: Yes
- Floating Pool IDs: floating-ip-pool
- T0 Router ID: T0 Router 001
- Failover Mode: Preemptive
- Master VMS NSGroup ID: NS Group name 1

[View a larger version of this image](#)

The screenshot shows the 'Container Network' configuration page in the Tanzu Management Console. The left sidebar includes 'TKG Integrated Edition', 'Quotas', and 'Profiles'. Under 'ADMINISTRATION', there are 'Identity Management', 'Configuration' (with sub-options 'Deployment Metadata', 'TKGI Configuration', 'TKGI Instance Upgrade', and 'TKGI Component Patch'), and 'Logs'. The main panel has sections for 'Container Network' settings (e.g., 'Use NSX-T L4 Virtual Server For K8s LoadBalancer' set to 'Yes'), 'LOG SETTINGS' (e.g., 'NCP Log Level' set to 'DEBUG'), and 'INGRESS PERSISTENCE SETTINGS' (e.g., 'Ingress Persistence Type' set to 'source\_ip', 'Persistence Timeout Interval In Seconds' set to '60', and 'Maximum Number Of L4 Virtual Servers Per Cluster' set to '25'). A 'Content Modified' button with 'REVERT' is at the top right.

[View a larger version of this image](#)

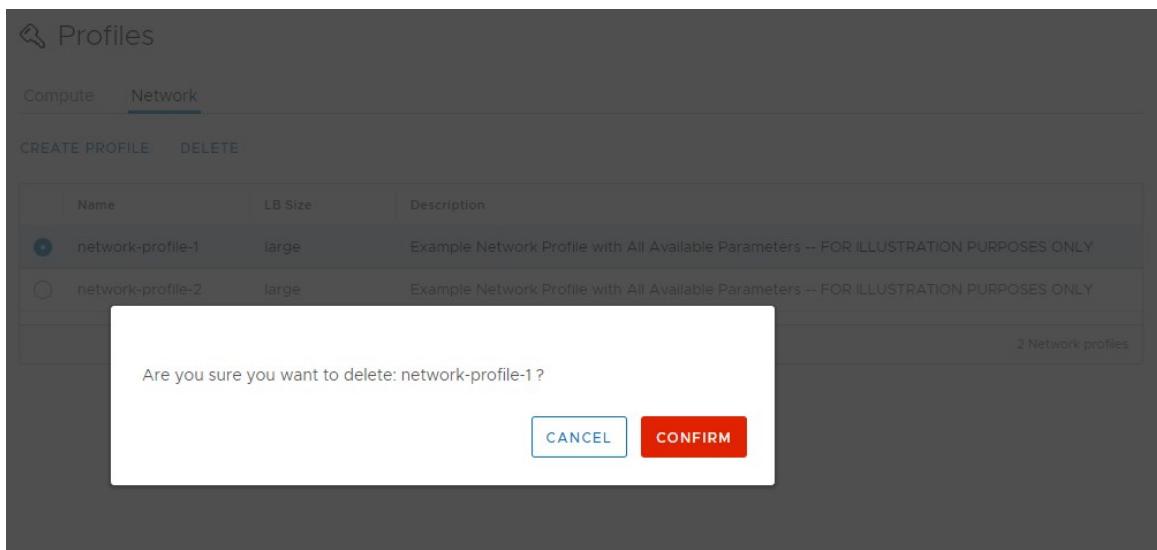
## Delete Network Profile

Use the Tanzu Kubernetes Grid Integrated Edition Management Console to delete network profile.



**NOTE:** You cannot delete a network profile that is in use by a cluster.

1. Select the **Network Profiles** tab.
2. Select the network profile to remove.
3. Click **Delete**.
4. Confirm deletion.



[View a larger version of this image](#)

## Advanced Network Parameters

The table lists and describes the available network profile options for customizing NSX-T.

| Profile Option                 | Description                                                                                                                       |
|--------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Load Balancer Size             | Size of the control plane load balancer: <a href="#">Small</a> , <a href="#">Medium</a> , <a href="#">Large</a> .                 |
| Pod IP Block IDs               | Array of Pod IP Block UUIDs defined in NSX-T.                                                                                     |
| Pod Subnet Prefix              | Size of the Pods IP Block subnet.                                                                                                 |
| Pod Routability                | Make routable the custom Pods subnet: <a href="#">Yes</a> or <a href="#">No</a> .                                                 |
| Floating Pool IDs              | Array of floating IP pool UUIDs defined in NSX-T.                                                                                 |
| T0 Router ID                   | Tenant Tier-0 Router UUID defined in NSX-T.                                                                                       |
| Failover Mode                  | Select <a href="#">Preemptive</a> or <a href="#">Non-preemptive</a> .                                                             |
| Master VMs NSGroup IDs         | Namespace Group UUID as defined in NSX-T.                                                                                         |
| Node IP Block IDs              | Array of Node IP Block UUIDs defined in NSX-T.                                                                                    |
| Node Routable                  | Make routable the custom Node subnet: <a href="#">Yes</a> or <a href="#">No</a> .                                                 |
| Node Subnet Prefix             | Size of the Node IP Block subnet.                                                                                                 |
| Nodes DNS                      | Array of DNS server IP addresses for lookup of Kubernetes nodes and pods.                                                         |
| DNS Lookup Mode                | DNS lookup for the API LB ( <a href="#">API</a> ) and ingress controller ( <a href="#">API_INGRESS</a> ).                         |
| Ingress Prefix                 | Ingress controller hostname prefix for DNS lookup.                                                                                |
| Single Tier Topology           | Use a single Tier-1 Router per cluster: <a href="#">Yes</a> or <a href="#">No</a> .                                               |
| Infrastructure Networks        | Array of IP addresses and subnets for use with a <a href="#">single tier topology</a> in a <a href="#">multi-T0 environment</a> . |
| Custom Infrastructure Networks | Comma-separated array of custom IP addresses or network CIDRs to be used for Infrastructure Networks.                             |

## Container Networks Parameters

The table lists and describes the available network profile options for customizing NCP.

| Profile Option                                                        | Description                                                                                                                                                         |
|-----------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Use NSX-T L4 Virtual Server for K8s Load Balancer                     | Use NSX-T layer 4 virtual server for each Kubernetes service of type LoadBalancer: <code>Yes</code> or <code>No</code> .                                            |
| Use NSX-T L7 Virtual Server as the Ingress Controller for K8s Cluster | Use NSX-T layer 7 virtual server as the ingress controller for the Kubernetes cluster: <code>Yes</code> or <code>No</code> .                                        |
| Use Same Source IP for Calling Clients                                | Use the same source IP for calling clients: <code>Insert</code> or <code>Replace</code> .                                                                           |
| Ingress controller IP address                                         | IP address to use for the ingress controller.                                                                                                                       |
| NCP Log Level                                                         | Configure NCP log levels: <code>INFO</code> , <code>WARNING</code> , <code>DEBUG</code> , <code>ERROR</code> , <code>CRITICAL</code> .                              |
| Log Dropped Firewall Traffic                                          | Log dropped firewall traffic: <code>Yes</code> or <code>No</code> .                                                                                                 |
| Log Firewall Traffic                                                  | Select <code>All</code> , <code>None</code> , or <code>Deny</code> .                                                                                                |
| Ingress Persistence Type                                              | Specify the ingress persistence type: <code>none</code> , <code>cookie</code> , <code>source_ip</code> .                                                            |
| Persistence Timeout Interval in Seconds                               | Persistence timeout interval in seconds.                                                                                                                            |
| Maximum Number of L4 Servers Per Cluster                              | Limit the number of L4 virtual servers per cluster.                                                                                                                 |
| L4 Persistence Type                                                   | Connection stickiness based on <code>source_ip</code> .                                                                                                             |
| L4 Load Balancer Behavior                                             | Customize the layer 4 load balancer behavior: <code>round_robin</code> , <code>least_connection</code> , <code>ip_hash</code> , <code>weighted_round_robin</code> . |
| Top Section-id for Distributed Firewall Section                       | UUID of the top <code>section-id</code> for the distributed firewall (DFW) section as defined in NSX-T.                                                             |
| Bottom Section-id for Distributed Firewall Section                    | UUID of the bottom <code>section-id</code> for the distributed firewall (DFW) section as defined in NSX-T.                                                          |
| Lb Http Request Header Size                                           | The default maximum request header size is 10,240 characters.                                                                                                       |
| Lb Http Response Header Size                                          | The default maximum response header size is 10,240 characters.                                                                                                      |
| Lb Http Response Timeout                                              | Timeout in seconds.                                                                                                                                                 |
| Connect Retry Timeout                                                 | Timeout in seconds.                                                                                                                                                 |

## Working with Network Profiles in Ops Manager

This section describes how to define and use network profiles in Ops Manager, for VMware Tanzu Kubernetes Grid Integrated Edition clusters deployed on NSX-T with vSphere.

See the following topics:

- [Defining Network Profiles](#)
- [Using Network Profiles](#)
- [Size a Load Balancer](#)
- [Customize Pod Networks](#)

- Customize Node Networks
- Customize Floating IP Pools
- Configure Bootstrap NSGroups
- Configure Edge Router Selection
- Specify Nodes DNS Servers
- Configure DNS for Pre-Provisioned IPs
  
- Configure the TCP Layer 4 Load Balancer
- Configure the HTTP/S Layer 7 Ingress Controller
- Define DFW Section Markers
- Configure NCP Logging
- Configure a Shared Tier-1 Router

## Creating and Managing Network Profiles (NSX Only)

This topic describes how VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) administrators can create and delete network profiles for Kubernetes clusters provisioned by TKGI on vSphere with NSX-T integration.

This topic also describes the use cases for when a TKGI administrator should use a network profile.

### Prerequisite

TKGI supports network profiles on TKGI on vSphere with NSX-T only.

To work with TKGI network profiles you must be either a cluster manager or cluster administrator:

- To create or delete a network profile, you must be a cluster administrator:  
`pks.clusters.admin`.
- To use a network profile, you must be a cluster manager: `pks.clusters.manage` or a cluster administrator: `pks.clusters.admin`.



**Note:** If a cluster manager, `pks.clusters.manage`, attempts to create or delete a network profile, the following error occurs: “*You do not have enough privileges to perform this action. Please contact the TKGI administrator.*”

### Overview

You can use network profiles to customize your TKGI Kubernetes clusters on vSphere with NSX-T. For information on when to use network profiles, see [Network Profile Use Cases](#) below.

TKGI cluster administrators can administer network profiles in the following ways:

- Create a Network Profile

- [Update an Existing Network Profile](#)
- [Delete a Network Profile](#)

TKGI cluster administrators can also use network profiles in all the ways that a cluster manager can:

- [List Network Profiles](#)
- [Create a Cluster with a Network Profile](#)
- [Assign a Network Profile to an Existing Cluster](#)
- [Update an Existing Network Profile](#)

For information on managing network profiles, see [Using and Managing Network Profiles](#).

## Create a Network Profile

The following is the basic structure of a network profile JSON configuration:

```
{
 "name": "PROFILE-NAME",
 "description": "PROFILE-DESCRIP",
 "parameters": {
 TOP-LEVEL-PARAMETERS,
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 CNI-CONFIGURATIONS-PARAMETERS
 }
 }
 }
}
```

Where:

- `PROFILE-NAME` is the internal name of the network profile.
- `PROFILE-DESCRIP` is an internal description of the network profile.
- `TOP-LEVEL-PARAMETERS` are one or more comma-delimited top-level `parameters` in a Network Profile. For more information, see [Top-Level Parameters](#) below.
- `CNI-CONFIGURATIONS-PARAMETERS` are one or more comma-delimited `cni_configurations` `parameters` in a Network Profile. For more information, see [cni\\_configurations Parameters](#) below.

To create a network profile in TKGI:

1. Create a network profile configuration JSON file with the following content:

```
{
 "name": "PROFILE-NAME",
 "description": "PROFILE-DESCRIP",
 "parameters": {

 "cni_configurations": {
 "type": "nsxt",
 }
 }
}
```

```
 "parameters": {
 "extensions": {
 "ncp": {
 "nsx_v3": {
 },
 "coe": {
 },
 "ha": {
 },
 "k8s": {
 }
 },
 "nsx-node-agent": {
 }
 }
 }
 }
 }
}
```

Where:

- ◆ `PROFILE-NAME` is the internal name for your network profile.
  - ◆ `PROFILE-DESCRIP` is an internal description for your network profile.
  2. Edit the file to specify your network parameters. For information about the available network parameters, see [Network Profile Parameters](#) below.
  3. Review your network profile configuration carefully. If you are modifying an existing cluster, ensure that you are modifying only parameters that support modification. For information on which network profile parameters are updateable in this version of TKGI, see the network profile parameters tagged `Updatable` in the [Network Profile Parameters](#) tables below. You cannot modify any other network profile parameters on an existing cluster.
  4. To create a network profile from your network profile configuration, run the following TKGI CLI command:

```
tkai create-network-profile PATH-TO-YOUR-NETWORK-PROFILE-CONFIGURATION
```

Where `PATH-TO-YOUR-NETWORK-PROFILE-CONFIGURATION` is the path to your network profile configuration file.

For example:

```
$ tkgi create-network-profile np-routable-pods.json
```

Network profile example-network-profile successfully created

5. Store a copy of your network profile configuration in case you need to modify the network profile in the future.

Cluster managers can create new clusters with your network profile and assign your network profile to existing clusters. For information on managing network profiles, see [Using and Managing Network](#)

[Profiles.](#)

## Network Profile Example

The following is an example of a complete network profile JSON configuration:

```
{
 "name": "example-network-profile",
 "description": "Example Network Profile with All Available Parameters -- FOR ILLUSTRATION PURPOSES ONLY",
 "parameters": {
 "lb_size": "large",
 "pod_ip_block_ids": [
 "ebe78a74-a5d5-4dde-ba76-9cf4067eee55",
 "ebe78a74-a5d5-4dde-ba76-9cf4067eee56"],
 "pod_subnet_prefix": 27,
 "pod_routable": true,
 "fip_pool_ids": [
 "e50e8f6e-1a7a-45dc-ad49-3a607baa7fa0",
 "ebe78a74-a5d5-4dde-ba76-9cf4067eee55"],
 "t0_router_id": "5a7a82b2-37e2-4d73-9cb1-97a8329e1a90",
 "master_vms_nsgroup_id": "9b8d535a-d3b6-4735-9fd0-56305c4a5293",
 "node_ip_block_ids": [
 "2250dc43-63c8-4bb8-b8cf-c6e12ccfb7de", "3d577e5c-dcaf-4921-9458-d12b0e1318e6"],
 "node_routable": true,
 "node_subnet_prefix": 20,
 "nodes_dns": [
 "8.8.8.8", "192.168.115.1", "192.168.116.1"],
 "dns_lookup_mode": "API_INGRESS",
 "ingress_prefix": "ingress",
 "single_tier_topology": true,
 "infrastructure_networks": [
 "30.0.0.0/24",
 "192.168.111.0/24",
 "192.168.115.1"],
 "failover_mode": "PREEMPTIVE",
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "nsx_lb": false,
 "x_forwarded_for": "insert",
 "ingress_ip": "192.168.160.212",
 "log_settings": {
 "log_level": "DEBUG",
 "log_firewall_traffic": "ALL" },
 "ingress_persistence_settings": {
 "persistence_type": "cookie",
 "persistence_timeout": 1 },
 "max_14_lb_service": 10,
 "l4_persistence_type": "source_ip",
 "l4_lb_algorithm": "weighted_round_robin",
 "top_firewall_section_marker": "section-id",
 "bottom_firewall_section_marker": "section-id",
 "lb_http_request_header_size": 60,
 "lb_http_response_header_size": 45,
 "lb_http_response_timeout": 30,
 "connect_retry_timeout": 30,
 "enable_hostport": true,
 "log_level": "INFO" }
 }
 }
 }
}
```

```
 "enable_nodelocaldns": true,
 "client_ssl_profile": "example_ssl_profile_ID",
 "lb_connection_multiplexing_enabled": true,
 "lb_connection_multiplexing_number": 80,
 "extensions": {
 "ncp": {
 "nsx_v3": {
 "retries": "10"
 },
 "coe": {
 "profiling": "False"
 },
 "ha": {
 "heartbeat_period": 6
 },
 "k8s": {
 "lb_ip_allocation": "relaxed"
 }
 },
 "nsx-node-agent": {
 "connect_retry_timeout": 60
 }
 }
 }
}
```



**Note:** This example network profile is for illustration purposes only. It is not intended to be used as a template for a network profile configuration.

# Update an Existing Network Profile

To update an existing cluster's network profile:

1. Confirm the Network Profile Property Supports Updates
  2. Create a Modified Network Profile Configuration
  3. Create a Modified Network Profile
  4. Update the Cluster With a Modified Network Profile

## Confirm the Network Profile Property Supports Updates

After you create a cluster, you can modify only specific network profile parameters. Ensure that you modify only parameters that support modification.

For information on which network profile parameters are updateable in this version of TKGI, see the network profile parameters tagged [Updatable](#) in the [Network Profile Parameters](#) tables below. You cannot modify any other network profile parameters on an existing cluster.

For more information, see [Update-Cluster Network Profile Validation Rules](#) below.

## Create a Modified Network Profile Configuration

To create a modified network profile configuration file:

1. Make a copy of your original network profile configuration file.  
If it is not possible to obtain the original network profile, create a new network profile with the original values in all of the fields.
2. Change the `name` field to a unique name.
3. If you are updating the `pod_ip_block_ids` field, reorder the IP Block IDs or add additional IP Block IDs.

For example, the following network profile has two `pod_ip_block_ids`, the first is the original IP block used when creating the cluster, and the second is the new IP block to use for pods.

```
{
 "description": "Example network profile for adding pod IP addresses to an existing cluster",
 "name": "pod-ips-add",
 "parameters": {
 "pod_ip_block_ids": [
 "ebe78a74-a5d5-4dde-ba76-9cf4067eee55",
 "ebe78a74-a5d5-4dde-ba76-9cf4067eee56"
]
 }
}
```



**Note:** Update only network profile properties that support being updated.

For more information on configuring a network profile, see [Network Profile Parameters](#) above.

4. Review and save the network profile configuration file.
5. Store a copy of your network profile configuration in case you need to modify the network profile in the future.

## Create a Modified Network Profile

To create a network profile from a configuration file:

1. Run the following TKGI CLI command:

```
tkgi create-network-profile PATH-TO-YOUR-NETWORK-PROFILE-CONFIGURATION
```

Where `PATH-TO-YOUR-NETWORK-PROFILE-CONFIGURATION` is the path to your network profile configuration file.

## Update the Cluster with a Modified Network Profile

To update a cluster with a modified network profile:

1. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.

- To apply the network profile created above to your cluster, run the following command:

```
tkgi update-cluster CLUSTER-NAME --network-profile NETWORK-PROFILE-NAME
```

Where:

- `CLUSTER-NAME` is the unique name of your cluster.
- `NETWORK-PROFILE-NAME` is the name of the network profile you want to use for your cluster.

TKGI validates the network profile before updating the cluster with the new network profile. For more information, see [Update-Cluster Network Profile Validation Rules](#) below.

## Update-Cluster Network Profile Validation Rules

TKGI uses strict validation rules before applying a network profile to a cluster with an existing network profile:

- If a field in the original network profile is empty, the system ignores the empty field even if the field is included in the new network profile.
- If a field in the new network profile is empty, the system ignores the field even if the field is not empty in the original network profile.
- If the `pod_ip_block_ids` field in the new network profile contains the same entries as the existing network profile, the entry passes validation.
- If a field in the new network profile conflicts with the field in the existing network profile, the system reports the conflict and fails the validation.

## Delete a Network Profile

TKGI administrators can delete a network profile that is not in use.

To delete a network profile:

- Run the following TKGI CLI command:

```
tkgi delete-network-profile NETWORK-PROFILE-NAME
```

Where `NETWORK-PROFILE-NAME` is the name of the network profile you want to delete.



**Note:** You cannot delete a network profile that is in use.

## Network Profile Parameters

The [Top-Level Parameters](#) and [cni\\_configurations Parameters](#) sections below describe the parameters you can add to a Network Profile.

### Top-Level Parameters

TKGI supports the following top-level network profile parameters:

| Parameter                            | Type   | Description                                                                                                                                                                                                    |
|--------------------------------------|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>name</code>                    | String | User-defined name of the network profile.                                                                                                                                                                      |
| <code>description</code>             | String | User-defined description for the network profile.                                                                                                                                                              |
| <code>parameters</code>              | Map    | Map containing one or more name-value pairs.                                                                                                                                                                   |
| <code>cni_configurations</code>      | Map    | Map containing <code>type</code> and <code>parameters</code> key-value pairs for configuring NCP (see table below).                                                                                            |
| <code>dns_lookup_mode</code>         | String | DNS lookup mode.<br>Values: <code>"API"</code> , <code>"API_INGRESS"</code> .<br>For Kubernetes API LB: <code>"API"</code> .<br>For Ingress controller: <code>"API_INGRESS"</code> .                           |
| <code>failover_mode</code>           | String | If the preferred node fails and recovers, enable the node to preempt a peer as the active node.<br>Values: <code>"PREEMPTIVE"</code> , <code>"NON_PREEMPTIVE"</code> .<br>Default: <code>"PREEMPTIVE"</code> . |
| <code>fip_pool_ids</code>            | String | Array of floating IP pool UUIDs defined in NSX-T.                                                                                                                                                              |
| <code>infrastructure_networks</code> | String | Array of IP addresses and subnets for Node Networks for use with a Shared Tier-1 topology in a Multi-Tier-0 environment.                                                                                       |

|                                    |                                                                                                                                                                                                                                                                                                                                          |
|------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>ingress_prefix</code>        | St Ingress controller hostname prefix for DNS lookup. If DNS mode is set to <code>API_INGRESS</code> , TKGI creates the cluster with <code>ingress_prefix.hostname</code> as the Kubernetes control plane FQDN. TKGI confirms that the ingress subdomain can be resolved as a subdomain prefix on the host before creating new clusters. |
| <code>lb_size</code>               | St Size of the NSX-T load balancer service.<br>ri Values: <code>"small"</code> , <code>"medium"</code> , <code>"large"</code> .<br>n Default: <code>"small"</code> .<br>g                                                                                                                                                                |
| <code>master_vms_nsgroup_id</code> | St Namespace Group UUID as defined in NSX-T.<br>ri<br>n<br>g                                                                                                                                                                                                                                                                             |
| <code>nodes_dns</code>             | St Array (up to 3) of DNS server IP addresses for lookup of Kubernetes nodes and pods.<br>n<br>g<br>U<br>p<br>d<br>at<br>a<br>bl<br>e                                                                                                                                                                                                    |
| <code>pod_ip_block_ids</code>      | St Array of Pod IP Block UUIDs.<br>ri<br>n<br>g<br>U<br>p<br>d<br>at<br>a<br>bl<br>e                                                                                                                                                                                                                                                     |
| <code>pod_routable</code>          | B Make the Pods subnet routable.<br>o Values: <code>true</code> , <code>false</code> .<br>ol Default: <code>false</code> .<br>e<br>a<br>n                                                                                                                                                                                                |
| <code>pod_subnet_prefix</code>     | In Size of the Pods IP Block subnet.<br>te<br>g<br>er                                                                                                                                                                                                                                                                                    |
| <code>single_tier_topology</code>  | B Use a single Tier-1 Router per cluster (shared).<br>o Values: <code>true</code> , <code>false</code> .<br>ol Default: <code>true</code> .<br>e<br>a<br>n                                                                                                                                                                               |

---

|                           |                    |                                             |
|---------------------------|--------------------|---------------------------------------------|
| <code>t0_router_id</code> | St<br>ri<br>n<br>g | Tenant Tier-0 Router UUID defined in NSX-T. |
|---------------------------|--------------------|---------------------------------------------|

---



**Note:** On an existing cluster, you can modify only the network profile parameters that are labeled **Updatable**.

## cni\_configurations Parameters

TKGI supports the following `cni_configurations` parameters:

| Parameter                                   | Type                                                | Description                                                                                                                                                                                                                |
|---------------------------------------------|-----------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>type</code>                           | C<br>o<br>ns<br>ta<br>nt<br>St<br>ri<br>n<br>g      | Values: “ <code>nsxt</code> ”.                                                                                                                                                                                             |
| <code>parameters</code>                     | M<br>ap                                             | Map containing one or more key-value pairs for NCP settings.                                                                                                                                                               |
| <code>bottom_firewall_section_marker</code> | St<br>ri<br>n<br>g<br>U<br>p<br>da<br>ta<br>bl<br>e | UUID of the bottom <code>section-id</code> for the distributed firewall (DFW) section as defined in NSX-T.<br>See also: <a href="#">top_firewall_section_marker</a> below and <a href="#">Define DFW Section Markers</a> . |
| <code>client_ssl_profile</code>             | St<br>ri<br>n<br>g<br>U<br>p<br>da<br>ta<br>bl<br>e | The NSX-T client-side ssl profile to use, exposed by NCP as <code>client_ssl_profile</code> .<br>Default: The default NCP client SSL profile. For more information, see <a href="#">client_ssl_profile</a> below.          |

---

|                                    |    |                                                                                        |
|------------------------------------|----|----------------------------------------------------------------------------------------|
| <code>connect_retry_timeout</code> | In | Configure HTTP LoadBalancer connection retry timeout.                                  |
|                                    | te | Example Value: <code>30</code> .                                                       |
|                                    | ge | See also: <code>lb_http_response_timeout</code> and <code>persistence_timeout</code> . |
|                                    | r  |                                                                                        |
|                                    | U  |                                                                                        |
|                                    | p  |                                                                                        |
|                                    | da |                                                                                        |
|                                    | ta |                                                                                        |
|                                    | bl |                                                                                        |
|                                    | e  |                                                                                        |
| <code>enable_hostport</code>       | B  | Enable NCP support for Kubernetes Host Port.                                           |
|                                    | o  | Values: <code>true</code> , <code>false</code> .                                       |
|                                    | ol | Default: <code>false</code> .                                                          |
|                                    | ea |                                                                                        |
|                                    | n  |                                                                                        |
|                                    | U  |                                                                                        |
|                                    | p  |                                                                                        |
|                                    | da |                                                                                        |
|                                    | ta |                                                                                        |
|                                    | bl |                                                                                        |
|                                    | e  |                                                                                        |
| <code>enable_nodelocaldns</code>   | B  | Enable NCP support for Kubernetes NodeLocal DNSCache.                                  |
|                                    | o  | Values: <code>true</code> , <code>false</code> .                                       |
|                                    | ol | Default: <code>false</code> .                                                          |
|                                    | ea |                                                                                        |
|                                    | n  |                                                                                        |
|                                    | U  |                                                                                        |
|                                    | p  |                                                                                        |
|                                    | da |                                                                                        |
|                                    | ta |                                                                                        |
|                                    | bl |                                                                                        |
|                                    | e  |                                                                                        |
| <code>extensions</code>            | St | Additional NCP and NSX Node Agent settings not included as explicit                    |
|                                    | ri | <code>cni_configurations</code> parameters.                                            |
|                                    | n  | For more information see <a href="#">extensions</a> below.                             |
|                                    | g  |                                                                                        |
|                                    | U  |                                                                                        |
|                                    | p  |                                                                                        |
|                                    | da |                                                                                        |
|                                    | ta |                                                                                        |
|                                    | bl |                                                                                        |
|                                    | e  |                                                                                        |
| <code>ingress_ip</code>            | St | IP address to use for the ingress controller load balancer.                            |
|                                    | ri |                                                                                        |
|                                    | n  |                                                                                        |
|                                    | g  |                                                                                        |

---

---

|                                                 |    |                                                                                 |
|-------------------------------------------------|----|---------------------------------------------------------------------------------|
| <code>ingress_persistence_settings</code>       | St | Map containing one or more key-value pairs for customizing Layer 7 persistence. |
|                                                 | ri | See also: <code>persistence_timeout</code> and <code>persistence_type</code> .  |
|                                                 | g  |                                                                                 |
|                                                 | U  |                                                                                 |
|                                                 | p  |                                                                                 |
|                                                 | da |                                                                                 |
|                                                 | ta |                                                                                 |
|                                                 | bl |                                                                                 |
|                                                 | e  |                                                                                 |
| <code>l4_lb_algorithm</code>                    | St | Layer 4 load balancer behavior.                                                 |
|                                                 | ri | Values: "round_robin", "least_connection", "ip_hash", "weighted_round_robin".   |
|                                                 | n  |                                                                                 |
|                                                 | g  | Default: "round_robin".                                                         |
|                                                 | U  | See also: <code>l4_persistence_type</code> and <code>max_l4_lb_service</code> . |
|                                                 | p  |                                                                                 |
|                                                 | da |                                                                                 |
|                                                 | ta |                                                                                 |
|                                                 | bl |                                                                                 |
|                                                 | e  |                                                                                 |
| <code>l4_persistence_type</code>                | St | Connection stickiness based on <code>source_ip</code> .                         |
|                                                 | ri | Values: "source_ip".                                                            |
|                                                 | n  | See also: <code>l4_lb_algorithm</code> and <code>max_l4_lb_service</code> .     |
|                                                 | g  |                                                                                 |
|                                                 | U  |                                                                                 |
|                                                 | p  |                                                                                 |
|                                                 | da |                                                                                 |
|                                                 | ta |                                                                                 |
|                                                 | bl |                                                                                 |
|                                                 | e  |                                                                                 |
| <code>lb_connection_multiplexing_enabled</code> | B  | Enable NSX-T load balancer TCP multiplexing.                                    |
|                                                 | o  | Values: <code>true</code> , <code>false</code> .                                |
|                                                 | ol | Default: <code>false</code> .                                                   |
|                                                 | ea |                                                                                 |
|                                                 | n  |                                                                                 |
|                                                 | U  |                                                                                 |
|                                                 | p  |                                                                                 |
|                                                 | da |                                                                                 |
|                                                 | ta |                                                                                 |
|                                                 | bl |                                                                                 |
|                                                 | e  |                                                                                 |
| <code>lb_connection_multiplexing_number</code>  | In | The maximum number of NSX-T load balancer TCP multiplexing connections.         |
|                                                 | te |                                                                                 |
|                                                 | ge | Default: 6.                                                                     |
|                                                 | r  |                                                                                 |
|                                                 | U  |                                                                                 |
|                                                 | p  |                                                                                 |
|                                                 | da |                                                                                 |
|                                                 | ta |                                                                                 |
|                                                 | bl |                                                                                 |
|                                                 | e  |                                                                                 |

---

---

|                                           |                                                             |                                                                                                                                                                                                                                                                                                                                         |
|-------------------------------------------|-------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>lb_http_request_header_size</code>  | In<br>te<br>ge<br>r<br><b>U</b><br>p<br>da<br>ta<br>bl<br>e | Configure HTTP LoadBalancer request header size.<br>Example Value: <code>60</code> .                                                                                                                                                                                                                                                    |
| <code>lb_http_response_header_size</code> | In<br>te<br>ge<br>r<br><b>U</b><br>p<br>da<br>ta<br>bl<br>e | Configure HTTP LoadBalancer response header size.<br>Example Value: <code>45</code> .                                                                                                                                                                                                                                                   |
| <code>lb_http_response_timeout</code>     | In<br>te<br>ge<br>r<br><b>U</b><br>p<br>da<br>ta<br>bl<br>e | Configure HTTP LoadBalancer response timeout.<br>Example Value: <code>30</code> .<br>See also: <code>connect_retry_timeout</code> and <code>persistence_timeout</code> .                                                                                                                                                                |
| <code>log_dropped_traffic</code>          | B<br>o<br>ol<br>ea<br>n<br>U<br>p<br>da<br>ta<br>bl<br>e    | <b>Deprecated.</b><br>Use <code>log_firewall_traffic</code> instead.<br>Log dropped firewall traffic.<br>Values: <code>true</code> , <code>false</code> .<br>Default: <code>false</code> .<br>A <code>log_settings</code> parameter. See also: <code>log_firewall_traffic</code> , <code>log_level</code> , <code>log_settings</code> . |
| <code>log_firewall_traffic</code>         | St<br>ri<br>n<br>g<br>U<br>p<br>da<br>ta<br>bl<br>e         | Log firewall traffic.<br>Values: <code>ALL</code> , <code>ALLOW</code> , <code>DENY</code> .<br>Default: <code>DENY</code> .<br>A <code>log_settings</code> parameter. See also: <code>log_level</code> , <code>log_settings</code> .                                                                                                   |

---

---

|                                     |                                                                                                                                                                                                                                                                                                                                |
|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>log_level</code>              | <p>St Values: <code>"INFO"</code>, <code>"WARNING"</code>, <code>"DEBUG"</code>, <code>"ERROR"</code>, <code>"CRITICAL"</code>.</p> <p>ri A <code>log_settings</code> parameter. See also: <code>log_firewall_traffic</code>, <code>log_settings</code>.</p> <p>g</p> <p>U</p> <p>p</p> <p>da</p> <p>ta</p> <p>bl</p> <p>e</p> |
| <code>log_settings</code>           | <p>M Parameters for configuring NCP logging.</p> <p>ap See also: <code>log_dropped_traffic</code>, <code>log_firewall_traffic</code>, <code>log_level</code>.</p> <p>U</p> <p>p</p> <p>da</p> <p>ta</p> <p>bl</p> <p>e</p>                                                                                                     |
| <code>max_14_lb_service</code>      | <p>In Limit the maximum number of layer 4 virtual servers per cluster.</p> <p>te Minimum Value:<code>1</code>.</p> <p>ge See also: <code>14_lb_algorithm</code> and <code>14_persistence_type</code>.</p> <p>r</p> <p>U</p> <p>p</p> <p>da</p> <p>ta</p> <p>bl</p> <p>e</p>                                                    |
| <code>nsx_ingress_controller</code> | <p>B <b>Deprecated</b>.</p> <p>o Use NSX-T layer 7 virtual server as the ingress controller for the</p> <p>ol Kubernetes cluster.</p> <p>ea Values: <code>true</code>, <code>false</code>.</p> <p>n Default: <code>true</code>.</p>                                                                                            |
| <code>nsx_lb</code>                 | <p>B Use NSX-T layer 4 virtual server for each Kubernetes service of type</p> <p>o LoadBalancer.</p> <p>ol Values: <code>true</code>, <code>false</code>.</p> <p>ea Default: <code>true</code>.</p> <p>n</p> <p>U</p> <p>p</p> <p>da</p> <p>ta</p> <p>bl</p> <p>e</p>                                                          |
| <code>persistence_timeout</code>    | <p>In An <code>ingress_persistence_settings</code> parameter. Persistence timeout</p> <p>te interval in seconds.</p> <p>ge See also: <code>connect_retry_timeout</code> and <code>lb_http_response_timeout</code>.</p> <p>r</p> <p>U</p> <p>p</p> <p>da</p> <p>ta</p> <p>bl</p> <p>e</p>                                       |

---

|                                          |                                                                                                                                                                                                                                                                                                                                                                                                                        |
|------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>persistence_type</code>            | St An <code>ingress_persistence_settings</code> parameter. Specify the ingress persistence type.<br>ri<br>n Values: <code>"cookie"</code> , <code>"source_ip"</code> .<br>g Default: <code>"none"</code> .<br><b>U</b><br><b>p</b><br><b>da</b><br><b>ta</b><br><b>bl</b><br><b>e</b>                                                                                                                                  |
| <code>top_firewall_section_marker</code> | St UUID of the top <code>section-id</code> for the distributed firewall (DFW) section as defined in NSX-T.<br>ri<br>n See also: <code>bottom_firewall_section_marker</code> above and <a href="#">Define DFW Section Markers</a> .<br>g<br><b>U</b><br><b>p</b><br><b>da</b><br><b>ta</b><br><b>bl</b><br><b>e</b>                                                                                                     |
| <code>x_forwarded_for</code>             | St Sets the original client source IP in the request header. Enabling the network profile <code>x_forwarded_for</code> parameter automatically enables the <code>x_forwarded_port</code> and <code>x_forward_protocol</code> parameters.<br>ri<br>n<br>g Accepts <code>"insert"</code> and <code>"replace"</code> .<br>U Default: <code>"none"</code> .<br><b>p</b><br><b>da</b><br><b>ta</b><br><b>bl</b><br><b>e</b> |

---



**Note:** On an existing cluster, you can modify only the network profile parameters that are labeled **Updatable**.

## cni\_configurations Extensions Parameters

Configure the less commonly configured NCP and NSX Node Agent settings in the network profile CNI configuration `extensions` field.

Use the network profiles `extensions` field to configure an NCP config map or NSX Node Agent config map property that is applicable to TKGI but is not explicitly supported as a `cni_configurations` parameter.

NCP and NSX Node Agent settings supported as explicit Network Profiles parameters cannot be configured through extensions.

For more information about any of these parameters, see [The nsx-ncp-config ConfigMap](#) and [The nsx-node-agent-config ConfigMap](#) in the VMware NSX-T Data Center documentation.

| Parameter  | T  | Description                                                                              |
|------------|----|------------------------------------------------------------------------------------------|
| ncp.k8s.en | B  | <b>Policy Only</b>                                                                       |
| able_name  | o  | Allow user to set ncp/subnets annotation on namespace to specify the subnets for no-snat |
| space_sub  | o  | namespace.                                                                               |
| nets       | le | Values: <code>TRUE</code> , <code>FALSE</code> .                                         |
|            | a  | Default: <code>FALSE</code> .                                                            |
| n          |    |                                                                                          |
| U          |    |                                                                                          |
| p          |    |                                                                                          |
| d          |    |                                                                                          |
| a          |    |                                                                                          |
| t          |    |                                                                                          |
| a          |    |                                                                                          |
| b          |    |                                                                                          |
| le         |    |                                                                                          |
| ncp.k8s.ht | I  | Port for HTTP ingress.                                                                   |
| tp_ingress | n  | Default: <code>80</code> .                                                               |
| _port      | t  |                                                                                          |
| e          |    |                                                                                          |
| g          |    |                                                                                          |
| e          |    |                                                                                          |
| r          |    |                                                                                          |
| U          |    |                                                                                          |
| p          |    |                                                                                          |
| d          |    |                                                                                          |
| a          |    |                                                                                          |
| t          |    |                                                                                          |
| a          |    |                                                                                          |
| b          |    |                                                                                          |
| le         |    |                                                                                          |
| ncp.k8s.ht | I  | Port for HTTPS ingress.                                                                  |
| tps_ingres | n  | Default: <code>443</code> .                                                              |
| s_port     | t  |                                                                                          |
| e          |    |                                                                                          |
| g          |    |                                                                                          |
| e          |    |                                                                                          |
| r          |    |                                                                                          |
| U          |    |                                                                                          |
| p          |    |                                                                                          |
| d          |    |                                                                                          |
| a          |    |                                                                                          |
| t          |    |                                                                                          |
| a          |    |                                                                                          |
| b          |    |                                                                                          |
| le         |    |                                                                                          |

---

|              |    |                                                                                         |
|--------------|----|-----------------------------------------------------------------------------------------|
| ncp.k8s.la   | S  | List of regex expressions defining the labels that should not be converted to NSX tags. |
| bel_filterin | tr | Default: [ ].                                                                           |
| g_regex_li   | i  |                                                                                         |
| st           | n  |                                                                                         |
|              | g  |                                                                                         |
|              | Li |                                                                                         |
|              | st |                                                                                         |
|              | U  |                                                                                         |
|              | p  |                                                                                         |
|              | d  |                                                                                         |
|              | a  |                                                                                         |
|              | t  |                                                                                         |
|              | a  |                                                                                         |
|              | b  |                                                                                         |
|              | le |                                                                                         |

---

|            |    |                                                                                                     |
|------------|----|-----------------------------------------------------------------------------------------------------|
| ncp.k8s.lb | S  | Allow a virtual IP that is not in the range of external_ip_pools_lb specified in Kubernetes service |
| _ip_alloca | tr | spec.loadBalancerIP.                                                                                |
| tion       | i  | Values: <code>relaxed</code> , <code>strict</code> .                                                |
|            | n  | Default: <code>relaxed</code> .                                                                     |
|            | g  |                                                                                                     |
|            | U  |                                                                                                     |
|            | p  |                                                                                                     |
|            | d  |                                                                                                     |
|            | a  |                                                                                                     |
|            | t  |                                                                                                     |
|            | a  |                                                                                                     |
|            | b  |                                                                                                     |
|            | le |                                                                                                     |

---

|             |    |                                                                      |
|-------------|----|----------------------------------------------------------------------|
| ncp.k8s.st  | B  | <b>Policy Only</b>                                                   |
| atefulset_i | o  | Locate Pod IP of statefulset in the statefulset annotation IP range. |
| p_range     | o  | Values: <code>TRUE</code> , <code>FALSE</code> .                     |
| le          |    | Default: <code>FALSE</code> .                                        |
|             | a  |                                                                      |
|             | n  |                                                                      |
|             | U  |                                                                      |
|             | p  |                                                                      |
|             | d  |                                                                      |
|             | a  |                                                                      |
|             | t  |                                                                      |
|             | a  |                                                                      |
|             | b  |                                                                      |
|             | le |                                                                      |

---

|            |    |                                                                                                   |
|------------|----|---------------------------------------------------------------------------------------------------|
| ncp.nsx_v  | B  | Skip fatal errors and retry the request instead when no endpoint in the NSX management cluster is |
| 3.cluster_ | o  | available to serve a request.                                                                     |
| unavailabl | o  | Default: <code>FALSE</code> .                                                                     |
| e_retry    | le |                                                                                                   |
|            | a  |                                                                                                   |
|            | n  |                                                                                                   |
|            | U  |                                                                                                   |
|            | p  |                                                                                                   |
|            | d  |                                                                                                   |
|            | a  |                                                                                                   |
|            | t  |                                                                                                   |
|            | a  |                                                                                                   |
|            | b  |                                                                                                   |
|            | le |                                                                                                   |

---

---

```
ncp.nsx_v l Maximum concurrent connections to all NSX managers.
3.concurre n Default: 10.
nt_connec t
tions e
g
e
r
U
p
d
a
t
a
b
le
```

---

```
ncp.nsx_v l Time in seconds to wait before ensuring connectivity to the NSX manager.
3.conn_idl n Default: 10.
e_timeout t
e
g
e
r
U
p
d
a
t
a
b
le
```

---

```
ncp.nsx_v l Maximum number of times to retry an HTTP connection.
3.http_retr n Default: 3.
ies t
e
g
e
r
U
p
d
a
t
a
b
le
```

---

|            |          |                                                                         |
|------------|----------|-------------------------------------------------------------------------|
| ncp.nsx_v  | I        | The time in seconds before aborting a HTTP connection to a NSX Manager. |
| 3.http_tim | n        | Default: <code>10</code> .                                              |
| eout       | t        |                                                                         |
|            | e        |                                                                         |
|            | g        |                                                                         |
|            | e        |                                                                         |
|            | r        |                                                                         |
|            | <b>U</b> |                                                                         |
|            | p        |                                                                         |
|            | d        |                                                                         |
|            | a        |                                                                         |
|            | t        |                                                                         |
|            | a        |                                                                         |
|            | b        |                                                                         |
|            | le       |                                                                         |

---

|            |          |                                                                                                                                                               |
|------------|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ncp.nsx_v  | B        | L4 load balancer auto scaling mode.                                                                                                                           |
| 3.l4_lb_au | o        | Values: <code>TRUE</code> , <code>FALSE</code> .                                                                                                              |
| to_scaling | o        | Default: <code>TRUE</code> .                                                                                                                                  |
|            | le       | Updating from <code>TRUE</code> to <code>FALSE</code> is not recommended.                                                                                     |
|            | a        | Must be set to <code>FALSE</code> before activating transparent mode load balancing via annotations in NCP.                                                   |
|            | n        | Transparent mode load balancing also requires a single-tier NSX Policy API topology. For                                                                      |
|            | <b>U</b> | additional transparent mode requirements, see <a href="#">Limitations</a> in <i>VMware NSX Container Plugin 4.0.0 Release Notes</i> in the NCP documentation. |
|            | p        | Requires NCP v4.0.0 or later.                                                                                                                                 |
|            | a        |                                                                                                                                                               |
|            | t        |                                                                                                                                                               |
|            | a        |                                                                                                                                                               |
|            | b        |                                                                                                                                                               |
|            | le       |                                                                                                                                                               |

---

|           |          |                                                                           |
|-----------|----------|---------------------------------------------------------------------------|
| ncp.nsx_v | I        | Maximum number of times to retry API requests upon stale revision errors. |
| 3.retries | n        | Default: <code>10</code>                                                  |
|           | t        |                                                                           |
|           | e        |                                                                           |
|           | g        |                                                                           |
|           | e        |                                                                           |
|           | r        |                                                                           |
|           | <b>U</b> |                                                                           |
|           | p        |                                                                           |
|           | d        |                                                                           |
|           | a        |                                                                           |
|           | t        |                                                                           |
|           | a        |                                                                           |
|           | b        |                                                                           |
|           | le       |                                                                           |

---

```

ncp.nsx_v S Enable logging for snat rule.
3.snat_rul tr Values: none, basic, extended.
e_logging i Default: none.
 n
 g
 U
 p
 d
 a
 t
 a
 b
 le

```

---

```

nsx_node l The time in seconds for nsx_node_agent to recover the Hyperbus connection.
_agent.co n Default: 220.
nnect_retr t
y_timeout e
 g
 e
 r
 U
 p
 d
 a
 t
 a
 b
 le

```

---



**Note:** On an existing cluster, you can modify only the network profile parameters that are labeled **Updatable**.

## Parameter Descriptions

The following describes commonly used network profile parameters:

### `client_ssl_profile`

The primary use case for updating the network profile `client_ssl_profile` field is to configure which NSX-T client-side ssl profile is to be used by NCP.

You can change the `client_ssl_profile` parameter to a valid NSX-T client-side ssl profile ID.

Configure `client_ssl_profile` in the `cni_configurations` parameters section of your network profile.

If `client_ssl_profile` is blank or incorrect, the default NCP client SSL profile value is used instead:

- In Management API mode the default is `nsx-default-client-ssl-profile`.
- In Policy API mode the default is `default-balanced-client-ssl-profile`.

Both of these default NCP client SSL profiles use balanced-level pre-defined ciphers.

### `enable_hostport`

The primary use case for updating the network profile `enable_hostport` field is to activate or deactivate NCP support for Kubernetes Host Port.

You can change the `enable_hostport` parameter to either `true` or `false`. Configure `enable_hostport` in the `cni_configurations` parameters section of your network profile.

The Kubernetes Host Port feature allows you to expose an application to be externally accessible through a single port from outside of your cluster. For more information on Host Port, see [Pod Security Policies](#) in the Kubernetes documentation.

### **enable\_nodelocaldns**

The primary use case for updating the network profile `enable_nodelocaldns` field is to activate or deactivate NCP support for Kubernetes NodeLocal DNSCache.

You can change the `enable_nodelocaldns` parameter to either `true` or `false`. Configure `enable_nodelocaldns` in the `cni_configurations` parameters section of your network profile.

NodeLocal DNSCache improves cluster DNS performance by running a dns caching agent on cluster nodes as a DaemonSet. For more information on NodeLocal DNSCache, see [Using NodeLocal DNSCache in Kubernetes clusters](#) in the Kubernetes documentation.

### **extensions**

The primary use case for configuring the network profile CNI configuration `extensions` field is to configure the less commonly configured NCP and NSX Node Agent settings.

Use the network profiles `extensions` field to configure an NCP config map or NSX Node Agent config map property that is applicable to TKGI but is not explicitly supported as a `cni_configurations` parameter.

NCP and NSX Node Agent settings supported as explicit Network Profiles parameters cannot be configured through extensions.

For more information, see [cni\\_configurations Extensions Parameters](#) above.

### **fip\_pool\_ids**

The primary use case for updating the network profile `fip_pool_ids` field is to add additional NSX-T floating IP pool UUIDs, for a cluster or load balancer, to a cluster.

To add NSX-T floating IP pool UUIDs to a cluster:

- If creating a new cluster:
  - To use only the default floating IP Pool, leave the `fip_pool_ids` field empty or add the UUID for the default floating IP Pool to the `fip_pool_ids` field.
  - To use both the default and your defined floating IP Pools, add the UUIDs for both to the network profile `fip_pool_ids` field.
  - To replace the default floating IP Pool with your defined floating IP Pool, add only the UUIDs for your defined floating IP Pools to the network profile `fip_pool_ids` field.
- If adding a `fip_pool_ids` parameter array to the network profile of an existing cluster:

- You must include the UUID of the default floating IP pool in your `fip_pool_ids` parameter array.
- If modifying an existing `fip_pool_ids` parameter array on the network profile of an existing cluster:
  - You can add more floating IP Pool UUIDs to the array.
  - You can reorder the floating IP Pool UUIDs in the array.
  - You cannot remove any of the floating IP Pool UUIDs from an existing `fip_pool_ids` parameter array. For example, do not create a copy of a network profile, remove `fip_pool_ids` array values, and assign the new profile to the cluster that has the original profile assigned.

For information on modifying a network profile `fip_pool_ids` field, see [Customize Floating IP Pools](#).

For more information on the `fip_pool_ids` field, see [Network Profile Parameters](#) above.

## **lb\_connection\_multiplexing\_enabled**

The primary use case for updating the network profile `lb_connection_multiplexing_enabled` field is to enable NSX-T load balancer TCP multiplexing.

You can change the `lb_connection_multiplexing_enabled` parameter to either `true` or `false`.

Configure `lb_connection_multiplexing_enabled` in the `cni_configurations` parameters section of your network profile.

## **lb\_connection\_multiplexing\_number**

The primary use case for updating the network profile `lb_connection_multiplexing_number` field is to configure the maximum number of NSX-T load balancer TCP multiplexing connections.

You can change the `lb_connection_multiplexing_number` parameter to an integer value.

Configure `lb_connection_multiplexing_number` in the `cni_configurations` parameters section of your network profile.

## **nodes\_dns**

The primary use case for updating the network profile `nodes_dns` field is to update the DNS server configuration for a cluster.

You can configure the network profile `nodes_dns` field to add, modify or remove IP addresses from a cluster DNS server configuration. For more information on the network profile `nodes_dns` field and an example of a `nodes_dns` configuration, see [Specify Nodes DNS Servers](#).



**Note:** If you modify a DNS server configuration, do not exceed the maximum of three DNS server IP addresses.

## **pod\_ip\_block\_ids**

The primary use case for updating the network profile `pod_ip_block_ids` field is to add additional IP addresses for pods when a cluster is at or near the point exhausting all available public IP addresses for pods.

You can change the `pod_ip_block_ids` parameter array as follows:

- Add more IP Block IDs to the array.
- Reorder the IP Block IDs in the array.

You cannot remove any of the IP Block IDs from an existing `pod_ip_block_ids` parameter array. Do not create a copy of a network profile, remove `pod_ip_block_ids` array values, and assign the new profile to a cluster that has the original profile assigned.

For more information on modifying a network profile `pod_ip_block_ids` field, see [Add Pod IPs in Customizing Pod Networks](#). For more information on the `pod_ip_block_ids` field, see [Network Profile Parameters](#) above.

## Network Profile Use Cases

Network profiles let you customize configuration parameters for Kubernetes clusters provisioned by TKGI on vSphere with NSX-T.

You can apply a network profile to a Kubernetes cluster for the following scenarios:

| Topic                                                           | Description                                                                                                         |
|-----------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------|
| <a href="#">Size a Load Balancer</a>                            | Customize the size of the NSX-T load balancer service that is created when a Kubernetes cluster is provisioned.     |
| <a href="#">Customizing Pod Networks</a>                        | Customize Kubernetes Pod Networks, including adding pod IP addresses, subnet size, and routability.                 |
| <a href="#">Customize Node Networks</a>                         | Customize Kubernetes Node Networks, including the IP addresses, subnet size, and routability.                       |
| <a href="#">Customize Floating IP Pools</a>                     | Specify a custom floating IP pool.                                                                                  |
| <a href="#">Configure Bootstrap NSGroups</a>                    | Specify an NSX-T Namespace Group where the Kubernetes control plane nodes will be added to during cluster creation. |
| <a href="#">Configure Edge Router Selection</a>                 | Specify the NSX-T Tier-0 router where Kubernetes node and Pod networks will be connected to.                        |
| <a href="#">Specify Nodes DNS Servers</a>                       | Specify one or more DNS servers for Kubernetes clusters.                                                            |
| <a href="#">Configure DNS for Pre-Provisioned IPs</a>           | Configure DNS lookup of the Kubernetes API load balancer or ingress controller.                                     |
| <a href="#">Configure the TCP Layer 4 Load Balancer</a>         | Configure layer 4 TCP load balancer settings; use a third-party load balancer.                                      |
| <a href="#">Configure the HTTP/S Layer 7 Ingress Controller</a> | Configure layer 7 HTTP/S ingress controller settings; use third-party ingress controller.                           |
| <a href="#">Define DFW Section Markers</a>                      | Configure top or bottom section markers for explicit DFW rule placement.                                            |
| <a href="#">Configure NCP Logging</a>                           | Configure NCP logging.                                                                                              |

| Topic                     | Description                                                                                                                  |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------|
| Dedicated Tier-1 Topology | Use dedicated Tier-1 routers, rather than a shared router, for each cluster's Kube node, Namespace, and NSX-T load balancer. |

## Size a Load Balancer

This topic describes how to size a load balancer using a network profile.

## Load Balancer Sizing

When you deploy a Kubernetes cluster using Tanzu Kubernetes Grid Integrated Edition on NSX-T, an NSX-T Load Balancer is automatically provisioned. By default the size of this load balancer is [small](#). Using a network profile, you can customize the size of this load balancer and use a [medium](#) or [large](#) load balancer for Kubernetes clusters.

The NSX-T Load Balancer is a logical load balancer that handles a number of functions using virtual servers and pools. For more information, see [Supported Load Balancer Features](#) in the NSX-T documentation.

The following virtual servers are required for Tanzu Kubernetes Grid Integrated Edition:

- 1 global virtual server for the Kubernetes API which runs on the control plane nodes
- 1 TCP layer 4 virtual server for **each** Kubernetes service of `type:LoadBalancer`
- 2 HTTP and HTTPS layer 7 global virtual servers for Kubernetes ingress controller resources

The number of virtual servers that you can run depends on the size of the load balancer, which in turn depends on the size of the NSX-T Edge Node hosting the load balancer service. See [Scaling Load Balancer Resources](#) in the NSX-T documentation. Because of the number of virtual servers required by Tanzu Kubernetes Grid Integrated Edition, you can only use the large NSX Edge Node VM or the bare metal NSX Edge Node with Tanzu Kubernetes Grid Integrated Edition.

You cannot modify the Load Balancer Size configuration on an existing cluster.

The following network profile, [np-lb-med](#), defines a medium load balancer:

```
{
 "name": "np-lb-med",
 "description": "Network profile for medium NSX-T load balancer",
 "parameters": {
 "lb_size": "medium"
 }
}
```

The following network profile, [np-lb-large](#), defines a large load balancer:

```
{
 "name": "np-lb-large",
 "description": "Network profile for large NSX-T load balancer",
 "parameters": {
 "lb_size": "large"
 }
}
```

## Customizing Pod Networks (NSX-T Only)

This topic describes how VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) administrators can define TKGI network profiles for pod networks on vSphere with NSX-T integration.

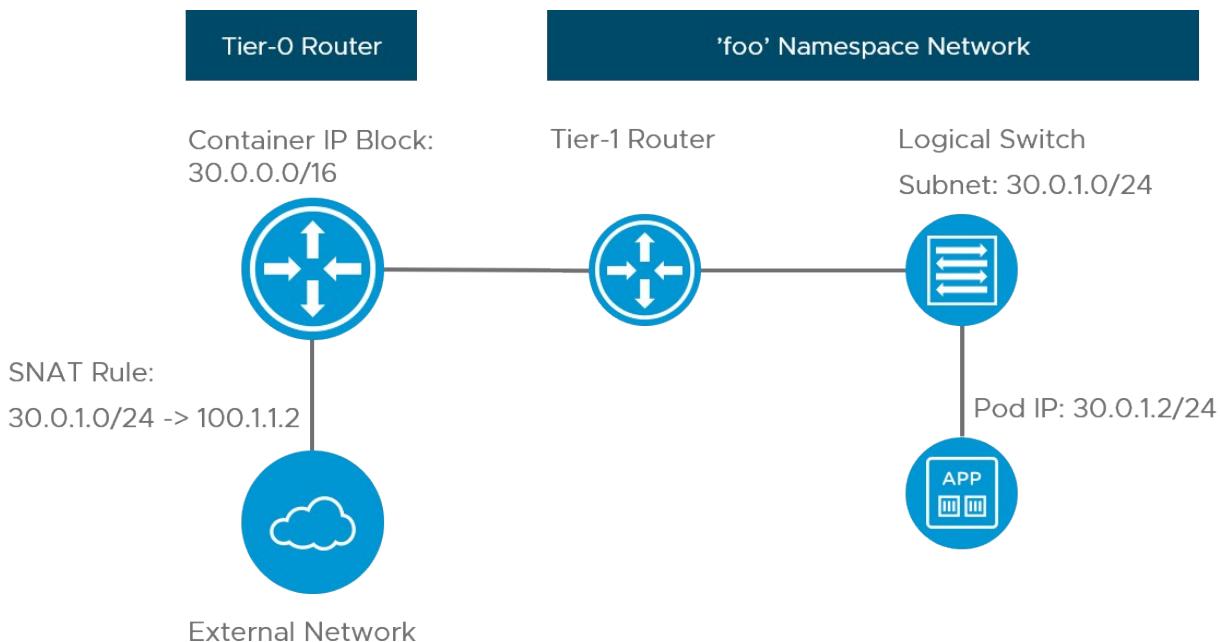
TKGI supports network profiles on TKGI on vSphere with NSX-T only.

To create or delete a network profile, you must be a cluster administrator, `pks.clusters.admin`.

## Custom Pod Networks

When you configure your NSX-T infrastructure for TKGI, you must create a **Pods IP Block**. For more information, see the [Plan IP Blocks](#) section of *Planning, Preparing, and Configuring NSX-T for Tanzu Kubernetes Grid Integrated Edition*.

By default, this subnet is non-routable. When a Kubernetes cluster is deployed, each pod receives an IP address from the **Pods IP Block** you created. Because the pod IP addresses are non-routable, NSX-T creates a SNAT rule on the Tier-0 router to allow network egress from the pods. This configuration is shown in the diagram below:



You can use a network profile to override the global **Pods IP Block** that you specify in the Tanzu Kubernetes Grid Integrated Edition tile with a custom IP block. To use a custom pods network, do the following after you deploy TKGI:

1. Define a custom IP block in NSX-T. For more information, see [Creating NSX-T Objects for Tanzu Kubernetes Grid Integrated Edition](#).
2. Define a network profile that references the custom pods IP block.

For example, the following network profile defines non-routable pod addresses from two IP blocks:

```
{
 "description": "Example network profile with 2 non-routable pod networks",
```

```

 "name": "non-routable-pod",
 "parameters": {
 "pod_ip_block_ids": [
 "ebe78a74-a5d5-4dde-ba76-9cf4067eee55",
 "ebe78a74-a5d5-4dde-ba76-9cf4067eee56"
]
 }
}

```



**Note:** You cannot use the same Pod IP Block ID (UUID) that is specified in the TKGI Tile. Create a new Pod IP Block ID (UUID) that is not referenced in the TKGI Tile and use it to define a network profile.

You can add pod addresses to an existing cluster. You cannot remove any pod addresses. For more information, see [Add Pod IPs](#) below.

## Pod Subnet Prefix

Each time a Kubernetes namespace is created, a subnet from the pods IP block is allocated. The default size of the subnet carved from this block for such purposes is /24. For more information, see the [Pods IP Block](#) section of *Planning, Preparing, and Configuring NSX-T for Tanzu Kubernetes Grid Integrated Edition*.

You can define a Network Profile using the `pod_subnet_prefix` parameter to customize the size of the pod subnet reserved for namespaces.

For example, the following network profile specifies /27 for the size of the two custom Pod IP Block IDs:

```

{
 "description": "Example network profile with 2 non-routable pod networks and custom prefix",
 "name": "non-routable-pod",
 "parameters": {
 "pod_subnet_prefix": 27,
 "pod_ip_block_ids": [
 "ebe78a74-a5d5-4dde-ba76-9cf4067eee55",
 "ebe78a74-a5d5-4dde-ba76-9cf4067eee56"
]
 }
}

```



**Note:** You cannot customize the size of the Pod IP Block ID (UUID) that is specified in the TKGI Tile. To customize the size of the Pod subnet block you must create a new Pod IP Block ID (UUID) that is not referenced in TKGI Tile and use it to define a network profile.

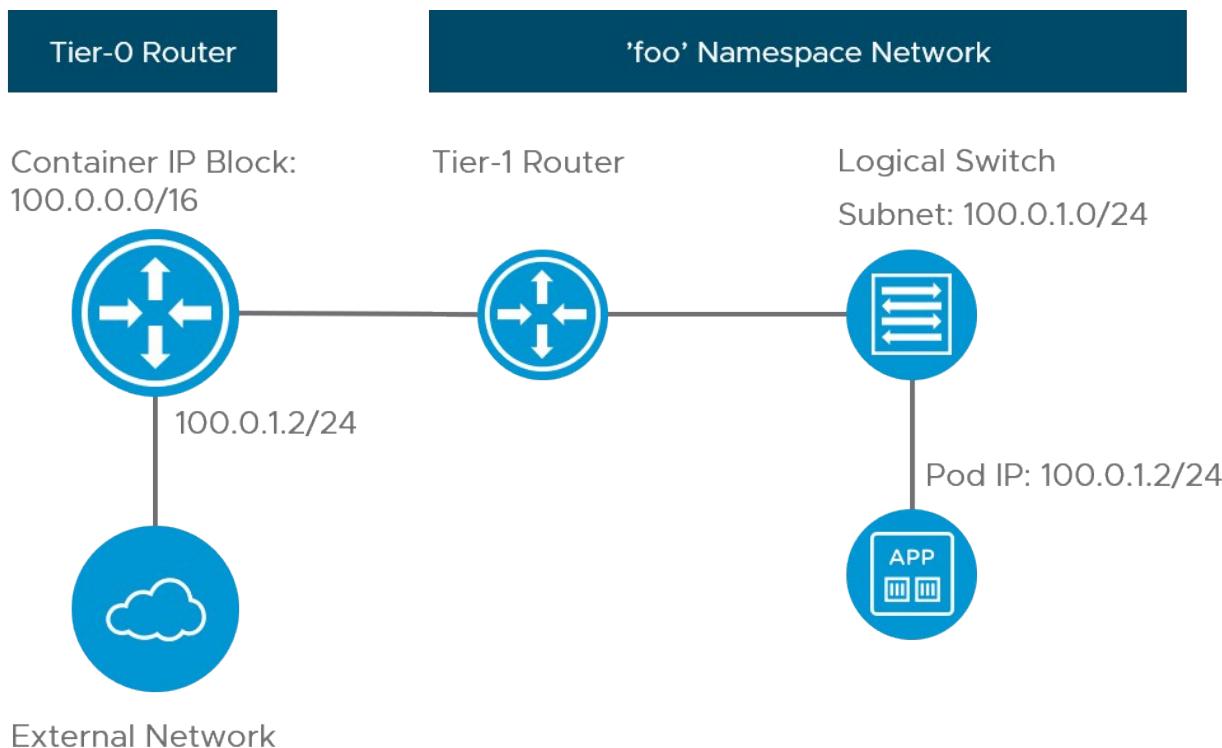


**Note:** The subnet size for a Pods IP Block must be consistent across all Network Profiles. TKGI does not support variable subnet sizes for a given IP Block.

You cannot modify the size of the pod subnet configuration on an existing cluster.

## Routable Pod Networks

Using a network profile, you can assign routable IP addresses from a dedicated routable IP block to pods in your Kubernetes cluster. When a cluster is deployed using that network profile, the routable IP block overrides the default non-routable IP block described created for deploying TKGI. When you deploy a Kubernetes cluster using that network profile, each pod receives a routable IP address. This configuration is shown in the diagram below. If you use routable pods, the SNAT rule is not created.



To use routable pods, do the following after you deploy TKGI:

1. Define a routable IP block in NSX-T. For more information, see [Creating NSX-T Objects for Tanzu Kubernetes Grid Integrated Edition](#).
2. Define a network profile that references the routable IP block.

For example, the following network profile defines routable pod addresses from two IP blocks:

```
{
 "description": "Example network profile with 2 routable pod networks and custom prefix",
 "name": "small-routable-pod",
 "parameters": {
 "pod_routable": true,
 "pod_subnet_prefix": 27,
 "pod_ip_block_ids": [
 "ebe78a74-a5d5-4dde-ba76-9cf4067eee55",
 "ebe78a74-a5d5-4dde-ba76-9cf4067eee56"
]
}
```

```
 }
}
```



**Note:** You cannot use the same Pod IP Block ID (UUID) that is specified in the TKGI Tile. Create a new Pod IP Block ID (UUID) that is not referenced in TKGI Tile and use it to define a network profile.

You can add pod addresses to an existing cluster. You cannot remove any pod addresses or modify the size of the pod subnet configuration on an existing cluster. For more information, see [Add Pod IPs](#) below.

## Add Pod IPs

If a cluster exhausts the number of IP addresses allocated to pods, you can use the network profile `pod_ip_block_ids` field to add pod IP addresses to existing clusters.

To add pod IP addresses to the network profile on an existing cluster:

1. Create one or more new Pod IP Blocks in NSX-T Manager.
2. Follow these procedures in *Creating and Deleting Network Profiles*:
  1. [Create a Modified Network Profile Configuration](#)
  2. [Create a Modified Network Profile](#)
  3. [Update the Cluster With a Modified Network Profile](#)

On update, the cluster automatically starts using IPs from the second block after the first block is exhausted.



**Note:** You cannot change a cluster's network profile to remove pod IP block IDs.

## Customize Node Networks

This topic describes how to define Network Profiles for customizing Kubernetes node networks provisioned with VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) on vSphere with NSX.

For information on how to define Network Profiles for other cluster customizations, see [Creating and Managing Network Profiles \(NSX Only\)](#).

## Configurable Node Network IP Blocks

You can use Network Profiles to configure a Kubernetes cluster with a custom Node Network IP Block.

A Network Profile **Node IP Block** is used by TKGI to assign address space to Kubernetes nodes when new clusters are deployed or a cluster increases its scale.

Your Network Profile Node IP Block configuration can define one or more custom Node IP Block networks, specify the size of the node subnet, and specify if the network is routable:

```
nodes-network.json
```

```
{
 "description": "DESCRIPTION",
 "name": "NAME",
 "parameters": {
 "node_ip_block_ids": [
 NODE - I P - B L O C K - I D S
],
 "node_routable": ROUTABLE,
 "node_subnet_prefix": SIZE
 }
}
```

Where:

- **DESCRIPTION** is the description of the Nodes Network IP Block.
- **NAME** is the name of the Nodes Network IP Block.
- **NODE-IP-BLOCK-IDS** is the comma-separated list of double-quoted node network Node IP Blocks for the cluster. For more information, see [Node IP Block IDs](#) below.
- **ROUTABLE** whether the node network is routable or non-routable. Accepts only `true` or `false`. For more information, see [Node Routable](#) below.
- **SIZE** is the subnet size. Accepts only standard integer subnet size values. For more information, see [Node Subnet Prefix](#) below.

For example:

```
nodes-network.json
{
 "description": "Configurable Nodes Network IP Block",
 "name": "network-profile_nodes-ip-block",
 "parameters": {
 "node_ip_block_ids": [
 "2250dc43-63c8-4bb8-b8cf-c6e12ccfb7de", "3d577e5c-dcaf-4921-9458-d12b0e1318e6"
],
 "node_routable": true,
 "node_subnet_prefix": 20
 }
}
```

## Node IP Block IDs

The network profile `node_ip_block_ids` parameter allows you to specify one or more Kubernetes node network Node IP Blocks for your clusters.

When a network profile is applied to a Kubernetes cluster, TKGI automatically creates a node subnet from one of the available IP blocks in the `node_ip_block_ids` configuration.

If the IP block is exhausted, the cluster uses one of the alternate IP blocks specified in the `node_ip_block_ids` configuration to create the node subnet.

The set of alternate IP blocks available to a cluster is fixed at cluster creation. If a modified Network Profile is applied to an existing cluster, the blocks available to the cluster remain unchanged.

The Network Profile `node_ip_block_ids` parameter is **Updatable**, but supports only adding new alternate IP blocks to the existing set of blocks.



**Note:** If you apply a new network profile to an existing cluster, the `node_ip_block_ids` configuration in the replacement network profile must include all of the Node IP Blocks specified in the cluster's original network profile.

If your network profile does not include a `node_ip_block_ids` configuration, TKGI creates a node subnet from one of the available IP blocks in the Node IP Blocks specified on the TKGI tile.



**Note:** When replacing a network profile that does not explicitly specify a `node_ip_block_ids` configuration, the replacement network profile must include the Node IP Blocks specified on the TKGI tile.

## Node Routable

The `node_routable` boolean lets you specify if the node network is routable or non-routable. This is the equivalent of enabling or deactivating NAT mode in the TKGI tile.

If the node network is configured as non-routable, `"node_routable":false`, the node network uses NAT mode. In this case, you must make sure that Kubernetes nodes have access to BOSH and other TKGI Management Plane components. See [Create Management Plane](#) in *Installing and Configuring NSX-T Data Center v3.0 for TKGI* for more information.

If the node network is configured as routable, `"node_routable":true`, the IP address space must be an externally routable address block.

The `node_routable` configuration on an existing cluster cannot be updated.



**Note:** The default routable setting for the node network is determined based on the selection made in the TKGI tile. If **NAT mode** is selected, the node network is non-routable. To override the default selection, provide the `node_routable` parameter in the network profile.

## Node Subnet Prefix

Configure the Node IP Block `node_subnet_prefix` parameter to specify a subnet size that optimizes the use of network address space for the number of nodes in your Kubernetes cluster.

For example, if the TKGI administrator has configured the default in the TKGI tile to be a routable network for the Node IP Block, the Kubernetes cluster administrator can deploy the cluster in the NAT'ed mode (non-routable) by specifying a network profile with an IP block that supports the NAT'ed address range.

By default, each Kubernetes cluster deployed by TKGI is allocated a /24 subnet, which allows up to 256 IP addresses to be assigned.

The `node_subnet_prefix` configuration on an existing cluster cannot be updated.



**Note:** Configure **Node Subnet Prefix** when your cluster nodes use a globally routable address space with the `node_routable` option set to `true`.

## Customize Floating IP Pools

This topic describes how to define network profiles for custom floating IP pools.

### Create a Custom Floating IP Pool

To deploy Tanzu Kubernetes Grid Integrated Edition to vSphere with NSX-T, you must define a Floating IP Pool in NSX Manager. IP addresses from the Floating IP Pool are used for SNAT IP addresses whenever a Namespace is created (NAT mode). In addition, IP addresses from the Floating IP Pool are assigned to load balancers automatically provisioned by NSX-T, including the load balancer fronting the TKGI API server and load balancers for pod ingress. For more information, see the [Plan Network CIDRs](#) section of *Planning, Preparing, and Configuring NSX-T for Tanzu Kubernetes Grid Integrated Edition*.

You can define a network profile that specifies a custom floating IP pool to use instead of the default pool specified in the Tanzu Kubernetes Grid Integrated Edition tile.



**Note:** TKGI allocates IP Addresses from the start of the floating IP pool range. To avoid conflicts with internal TKGI functions, always use IP addresses from the end of the floating IP pool.

To define a custom floating IP pool, follow the steps below:

1. Create a floating IP pool using NSX Manager prior to provisioning a Kubernetes cluster using Tanzu Kubernetes Grid Integrated Edition. For more information, see [Create IP Pool](#) in the NSX-T documentation.
2. Ensure routing to your external Tier-0 Router allows traffic to the new custom Floating IP subnet.
3. Define a network profile with a `fip_pool_ids` array containing the UUIDs for the floating IP pools that you defined. If you want to include the default floating IP pool, also add the UUID of the default floating IP pool to the `fip_pool_ids` array.

The following example defines a custom floating IP pool:

```
{
 "name": "np-custom-fip",
 "description": "Network Profile for Custom Floating IP Pool",
 "parameters": {
 "fip_pool_ids": [
 "e50e8f6e-1a7a-45dc-ad49-3a607baa7fa0",
 "e b e 7 8 a 7 4 - a 5 d 5 - 4 d d e - b a 7 6 - 9 c f 4 0 6 7 e e e 5 5 "
]
 }
}
```

}

The example above uses two floating IP pools. With this configuration, if the first pool of IP addresses, `e50e8f6e-1a7a-45dc-ad49-3a607baa7fa0`, is exhausted, the system will use the IP addresses in the next IP pool that is listed, `ebe78a74-a5d5-4dde-ba76-9cf4067eee55`.



**Note:** If you are using multiple Floating IP Pools within the same Tier-0 router, the Floating IP Pools cannot overlap. Overlapping Floating IP Pools are allowed across Tier-0 routers, but not within the same Tier-0 router.

## Modify a Floating IP Pool

You can modify the floating IP pool of an existing cluster. For more information, see [fip\\_pool\\_ids](#) in *Creating and Managing Network Profiles*.

## Configure Bootstrap NSGroups

This topic describes how to define network profiles for Kubernetes clusters provisioned with VMware Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T.

## Bootstrap Security Group

Most of the NSX-T virtual interface tags used by Tanzu Kubernetes Grid Integrated Edition are added to the Kubernetes control plane node or nodes during the node initialization phase of cluster provisioning. To add tags to virtual interfaces, the Kubernetes control plane node needs to connect to the NSX-T Manager API. Network security rules provisioned prior to cluster creation time do not allow nodes to connect to NSX-T if the rules are based on a Namespace Group (NSGroup) managed by Tanzu Kubernetes Grid Integrated Edition.

To address this bootstrap issue, Tanzu Kubernetes Grid Integrated Edition exposes an optional configuration parameter in Network Profiles to systematically add Kubernetes control plane nodes to a pre-provisioned NSGroup. The BOSH vSphere cloud provider interface (CPI) has the ability to use the NSGroup to automatically manage members following the BOSH VM lifecycle for Kubernetes control plane nodes.

To configure a Bootstrap Security Group, complete the following steps:

1. Create the NSGroup in NSX Manager prior to provisioning a Kubernetes cluster using Tanzu Kubernetes Grid Integrated Edition. For more information, see [Create an NSGroup](#) in the NSX-T documentation.
2. Define a network profile that references the NSGroup UUID that the BOSH CPI can use to bootstrap the control plane node or nodes.

For example, the following network profile specifies an NSGroup for the BOSH CPI to use to dynamically update Kubernetes control plane node memberships:

```
{
 "name": "np-boot-nsgroups",
 "description": "Network Profile for Customer B",
```

```
 "parameters": {
 "master_vms_nsgroup_id": "9b8d535a-d3b6-4735-9fd0-56305c4a5293"
 }
 }
```

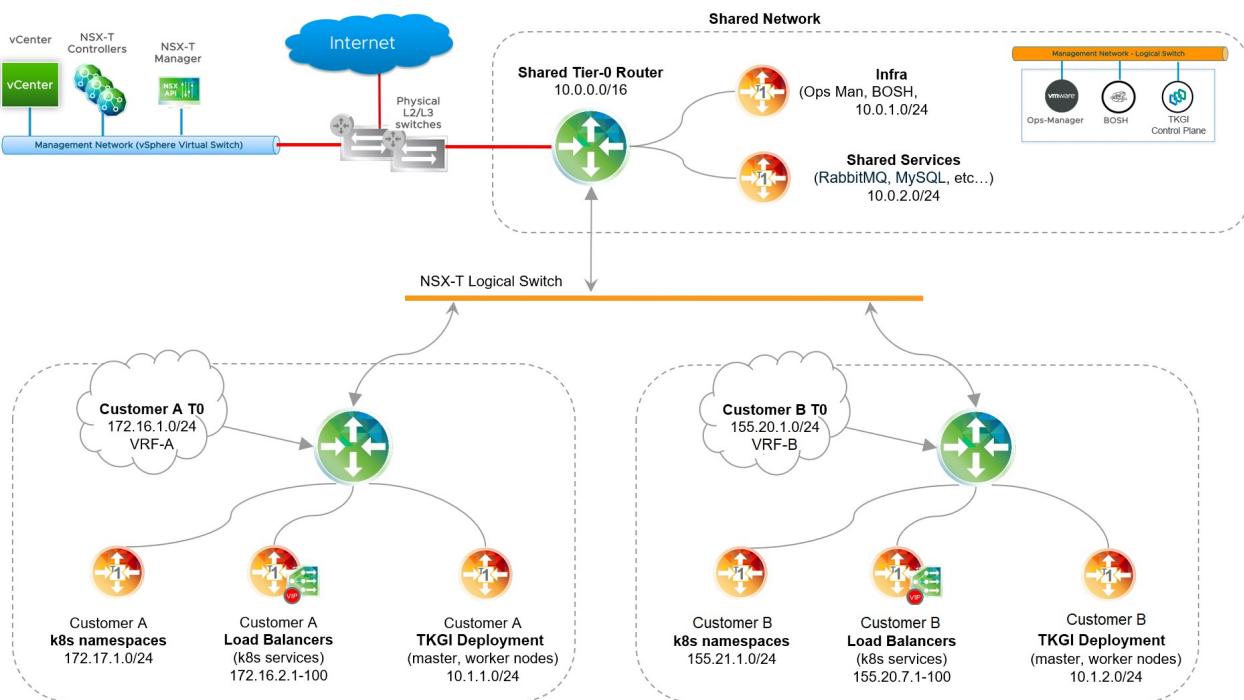
You cannot modify the Bootstrap Security Group configuration on an existing cluster.

# Configure Edge Router Selection

This topic describes how to define network profiles for Kubernetes clusters provisioned with VMware Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T.

# Edge Router Selection

Using Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T, you can deploy Kubernetes clusters on dedicated Tier-0 routers, creating a multi-tenant environment for each Kubernetes cluster. As shown in the diagram below, with this configuration a shared Tier-0 router hosts the TKGI control plane and connects to each customer Tier-0 router using BGP. To support multi-tenancy, configure firewall rules and security settings in NSX Manager.



To deploy Kubernetes clusters on tenancy-based Tier-0 router(s), follow the steps below:

1. For each Kubernetes tenant, create a dedicated Tier-0 router, and configure static routes, BGP, NAT and Edge Firewall security rules as required by each tenant. For instructions, see [Isolating Tenants](#).
  2. Define a network profile per tenant that references the Tier-0 router UUID provisioned for that tenant. For example, the following network profiles define two tenant Tier-0 routers with a NATed topology.

```
np_customer_A-NAT.json
{
 "description": "network profile for Customer A",
```

```

 "name": "network-profile-Customer-A",
 "parameters": {
 "lb_size": "medium",
 "t0_router_id": "82e766f7-67f1-45b2-8023-30e2725600ba",
 "fip_pool_ids": ["8ec655f-009a-79b7-ac22-40d37598c0ff"],
 "pod_ip_block_ids": ["fce766f7-aaf1-49b2-d023-90e272e600ba"]
 }
}

```

```

np_customer_B-NAT.json
{
 "description": "network profile for Customer B",
 "name": "network-profile-Customer-B",
 "parameters": {
 "lb_size": "small",
 "t0_router_id": "a4e766cc-87ff-15bd-9052-a0e2425612b7",
 "fip_pool_ids": ["4ec625f-b09b-29b4-dc24-10d37598c0d1"],
 "pod_ip_block_ids": ["91e7a3a1-c5f1-4912-d023-90e272260090"]
 }
}

```

The following network profiles define two customer Tier-0 routers for a no-NAT topology:

```

np_customer_A.json
{
 "description": "network profile for Customer A",
 "name": "network-profile-Customer-A",
 "parameters": {
 "lb_size": "medium",
 "t0_router_id": "82e766f7-67f1-45b2-8023-30e2725600ba",
 "fip_pool_ids": [
 "8ec655f-009a-79b7-ac22-40d37598c0ff",
 "7ec625f-b09b-29b4-dc24-10d37598c0e0"
],
 "pod_routable": "true",
 "pod_ip_block_ids": [
 "fce766f7-aaf1-49b2-d023-90e272e600ba",
 "6faf46fd-ccce-4332-92d2-d918adcccc0"
]
 }
}

```

```

np_customer_B.json
{
 "description": "network profile for Customer B",
 "name": "network-profile-Customer-B",
 "parameters": {
 "lb_size": "small",
 "t0_router_id": "a4e766cc-87ff-15bd-9052-a0e2425612b7",
 "fip_pool_ids": [
 "4ec625f-b09b-29b4-dc24-10d37598c0d1",
 "6ec625f-b09b-29b4-dc24-10d37598dDd1"
],
 "pod_routable": "true",
 "pod_ip_block_ids": [
 "91e7a3a1-c5f1-4912-d023-90e272260090",
 "6faf46fd-ccce-4332-92d2-d918adcccc0"
]
 }
}

```

```

]
 }
}

```



**Note:** The `pod_routable` parameter controls the routing behavior of a tenant Tier-0 router. If the parameter is set to `true`, the custom Pods IP Block subnet is routable and NAT is not used. If `pod_routable` is not present or is set to `false`, the custom Pods IP Block is not routable and the tenant Tier-0 is deployed in NAT mode.

## Specify Nodes DNS Servers

This topic describes how to define network profiles for Kubernetes clusters provisioned with VMware Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T.

## DNS Configuration for Kubernetes Clusters

You can specify multiple DNS entries in a Network Profile to override the **Nodes DNS** parameter configured in the TKGI tile. In a multi-tenant environment, for example, each tenant can have a different set of DNS servers to do a DNS lookup.

Using a network profile, you can define one or more DNS servers for use with Kubernetes clusters. Elements in the `nodes_dns` field of a network profile override the DNS server that is configured in the Networking section of the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Networking](#).

The `nodes_dns` field accepts an array with up to three elements. Each element must be a valid IP address of a DNS server. If you are deploying Tanzu Kubernetes Grid Integrated Edition in a multi-tenant environment with multiple Tier-0 routers and a single TKGI foundation (installation) shared across all the tenants, or if you have shared services that can be accessed by all Kubernetes clusters deployed across multiple Tier-0 routers, the first DNS server entered should be a shared DNS server. Subsequent DNS entries in the Network Profile can be specific to the tenant.



**Note:** TKGI allocates IP Addresses from the start of the floating IP pool range. To avoid conflicts with internal TKGI functions, always use IP addresses from the end of the floating IP pool.

The following example network profile, `nodes-dns.json`, demonstrates the configuration of the `nodes_dns` parameter with 3 DNS servers. Each entry is the IP address of a DNS server, with the first entry being a public DNS server.

```

nodes-dns.json
{
 "description": "Overwrite Nodes DNS Entry",
 "name": "nodes_dns_multiple",
 "parameters": {
 "nodes_dns": [
 "8.8.8.8", "192.168.115.1", "192.168.116.1"
]
 }
}

```

```
 }
}
```

You can modify the Node DNS configuration on an existing cluster.

## Configure DNS for Pre-Provisioned IPs

This topic describes how to define network profile for performing DNS lookup of the pre-provisioned IP addresses for the Kubernetes API load balancer and ingress controller.

### About DNS Lookup of Pre-Provisioned IP Addresses

In an Tanzu Kubernetes Grid Integrated Edition environment on NSX-T, when you provision a Kubernetes cluster using the command `tkgi create-cluster`, NSX-T creates a layer 4 load balancer that fronts the Kubernetes API server running on the control plane node(s). In addition, NCP creates two layer 7 virtual servers (HTTP and HTTPS) as front-end load balancers for the ingress resources in Kubernetes servers.

The IP addresses that are assigned to the API load balancer and ingress controller are derived from the floating IP pool in NSX-T. These IP addresses are not known in advance, and you have to wait for the IP addresses to be allocated to know what they are so you can update your DNS records.

If you want to pre-provision these IP addresses, you define a network profile to lookup the IP addresses for these components from your DNS server. In this way you can tell TKGI what IP addresses to use for these resources when the cluster is created, and be able to have DNS records for them so FQDNs can be used.

### DNS Lookup Parameters

Using the `dns_lookup_mode` parameter, you can define a network profile to specify the lookup mode: `API` or `API_INGRESS`. If the mode is `API`, TKGI will perform a lookup of the pre-provisioned IP address for the Kubernetes API load balancer. If the mode is `API_INGRESS`, TKGI will perform a lookup of the pre-provisioned IP addresses for the Kubernetes API load balancer and the ingress controller.

The IP addresses used must come from the floating IP pool. The floating IP pool comes from the TKGI tile configuration unless specified in the network profile.



**Note:** TKGI allocates IP Addresses from the start of the floating IP pool range. To avoid conflicts with internal TKGI functions, always use IP addresses from the end of the floating IP pool.

The DNS lookup, whether for the Kubernetes control plane node(s) load balancer or the ingress controller, is performed in the Kubernetes control plane VM using the DNS server(s) configured in the TKGI tile or the `nodes_dns` field in the network profile.

You cannot modify the DNS lookup mode configuration on an existing cluster.

### Example API Load Balancer Lookup

The following network profile, `api.json`, triggers a DNS lookup for the Kubernetes control plane

node(s) IP address. In this example, a custom floating IP pool is specified, and DNS servers. If these parameters are not specified, the values in the TKGI tile are used.

```
{
 "name": "example-network-profile",
 "description": "Network profile using API lookup mode",
 "parameters": {
 "nodes_dns": [
 "8.8.8.8", "192.168.115.1", "192.168.116.1"
],
 "fip_pool_ids": [
 "FIP-POOL-ID1",
 "FIP-POOL-ID2"
],
 "dns_lookup_mode": "API"
 }
}
```

Where `FIP-POOL-ID1` and `FIP-POOL-ID2` are Floating IP Pool IDs.

## Performing DNS Lookup of the Ingress Controller Using Network Profile

The following example network profile, `api_ingress.json`, triggers a DNS lookup for the Kubernetes control plane node(s) IP address and the ingress controller IP address.

```
{
 "name": "api_ingress",
 "description": "Network profile using API_INGRESS dns lookup mode",
 "parameters": {
 "fip_pool_ids": [
 "FIP-POOL-ID1",
 "FIP-POOL-ID2"
],
 "dns_lookup_mode": "API_INGRESS",
 "ingress_prefix": "INGRESS-SUBDOMAIN"
 }
}
```

Where:

- `FIP-POOL-ID1` and `FIP-POOL-ID2` are Floating IP Pool IDs.
- `INGRESS-SUBDOMAIN` is the ingress subdomain prefix.

Because DNS mode is set to `API_INGRESS`, TKGI creates the cluster with `ingress_prefix.hostname` as the Kubernetes control plane FQDN. TKGI confirms that the ingress subdomain can be resolved as a subdomain prefix on the host before creating new clusters.

## Setting the Control Plane Node IP Address on the Command Line

As an alternative to DNS lookup, you can specify a fixed IP address in the command line so that it will be used for the Kubernetes control plane node(s) load balancer.

Previously, to create a cluster, you were required to specify an external hostname for the cluster. For example:

```
$ tkgi create-cluster my-cluster --external-hostname example.hostname --plan small
```

Now you can specify the IP address for the load balancer that fronts the Kubernetes control plane node(s) using the `--external-hostname` or `-e` flag. For example:

```
$ tkgi create-cluster my-cluster -e 192.168.160.20 -p small
```

The IP address that you use must belong to a valid floating IP pool created in NSX-T.



**Note:** TKGI allocates IP Addresses from the start of the floating IP pool range. To avoid conflicts with internal TKGI functions, always use IP addresses from the end of the floating IP pool.

## Configure the TCP Layer 4 Load Balancer

This topic describes how to define network profile to configure the NSX-T Load Balancer for VMware Tanzu Kubernetes Grid Integrated Edition.

### Overview

The NSX-T Load Balancer is a logical load balancer that handles a number of functions using virtual servers and pools.

The NSX-T load balancer creates a load balancer service for each Kubernetes cluster provisioned by Tanzu Kubernetes Grid Integrated Edition with NSX-T. For each load balancer service, NCP, by way of the CRD, creates corresponding NSXLoadBalancerMonitor objects.

By default Tanzu Kubernetes Grid Integrated Edition deploys the following NSX-T virtual servers for each Kubernetes cluster:

- One TCP layer 4 load balancer virtual server for the Kubernetes API server.
- One TCP layer 4 auto-scaled load balancer virtual server for **each** Kubernetes service resource of `type: LoadBalancer`.
- Two HTTP/HTTPS layer 7 ingress routing virtual servers. These virtual servers are attached to the Kubernetes Ingress Controller cluster load balancer service and can be manually scaled. Tanzu Kubernetes Grid Integrated Edition uses Kubernetes custom resources to monitor the state of the NSX-T load balancer service and scale the virtual servers created for ingress.

For information about configuring TCP layer 4 ingress controller see [Configure the TCP Ingress Controller Network Profile](#), below.

For information about configuring layer 7 ingress routing load balancers see [Scaling the HTTP/S Layer 7 Ingress Load Balancers Using the LoadBalancer CRD](#). For information about configuring the layer 7 ingress controller see [Defining Network Profiles for the HTTP/S Layer 7 Ingress Controller](#).

For more information about the NSX-T Load Balancer, see [Create an IP Pool in Manager Mode](#) or [Add an IP Address Pool](#) in the VMware documentation.

For more information about Kubernetes custom resources, see [Custom resources](#) in the Kubernetes documentation.

## Configure the TCP Ingress Controller Network Profile

The TCP layer 4 virtual server provisioned for each Kubernetes service is controlled by the parameters exposed in a network profile.



**Note:** The TCP layer 4 virtual server that fronts the Kubernetes API server is always created, and it is not controlled by the parameters exposed in the network profile.

### NSX-T TCP Ingress Controller Network Profile Configuration

The NSX Ingress Controller is configured using the `ncp.ini` network profile configuration file.

The TCP Ingress Controller network profile has the following format:

```
{
 "name": "network_profile",
 "description": "DESCRIP",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "nsx_lb": NSX-LB,
 "x_forwarded_for": "FORWARD-TYPE",
 "max_l4_lb_service": MAX-SERVERS,
 "l4_persistence_type": "source_ip",
 "l4_lb_algorithm": "LB-BEHAVIOR"
 }
 }
 }
}
```

Where:

- `DESCRIP` is your description for this network profile configuration.
- `NSX-LB` is your preference for whether the NSX-T Load Balancer is used for your Kubernetes clusters. For more information see [Configure Which NSX Load Balancer to Use](#), below.
- `FORWARD-TYPE` (Optional) is your request header original client source IP. For more information see the [Configure the Client Source IP Address](#), below.
- `MAX-SERVERS` is your maximum number of layer 4 virtual servers per cluster. For more information see the [Configure the Maximum Number of Layer 4 Load Balancer Virtual Servers](#), below.
- `LB-BEHAVIOR` (Optional) is your load balancer behavior. For more information see the [Configure the Layer 4 Load Balancer Algorithm](#), below.

For example:

```
{
```

```

"name": "network_profile",
"description": "DESCRIP",
"parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "nsx_lb": true,
 "x_forwarded_for": "replace",
 "max_14_lb_service": 10,
 "l4_persistence_type": "source_ip",
 "l4_lb_algorithm": "weighted_round_robin"
 }
 }
}
}

```

The following table describes the Ingress Controller configuration parameters:

| Parameter                       | Type     | Description                                                                                                                                                            |
|---------------------------------|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>name</code>               | String   | User-defined name of the network profile.                                                                                                                              |
| <code>description</code>        | String   | User-defined description for the network profile.                                                                                                                      |
| <code>parameters</code>         | Map      | Map containing one or more key-value pairs.                                                                                                                            |
| <code>cni_configurations</code> | Map      | Map containing <code>type</code> and <code>parameters</code> key-value pairs for configuring NCP.                                                                      |
| <code>type</code>               | Constant | Values: "nsxt".                                                                                                                                                        |
| <code>parameters</code>         | Map      | Map containing one or more key-value pairs for NCP settings.                                                                                                           |
| <code>nsx_lb</code>             | Boolean  | Use NSX-T layer 4 virtual server for each Kubernetes service of type LoadBalancer.<br>Values: <code>true</code> , <code>false</code> .<br>Default: <code>true</code> . |

|                                  |                                                            |                                                                                                                                                                                                                                                                                                      |
|----------------------------------|------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>x_forwarded_for</code>     | St<br>rin<br>g<br><b>U</b><br><b>pd</b><br>at<br>ab<br>le  | Sets the original client source IP in the request header. Enabling the network profile <code>x_forwarded_for</code> parameter automatically enables the <code>x_forwarded_port</code> and <code>x_forward_protocol</code> parameters.<br>Values: "none", "insert" and "replace".<br>Default: "none". |
| <code>max_14_lb_service</code>   | Int<br>eg<br>er<br><b>U</b><br><b>pd</b><br>at<br>ab<br>le | Limit the maximum number of layer 4 virtual servers per cluster.<br>Minimum Value:1.<br>See also: <code>14_lb_algorithm</code> and <code>14_persistence_type</code> .                                                                                                                                |
| <code>14_persistence_type</code> | St<br>rin<br>g<br><b>U</b><br><b>pd</b><br>at<br>ab<br>le  | Connection stickiness based on <code>source_ip</code> .<br>Values: "source_ip".<br>See also: <code>14_lb_algorithm</code> and <code>max_14_lb_service</code> .                                                                                                                                       |
| <code>14_lb_algorithm</code>     | St<br>rin<br>g<br><b>U</b><br><b>pd</b><br>at<br>ab<br>le  | Layer 4 load balancer behavior.<br>Values: "round_robin", "least_connection", "ip_hash", "weighted_round_robin".<br>Default: "round_robin".<br>See also: <code>14_persistence_type</code> and <code>max_14_lb_service</code> .                                                                       |

The `nsx_lb` parameter is used to control the TCP layer 4 virtual server that is provisioned for each Kubernetes service of type: [LoadBalancer](#).

When you configure an NSX Load Balancer as your Kubernetes cluster [ingress resource](#), NCP instructs the NSX-T Load Balancer to provision two layer 7 virtual services (HTTP and HTTPS) as the cluster [Ingress Controller](#):

| <code>nsx_lb</code> setting | Description                                                                                           |
|-----------------------------|-------------------------------------------------------------------------------------------------------|
| <code>nsx_lb: true</code>   | Use an NSX-T Layer 4 LoadBalancer and NCP-provisioned Layer 7 Ingress Controller.                     |
| <code>nsx_lb: false</code>  | Use a third-party load balancer and a third-party ingress controller, such as <a href="#">NGINX</a> . |

## Configure Which NSX Load Balancer to Use

The `nsx_lb` flag controls whether to deploy either the NSX-T Load Balancer or a third-party load balancer, such as Nginx.

The `nsx_lb` parameter accepts `true` or `false`. The default setting is `true` and the NSX-T Load Balancer is deployed. To use a third party load balancer, set this parameter to `false`.

For example:

```
{
 "name": "example_network_profile",
 "description": "nsx_lb is enabled",
 "parameters": {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "nsx_lb": false
 }
 }
 }
}
```



**Note:** The `nsx_lb` parameter maps to the `use_native_loadbalancer` parameter in NCP.ini.

For more information about configuring the NSX Ingress Controller component of the NSX-T Load Balancer, see [Defining Network Profiles for the HTTP/S Layer 7 Ingress Controller](#).

### Configure the Client Source IP Address

The `X-Forwarded-For` HTTP header field is used to identify the originating IP address of a client connecting to a web server through an HTTP proxy or load balancer.

In network profile, the `x_forwarded_for` parameter can be enabled to ensure that the client IP address will be set in the HTTP request header. The `x_forwarded_for` parameter is useful in situations where it is important to know the source IP address of the client request, such as for auditing purposes.



**Note:** Enabling the network profile `x_forwarded_for` parameter automatically enables the `x_forwarded_port` and `x_forward_protocol` parameters. This is supported with NSX-T v3.0.x and later.

The `x_forwarded_for` parameter type is String that accepts `"insert"` and `"replace"`. Any other type will be rejected. Missing entry is accepted.

If `x_forwarded_for` is set to `"insert"`, the client source IP will be appended (comma separated) to the existing set of client source IP addresses. If `x_forwarded_for` is set to `"replace"`, any existing client source IP address will be replaced with the current client source IP address.

For example, with the following network profile the source IP address of the client will be appended to the existing set of client source IP addresses:

```
{
 "name": "example-network-profile",
 "description": "x_forwarded_for insert",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "x_forwarded_for": "insert"
 }
 }
 }
}
```

```

 }
 }
}
```

For example, with the following network profile the existing source IP address of the client will replace all other client source IP entries:

```
{
 "name": "example-network-profile",
 "description": "x_forwarded_for_replace",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "x_forwarded_for": "replace"
 }
 }
 }
}
```

## Configure the Maximum Number of Layer 4 Load Balancer Virtual Servers

The default NSX-T Load Balancer behavior is that auto-scaling is unlimited. This means that the number of layer 4 virtual servers that can be deployed is governed only by the capacity of the Edge Cluster where the load balancer service is deployed. As a result, in theory it is possible for a single Kubernetes cluster to use up all of the layer 4 virtual servers that the Edge Cluster can support.

The `max_14_service` parameter sets the upper limit for the number of virtual servers that can be used by a Kubernetes cluster. You can use this parameter to limit the number of virtual servers that can be created per Kubernetes cluster.

The `max_14_lb_service` data type is an integer. The value must be larger or equal to 1. Missing entry is accepted.

For example, the following network profile uses the `max_14_lb_service` parameter to limit the number of layer 4 virtual servers to 100 per cluster:

```
{
 "name": "example_network_profile",
 "description": "max_14_lb_service",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "max_14_lb_service": 100
 }
 }
 }
}
```

## Configure the Layer 4 Persistence Type

The `14_persistence_type` is used to set connection stickiness based on `source_ip`.

The `14_persistence_type` data type is string. The only accepted value is `source_ip`.

```
{
 "name": "example_network_profile",
 "description": "l4_persistence_type",
 "parameters": {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "l4_persistence_type": "source_ip"
 }
 }
 }
}
```

## Configure the Layer 4 Load Balancer Algorithm

The `l4_lb_algorithm` is used to set the algorithm type for the layer 4 NSX-T Load Balancer service.

The `l4_lb_algorithm` data type is string enumeration that accepts one of the following values:

- `"round_robin"` (default)
- `"least_connection"`
- `"ip_hash"`
- `"weighted_round_robin"`

For example, the following network profile specifies the `weighted_round_robin` as the load balancer algorithm:

```
{
 "name": "example_network_profile",
 "description": "l4_lb_algorithm",
 "parameters": {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "l4_lb_algorithm": "weighted_round_robin"
 }
 }
 }
}
```

## Configure the HTTP/S Layer 7 Ingress Controller

This topic describes how to define network profiles for Kubernetes clusters provisioned with VMware Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T.

### Overview

The NSX-T Load Balancer is a logical load balancer that handles a number of functions using virtual servers and pools.

The NSX-T load balancer creates a load balancer service for each Kubernetes cluster provisioned by Tanzu Kubernetes Grid Integrated Edition with NSX-T. For each load balancer service, NCP, by way

of the CRD, creates corresponding NSXLoadBalancerMonitor objects.

By default Tanzu Kubernetes Grid Integrated Edition deploys the following NSX-T virtual servers for each Kubernetes cluster:

- One TCP layer 4 load balancer virtual server for the Kubernetes API server.
- One TCP layer 4 auto-scaled load balancer virtual server for **each** Kubernetes service resource of `type: LoadBalancer`.
- Two HTTP/HTTPS layer 7 ingress routing virtual servers. These virtual servers are attached to the Kubernetes Ingress Controller cluster load balancer service and can be manually scaled. Tanzu Kubernetes Grid Integrated Edition uses Kubernetes custom resources to monitor the state of the NSX-T load balancer service and scale the virtual servers created for ingress.

For information about configuring the layer 7 ingress controller see [Defining Network Profiles for the HTTP/S Layer 7 Ingress Controller](#), below. For information about configuring layer 7 ingress routing load balancers see [Scaling the HTTP/S Layer 7 Ingress Load Balancers Using the LoadBalancer CRD](#).

For information about configuring TCP layer 4 ingress routing load balancers see [Defining Network Profiles for the TCP Layer 4 Load Balancer](#).

For more information about the NSX-T Load Balancer, see [Create an IP Pool in Manager Mode](#) or [Add an IP Address Pool](#) in the VMware documentation.

For more information about Kubernetes custom resources, see [Custom resources](#) in the Kubernetes documentation.

## Configure the HTTP/HTTPS Ingress Controller Network Profile

The HTTP/HTTPS layer 7 virtual servers provisioned for each Kubernetes service are controlled by the parameters exposed in a network profile.

### NSX-T HTTP/HTTPS Ingress Controller Network Profile Configuration

The NSX Ingress Controller is configured using the `ncp.ini` network profile configuration file.

The HTTP/HTTPS Ingress Controller network profile has the following format:

```
{
 "name": "ncp_network_profile",
 "description": "DESCRIP",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "nsx_lb": NSX-LB,
 "ingress_ip": "IP-ADDRESS",
 "ingress_persistence_settings": {
 "persistence_type": "PERS-TYPE",
 "persistence_timeout": TIMEOUT
 }
 }
 }
 }
}
```

```

 }
}
```

Where:

- **DESCRIP** is your description for this network profile configuration.
- **NSX-LB** is your preference for whether the NSX-T Load Balancer is used for your Kubernetes clusters. For more information, see [Configure the NSX Ingress Controller](#) below.
- **IP-ADDRESS** is IP address to use for ingress controller load balancer. For more information, see [Configure the Ingress IP](#) below.
- **PERS-TYPE** is the persistence type to use for ingress controller load balancer. For more information, see [Configure the Ingress Persistence Settings](#) below.
- **TIMEOUT** is the persistence timeout to use for ingress controller load balancer. For more information, see [Configure the Ingress Persistence Settings](#) below.

For example:

```
{
 "name": "ncp_network_profile",
 "description": "Example network profile for ingress controller",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "nsx_lb": true,
 "ingress_ip": "192.168.160.212",
 "ingress_persistence_settings": {
 "persistence_type": "cookie",
 "persistence_timeout": 1
 }
 }
 }
 }
}
```

The following table describes the Ingress Controller configuration parameters:

| Parameter                       | Type            | Description                                                                                       |
|---------------------------------|-----------------|---------------------------------------------------------------------------------------------------|
| <code>name</code>               | String          | User-defined name of the network profile.                                                         |
| <code>description</code>        | String          | User-defined description for the network profile.                                                 |
| <code>parameters</code>         | Map             | Map containing one or more key-value pairs.                                                       |
| <code>cni_configurations</code> | Map             | Map containing <code>type</code> and <code>parameters</code> key-value pairs for configuring NCP. |
| <code>type</code>               | Constant String | Values: <code>"nsxt"</code> .                                                                     |
| <code>parameters</code>         | Map             | Map containing one or more key-value pairs for NCP settings.                                      |

|                                           |                             |                                                                                                                                                                                              |
|-------------------------------------------|-----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>nsx_lb</code>                       | Boolean<br><b>Updatable</b> | Use NSX-T layer 4 virtual server for each Kubernetes service of type LoadBalancer.<br>Values: <code>true</code> , <code>false</code> .<br>Default: <code>true</code> .                       |
| <code>nsx_ingress_controller</code>       | Boolean                     | Use NSX-T layer 7 virtual server as the ingress controller for the Kubernetes cluster.<br>Values: <code>true</code> , <code>false</code> .<br>Default: <code>true</code> .                   |
| <code>ingress_ip</code>                   | String                      | IP address to use for ingress controller load balancer.                                                                                                                                      |
| <code>ingress_persistence_settings</code> | String<br><b>Updatable</b>  | Map containing one or more key-value pairs for customizing Layer 7 persistence.<br>See also: <code>persistence_timeout</code> and <code>persistence_type</code>                              |
| <code>persistence_type</code>             | String<br><b>Updatable</b>  | An <code>ingress_persistence_settings</code> parameter. Specify the ingress persistence type.<br>Values: <code>"none"</code> , <code>"cookie"</code> , <code>"source_ip"</code> .            |
| <code>persistence_timeout</code>          | Integer<br><b>Updatable</b> | An <code>ingress_persistence_settings</code> parameter. Persistence timeout interval in seconds.<br>See also: <code>connect_retry_timeout</code> and <code>lb_http_response_timeout</code> . |

The `nsx_lb` parameter is used to control the TCP layer 4 virtual server that is provisioned for each Kubernetes service of type: [LoadBalancer](#).

When you configure an NSX Load Balancer as your Kubernetes cluster [ingress resource](#), NCP instructs the NSX-T Load Balancer to provision two layer 7 virtual services (HTTP and HTTPS) as the cluster [Ingress Controller](#):

| nsx_lb setting             | Description                                                                                           |
|----------------------------|-------------------------------------------------------------------------------------------------------|
| <code>nsx_lb: true</code>  | Use an NSX-T Layer 4 LoadBalancer and NCP-provisioned Layer 7 Ingress Controller.                     |
| <code>nsx_lb: false</code> | Use a third-party load balancer and a third-party ingress controller, such as <a href="#">NGINX</a> . |

## Configure the NSX Ingress Controller

NCP depends on the NSX-T Load Balancer to fulfill its role as an Ingress Controller. To use a third-party ingress controller, such as the [NGINX Ingress Controller](#), set `nsx_lb` to `false`.

For example:

- The following network profile uses the NSX-T Load Balancer and NSX Ingress Controller:

```
{
 "name": "example_network_profile",
 "description": "Use the NSX-T Load Balancer and NSX Ingress Controller",
 "parameters": {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "nsx_lb": true
 }
 }
 }
}
```

```
}
```

- The following network profile uses a third party load balancer and a third-party ingress controller:

```
{
 "name": "example_network_profile",
 "description": "Use a 3rd party load balancer and ingress controller",
 "parameters": {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "nsx_lb": false
 }
 }
 }
}
```

## Configure the Ingress IP

The `ingress_ip` parameter instructs NCP to create an ingress virtual server with the given IP address.

The `ingress_ip` parameter type is a string that accepts any valid IP address. Missing entry is accepted.

Example network profile for `ingress_ip`:

```
{
 "name": "example-network-profile",
 "description": "ingress_ip",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "ingress_ip": "192.168.160.212"
 }
 }
 }
}
```

An invalid IP address is rejected with an invalid parameter value error.

For example:

- The following network profile parameters cannot be parsed because the `"ingress_ip"` configuration specifies an invalid IP address:

```
{
 "name": "example-network-profile",
 "description": "ingress_ip-ERROR",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "ingress_ip": "192.168.460.212"
 }
 }
 }
}
```

```

 }
 }
}
```

- The following network profile cannot be parsed because the "ingress\_ip" configuration is not a string and the JSON input is invalid:

```
{
 "name": "example-network-profile",
 "description": "ingress_ip-ERROR",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "ingress_ip": 192.168.160.212
 }
 }
 }
}
```

## Configure the Ingress Persistence Settings

The `ingress_persistence` parameter lets you customize layer 7 persistence for Kubernetes services.

The `ingress_persistence_settings` parameter is a map that supports two keys:

- `persistence_type`
- `persistence_timeout`

These two keys are correlated and must be set/unset at the same time. If `persistence_type` and `persistence_timeout` are not both specified, the network profile fails validation.

| Parameter                        | Data Type | Description                                                                                      |
|----------------------------------|-----------|--------------------------------------------------------------------------------------------------|
| <code>persistence_type</code>    | String    | Valid values are <code>cookie</code> or <code>source_ip</code> . An empty value is not accepted. |
| <code>persistence_timeout</code> | Integer   | Value that is equal to 1 or larger. Empty value is not accepted.                                 |

For example:

- Network profile for `ingress_persistence_settings`:

```
{
 "name": "example_network_profile",
 "description": "ingress_persistence_settings",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "ingress_ip": "192.168.160.212"
 "ingress_persistence_settings": {
 "persistence_type": "cookie",
 "persistence_timeout": 1
 }
 }
 }
}
```

```

 }
 }
}

```

- Network profile for `ingress_persistence_settings`:

```

{
 "name": "example_network_profile",
 "description": "ingress_persistence_settings",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "ingress_ip": "192.168.160.212"
 "ingress_persistence_settings": {
 "persistence_type": "source_ip",
 "persistence_timeout": 100
 }
 }
 }
 }
}

```

## Defining DFW Section Markers

This topic describes how VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) administrators can define network profiles to create markers for NSX-T distributed firewall (DFW).



**Note:** The NSX-T Policy API features a tiered policy model using categories and does not support prioritizing operational rules using the procedure below.

## Overview

When a TKGI administrator creates a Network Profile or a Kubernetes developer creates a Kubernetes [Network Policy](#), NCP translates their configuration into a distributed firewall rule.

The first DFW rules have precedence over later DFW rules. When using the default DFW configuration, a TKGI administrator cannot define a Network Profile with precedence over existing DFW rules, including rules created from a developer's Kubernetes Network Policy configuration.

To define a Network Profile with precedence over existing DFW rules, configure a DFW Firewall Section Marker.

To configure a DFW Firewall Section Marker:

- [Create a Top Firewall Section Marker](#)
- [Create a Bottom Firewall Section Marker](#)

For more information about DFW Firewall Section Marker rules in Network Profiles, see [About DFW Section Markers](#) below.



**Note:** The NSX-T Policy API feature does not support DFW Firewall Section Marker configuration because it uses a category-based tiered policy model. NCP inserts

Network Policy rules into the [Application](#) tier, the last and lowest tier, which is always preceded by other tiers, such as [Infrastructure](#) and [Environment](#).

## About DFW Section Markers

Distributed firewall rules are either [Allow](#) or [Deny/Drop](#) rules:

- DFW [Allow](#) rules are placed before [Deny/Drop](#) rules and take higher precedence than [Deny](#) rules.
- When there are multiple DFW [Allow](#) or [Deny/Drop](#) rules, precedence is defined by the order the rules were created:
  - The first [Allow](#) rules have precedence over later [Allow](#) rules.
  - The first [Deny](#) rules have precedence over later [Deny](#) rules.

Because the first DFW rules have precedence over later DFW rules, existing DFW rules, including rules created from a Kubernetes developer's Network Policy configuration, have precedence over the rules defined afterward by a TKGI administrator using Network Profiles.

The VMware NSX-T Data Center supports organizing DFW rules into sections and TKGI supports creating a **Top Firewall Section Marker** and a **Bottom Firewall Section Marker** within DFW rules. DFW rules placed within the Top Firewall Section or Bottom Firewall Section have precedence over the other NCP-created DFW rules.

With the Top and Bottom Firewall Sections, an administrator can position operational rules to have precedence over developer-created DFW rules.

For more information about Distributed Firewalls in VMware NSX, see [Firewall in Manager Mode](#) or [Distributed Firewall](#) for Policy API environments, both in the VMware NSX Data Center for vSphere documentation.



**Note:** NSX-T applies DFW rules on ESXi transport nodes to control east-west traffic. Edge firewall (EFW) rules run on Edge transport nodes and control north-south traffic.

## Create a Top Firewall Section Marker

Using the [top\\_firewall\\_section\\_marker](#), the operational rules can be created in the top section, NCP will create any rules when Kubernetes network policies are created always below this top section marker. Thus, the operational rules created will not be over-ridden by developers.

```
{
 "name": "Example network profile",
 "description": "Network profile to enable DFW section marking",
 "parameters": {
 "cni_configurations": [
 {
 "type": "nsxt",
 "parameters": {
 "top_firewall_section_marker": "section-id"
 }
 }
]
 }
}
```

```

 }
 }
}
```

You cannot modify DFW rule configuration on an existing cluster.

## Create a Bottom Firewall Section Marker

Drop/deny rules operate in the bottom section. You can define a `bottom_firewall_section_marker` so that drop/deny rules created by NCP do not displace existing rules.

```
{
 "name": "Example network profile",
 "description": "Network profile to enable DFW section marking",
 "parameters": {
 "cni_configurations": [
 {
 "type": "nsxt",
 "parameters": {
 "bottom_firewall_section_marker": "section-id"
 }
 }
]
 }
}
```

You cannot modify DFW rule configuration on an existing cluster.

## Configure NCP Logging

This topic describes how to define network profiles for logging NCP configurations.

## About Logging for NCP Configurations

VMware Tanzu Kubernetes Grid Integrated Edition provides network profile parameters for logging `ncp.ini` configurations.

## Parameters for NCP Logging

The parameter `cni_configurations` is a map with two keys: `type` and `parameters`. The following table shows the parameters for configuring NCP:

The following table describes the Ingress Controller configuration parameters:

| Parameter                       | Type   | Description                                                                                       |
|---------------------------------|--------|---------------------------------------------------------------------------------------------------|
| <code>name</code>               | String | User-defined name of the network profile.                                                         |
| <code>description</code>        | String | User-defined description for the network profile.                                                 |
| <code>parameters</code>         | Map    | Map containing one or more key-value pairs.                                                       |
| <code>cni_configurations</code> | Map    | Map containing <code>type</code> and <code>parameters</code> key-value pairs for configuring NCP. |

|                                  |                 |                                                                                                                    |
|----------------------------------|-----------------|--------------------------------------------------------------------------------------------------------------------|
| <code>type</code>                | Constant String | Values: "nsxt".                                                                                                    |
| <code>parameters</code>          | Map             | Map containing one or more key-value pairs for NCP settings.                                                       |
| <code>log_settings</code>        | Map             | Map containing one or more key-value pairs for configuring NCP logging.                                            |
| <code>log_level</code>           | String          | The logging level.<br>Values: "INFO", "WARNING", "DEBUG", "ERROR", "CRITICAL".                                     |
| <code>log_dropped_traffic</code> | Boolean         | Log dropped firewall traffic.<br>Values: <code>true</code> , <code>false</code> .<br>Default: <code>false</code> . |

The `nsx_lb` parameter is used to control the TCP layer 4 virtual server that is provisioned for each Kubernetes service of type: [LoadBalancer](#).

When you configure an NSX Load Balancer as your Kubernetes cluster [ingress resource](#), NCP instructs the NSX-T Load Balancer to provision two layer 7 virtual services (HTTP and HTTPS) as the cluster [Ingress Controller](#):

| <code>nsx_lb setting</code> | <code>Description</code>                                                                              |
|-----------------------------|-------------------------------------------------------------------------------------------------------|
| <code>nsx_lb: true</code>   | Use an NSX-T Layer 4 LoadBalancer and NCP-provisioned Layer 7 Ingress Controller.                     |
| <code>nsx_lb: false</code>  | Use a third-party load balancer and a third-party ingress controller, such as <a href="#">NGINX</a> . |

## Example Network Profile for NCP Logging

The following network profile is an example that illustrates the parameters exposed for NCP logging.

```
{
 "name": "ncp_network_profile",
 "description": "Example network profile for NCP logging and error handling",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "log_settings": {
 "log_level": "WARNING",
 "log_dropped_traffic": true
 }
 }
 }
 }
}
```

## Log Settings

The parameter `log_settings` is a map that supports two keys: `log_level` and `log_dropped_traffic`.

## Log Level

The `log_level` parameter type is a string. The `log_level` value is an enumeration that supports the following values:

- “INFO”
- “WARNING”
- “DEBUG”
- “ERROR”
- “CRITICAL”

Any other value results in an error.

The value is set for three `ncp.ini` keys: `coe.nsxlib_loglevel`, `coe.loglevel`, and `k8s.loglevel`. The default log levels for these keys are as follows:

| <code>ncp.ini</code> key         | Default log level |
|----------------------------------|-------------------|
| <code>coe.nsxlib_loglevel</code> | INFO              |
| <code>coe.loglevel</code>        | NONE              |
| <code>k8s.loglevel</code>        | NONE              |

## Log Dropped Traffic

The `log_dropped_traffic` type is a boolean: `true` or `false`. Any other type is rejected, such as “true”. Missing entry is accepted. Enabling this parameter is used in distributed firewall configurations to log the traffic for dropped rules.

Example network profile for logging:

```
{
 "name": "example-network-profile",
 "description": "log_settings",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "log_settings": {
 "log_level": "DEBUG",
 "log_dropped_traffic": true
 }
 }
 }
 }
}
```

## enable\_err\_crd

The `enable_err_crd` parameter provides a mechanism of reporting NSX backend errors to Kubernetes cluster using a [custom resource definition \(CRD\)](#).

The Kubernetes resource “NSXError” is defined to hold error information. For each kubernetes resource object that has NSX backend failures, one “NSXError” object is generated with error

information. There is a ‘common error object’ containing all cluster-wide errors.

The `enable_err_crd` data type is a boolean: `true` or `false`. Missing entry is accepted. If `enable_err_crd` is set to `true`, you define a CRD to handle the “NSXError” common error object.

Example: network profile for `enable_err_crd`:

```
{
 "name": "example_network_profile",
 "description": "enable_err_crd",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "enable_err_crd": true
 }
 }
 }
}
```

## Shared and Dedicated Tier-1 Router Topologies

This topic describes shared and dedicated Tier-1 router topologies for Tanzu Kubernetes Grid Integrated Edition Kubernetes clusters on vSphere with NSX-T.

Shared Tier-1 topology is the default. This topic also explains how to define a network profile that overrides this default, to specify Dedicated Tier-1 topology for Tanzu Kubernetes Grid Integrated Edition clusters.

### Shared Tier-1 Topology

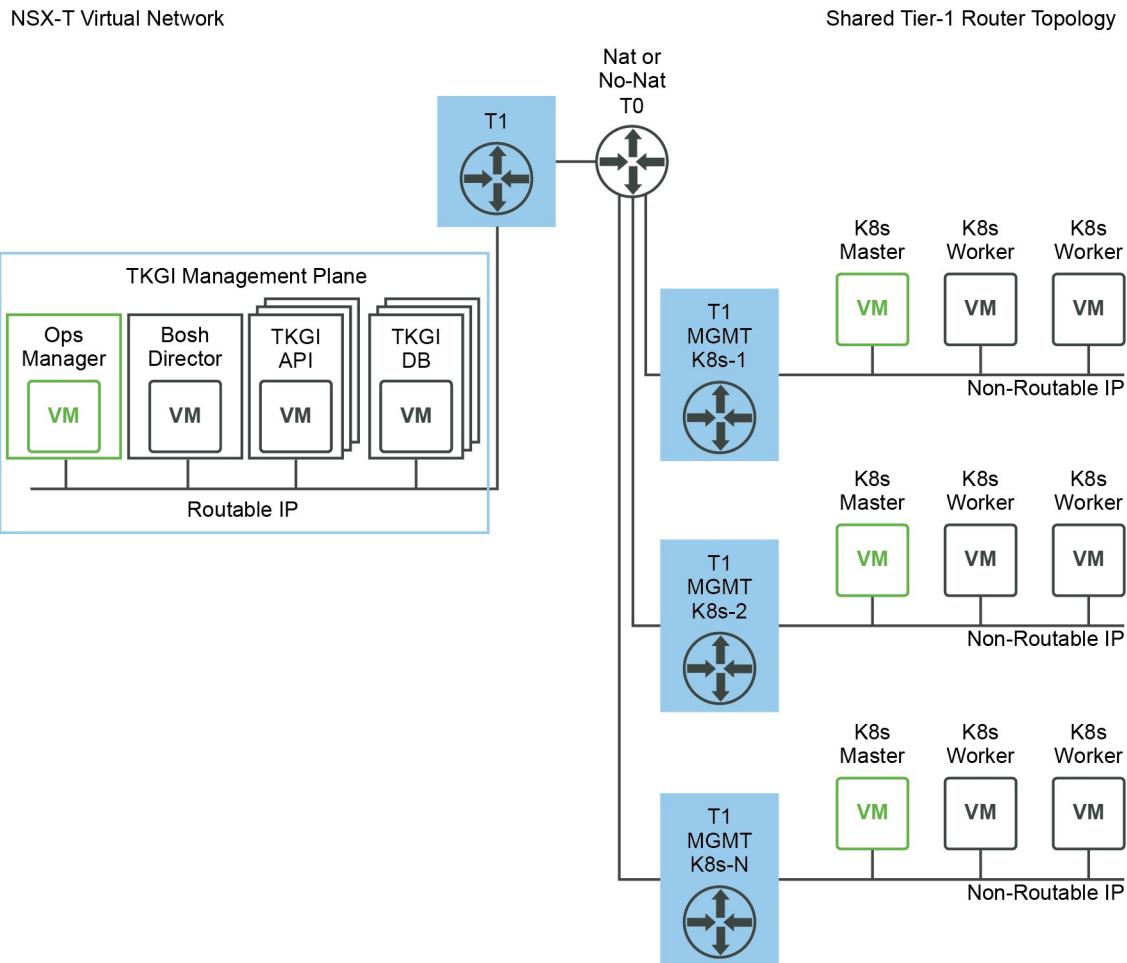
By default, Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition with NSX-T have the Shared Tier-1 network topology, in which each cluster shares a single Tier-1 router for its external-facing components: the Kubernetes node, namespace, and NSX-T load balancer.



**Note:** The Shared Tier-1 topology requires NSX-T Data Center v2.5.

This topology uses a single, shared Tier-1 switch and router for each Kubernetes cluster. The shared Tier-1 model only uses one Tier-1 router and multiple logical switches connected to the shared Tier-1 to connect all Kubernetes cluster components, including:

- Kubernetes Nodes Networks
- Kubernetes Namespaces
- NSX-T load balancer instances allocated for the Kubernetes cluster



## Comparison to Dedicated Tier-1

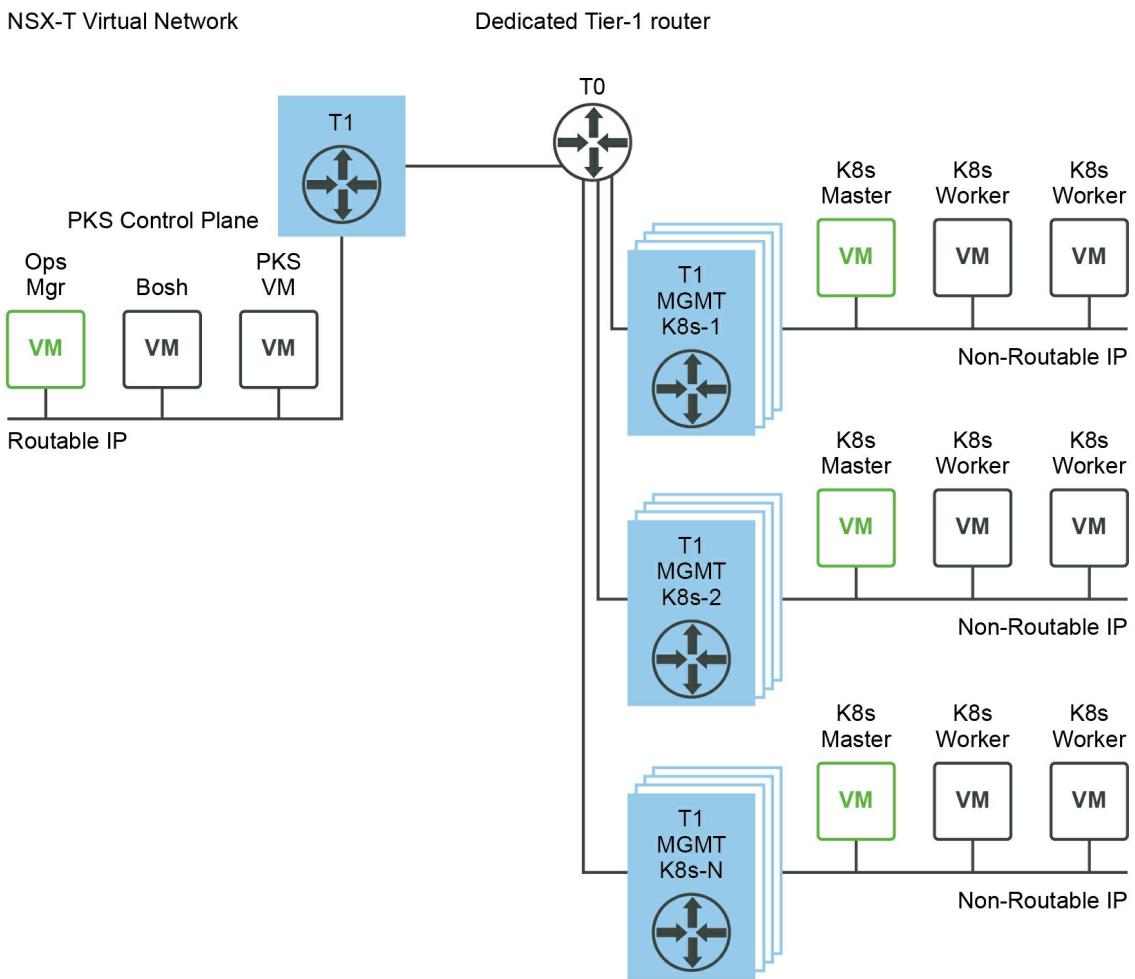
Unlike the [Dedicated Tier-1 Topology](#), the shared Tier-1 model configures any necessary NAT rules (if using NAT mode) on the single Tier-1 router directly. The Tier-0 router is not used for any NAT configuration. As a result, the Tier-0 router can operate in Active-Active mode if all Kubernetes clusters are deployed using the Shared Tier-1 model.

The Shared Tier-1 model enables higher scale numbers for TKGI as the number of NSX-T objects allocated per Kubernetes cluster is drastically reduced, in comparison to dedicated Tier-1. The advantage of the shared Tier-1 topology is that you can increase the number of NSX-T objects that can be supported in a given cluster.

## Dedicated Tier-1 Topology

When you provision a Kubernetes cluster with a network profile that overrides the Shared Tier-1 topology, Tanzu Kubernetes Grid Integrated Edition creates following NSX-T objects:

- 1 Logical Switch and Tier-1 Router for each Kubernetes Nodes subnet
- 1 Logical Switch and Tier-1 Router for each Kubernetes namespace
- 1 Logical Switch and Tier-1 Router each NSX-T Load Balancer that is allocated for the Kubernetes cluster



As depicted above, the result is that a given Kubernetes cluster will run several Tier-1 switches and routers in its topology.

## Network Profile for Dedicated Tier-1 Topology

To create clusters with Dedicated Tier-1 topology, you define and use a network profile that overrides the default Shared Tier-1 topology by setting the `single_tier_topology` key to `false`.

Shown below is an example network profile that deactivates the Shared Tier-1 Router for Kubernetes clusters:

```
{
 "name": "example-network-profile-shared-t1",
 "description": "Shared-Tier-1 topology network profile",
 "parameters": {
 "single_tier_topology": false
 }
}
```

To create a Shared Tier-1 network profile, see [Create Network Profile](#).

To create a cluster using a Shared Tier-1 network profile, see [Create a Cluster with a Network Profile](#).

## Implementing a Shared Tier-1 Topology in a Multi-Tier-0 Environment

In a Shared Tier-1 Router topology, all Kubernetes cluster traffic is automatically NATed in the single Tier-1 router that services that cluster. However, in a [Multi-Tier-0 environment](#), traffic from Kubernetes Node Networks to the Shared Tier-0 Router cannot be NATed.

To implement a Shared Tier-1 topology in a [Multi-Tier-0 environment](#), use the `infrastructure_networks` field in the network profile and include the subnets where your infrastructure is running. During Kubernetes cluster creation, Tanzu Kubernetes Grid Integrated Edition will add a NO\_SNAT rule from the Node Network to subnets specified in the `infrastructure_networks` field.

In the following example network profile, the `infrastructure-networks` field includes three subnets for which NO\_SNAT rules will be created. These subnets map to the PKS Control Plane (`30.0.0.0/24`), vCenter and NSX-T VMs (`192.168.111.0/24`), and the Nodes DNS server (`192.168.115.1`).

```
{
 "name": "tenant-A-shared-T1",
 "description": "Example Network Profile for Tenant A Shared Tier-1 Router Topology",
 "parameters": {
 "t0_router_id": "a6add27-24ce-469a-979e-cf742a19ef5c",
 "fip_pool_ids": [
 "a8b7f715-42f0-46bf-a4f2-1599c55058b6"],
 "pod_ip_block_ids": [
 "edd59bf6-ff04-420c-88de-2c43d47f7130"],
 "infrastructure_networks": [
 "30.0.0.0/24",
 "192.168.111.0/24",
 "192.168.115.1"
],
 "single_tier_topology": true
 }
}
```

## Compute Profiles and Host Groups (vSphere Only)

This section describes how to use compute profiles and host groups for VMware Tanzu Kubernetes Grid Integrated Edition clusters on vSphere.

See the following topics:

- [Creating and Managing Compute Profiles in the Management Console](#)
- [Creating and Managing Compute Profiles with the CLI \(vSphere\)](#)
- [Using Compute Profiles \(vSphere\)](#)
- [Using vSphere Host Groups with Tanzu Kubernetes Grid Integrated Edition](#)

## Creating and Managing Compute Profiles in the Management Console

You can add, view and remove compute profiles using the Tanzu Kubernetes Grid Integrated Edition Management Console on vSphere.

## About Compute Profiles

A compute profile enables cluster administrators, `pks.clusters.admin`, to override the default settings defined by a plan.

Using a compute profile, cluster administrators can customize the following:

- Compute resources such as CPU, memory, ephemeral disk, and persistent disk for Kubernetes control plane and worker nodes
- vSphere resources for Kubernetes control plane and worker nodes



**Note:** A compute profile overrides only those CPU, memory, disk, and AZ settings that you define in the profile. If you do not define a setting in the profile, its configuration is inherited from the plan.

After you create a compute profile, cluster managers, `pks.clusters.manage`, can apply it to one or more Kubernetes clusters.



**Note:** If you use vSphere without NSX-T networking, creating Windows clusters with compute profiles is not supported.

For more information, see [Compute Profiles vs. Plans](#).

## Create Cluster with Compute Profile

Use the Tanzu Kubernetes Grid Integrated Edition Management Console to create a cluster with an existing compute profile.

1. Select **TKG Integrated Edition > Clusters**, and select **Create Cluster**.
2. Use the **Compute Profile** drop-down menu to select the compute profile to use.
3. Click **Show More** to view the profile.

## Compute Profile

View the compute profile 'dev'

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |      |       |                 |                                                                                                                                                              |                      |   |                        |                                                                                                                                                                                                       |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|-------|-----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------|---|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name                   | dev                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |      |       |                 |                                                                                                                                                              |                      |   |                        |                                                                                                                                                                                                       |
| Description            | For development clusters                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |      |       |                 |                                                                                                                                                              |                      |   |                        |                                                                                                                                                                                                       |
| Availability Zones     | <div style="border: 1px solid #ccc; padding: 5px;"><p>&gt; z1</p><p>&gt; z2</p></div>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |      |       |                 |                                                                                                                                                              |                      |   |                        |                                                                                                                                                                                                       |
| Control plane AZs      | <div style="border: 1px solid #ccc; padding: 5px;"><p>z1</p><p>z2</p></div>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |      |       |                 |                                                                                                                                                              |                      |   |                        |                                                                                                                                                                                                       |
| Node Pool Groups       | <div style="border: 1px solid #ccc; padding: 5px;"><p>▼ node1</p><table><tr><td>Name</td><td>node1</td></tr><tr><td>Worker node AZs</td><td><div style="border: 1px solid #ccc; padding: 2px;">z1</div><div style="margin-top: 10px;"><div style="border: 1px solid #ccc; padding: 2px;">z2</div></div></td></tr><tr><td>Max worker instances</td><td>4</td></tr><tr><td>Labels of worker nodes</td><td><div style="border: 1px solid #ccc; border-radius: 15px; padding: 2px 10px; margin-right: 10px;">k1=v1</div><div style="border: 1px solid #ccc; border-radius: 15px; padding: 2px 10px;">k2=v2</div></td></tr></table></div> | Name | node1 | Worker node AZs | <div style="border: 1px solid #ccc; padding: 2px;">z1</div> <div style="margin-top: 10px;"><div style="border: 1px solid #ccc; padding: 2px;">z2</div></div> | Max worker instances | 4 | Labels of worker nodes | <div style="border: 1px solid #ccc; border-radius: 15px; padding: 2px 10px; margin-right: 10px;">k1=v1</div> <div style="border: 1px solid #ccc; border-radius: 15px; padding: 2px 10px;">k2=v2</div> |
| Name                   | node1                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |      |       |                 |                                                                                                                                                              |                      |   |                        |                                                                                                                                                                                                       |
| Worker node AZs        | <div style="border: 1px solid #ccc; padding: 2px;">z1</div> <div style="margin-top: 10px;"><div style="border: 1px solid #ccc; padding: 2px;">z2</div></div>                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |      |       |                 |                                                                                                                                                              |                      |   |                        |                                                                                                                                                                                                       |
| Max worker instances   | 4                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |      |       |                 |                                                                                                                                                              |                      |   |                        |                                                                                                                                                                                                       |
| Labels of worker nodes | <div style="border: 1px solid #ccc; border-radius: 15px; padding: 2px 10px; margin-right: 10px;">k1=v1</div> <div style="border: 1px solid #ccc; border-radius: 15px; padding: 2px 10px;">k2=v2</div>                                                                                                                                                                                                                                                                                                                                                                                                                                |      |       |                 |                                                                                                                                                              |                      |   |                        |                                                                                                                                                                                                       |

[View a larger version of this image](#)

4. Click **Modify Worker Nodes** to increase or reduce the number of worker nodes.

## >Create Cluster

Create a new Kubernetes cluster by first choosing a plan and then specify the rest of the attributes

| Plan                                                                                                                                                                                                                             | Select a plan ▾               |                |                        |       |                      |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------|----------------|------------------------|-------|----------------------|
| Name                                                                                                                                                                                                                             | <input type="text"/>          |                |                        |       |                      |
| Hostname                                                                                                                                                                                                                         | FQDN <input type="text"/>     |                |                        |       |                      |
| Worker Nodes                                                                                                                                                                                                                     | <input type="text"/>          |                |                        |       |                      |
| Network Profile                                                                                                                                                                                                                  | Select a Network Profile ▾    |                |                        |       |                      |
| Compute Profile                                                                                                                                                                                                                  | dev ▾                         |                |                        |       |                      |
| <a href="#">SHOW MORE</a> <a href="#">CANCEL MODIFY WORKER NODES</a>                                                                                                                                                             |                               |                |                        |       |                      |
| <table border="0"> <thead> <tr> <th style="width: 30%;">Node pool name</th> <th style="width: 70%;">Number of worker nodes</th> </tr> </thead> <tbody> <tr> <td>node1</td> <td><input type="text"/></td> </tr> </tbody> </table> |                               | Node pool name | Number of worker nodes | node1 | <input type="text"/> |
| Node pool name                                                                                                                                                                                                                   | Number of worker nodes        |                |                        |       |                      |
| node1                                                                                                                                                                                                                            | <input type="text"/>          |                |                        |       |                      |
| Kubernetes Profile                                                                                                                                                                                                               | Select a Kubernetes Profile ▾ |                |                        |       |                      |
| <input type="button" value="CREATE"/> <input type="button" value="CANCEL"/>                                                                                                                                                      |                               |                |                        |       |                      |

[View a larger version of this image](#)

## Define Compute Profile

Use the Tanzu Kubernetes Grid Integrated Edition Management Console to define a compute profile.

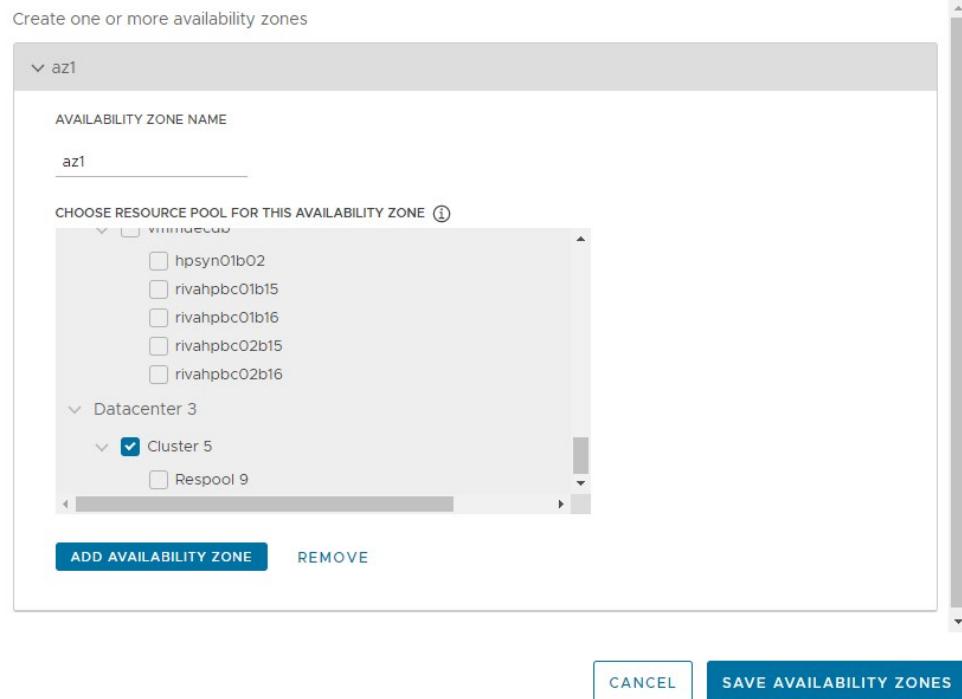


**NOTE:** You must be at the console home page to view the **Compute Profiles** tab.

1. Select **Profiles** and select the **Compute** tab.
2. Click **Create Profile**.
3. Enter a **Name** for the profile.
4. Enter a suitable **Description** for the profile.
5. Click **Add Availability Zones** to create a new availability zone for the compute profile.
  1. Enter a name for the availability zone.
  2. Enter the name of an existing datacenter, cluster, resource pool, and host group.

3. Click **Add Cluster** to add more clusters to the availability zone.
4. Click **Add Datacenter** to add more datacenters to the availability zone.
5. Click **Add Availability Zones** to create more availability zones.
6. Click **Save Availability Zones**.

### Add Availability Zones



[View a larger version of this image](#)

6. For **Control plane AZs**, select the availability zone to use for the cluster control plane.
7. Configure the new profile as needed, to override the values set in plans for the resources that are allocated to control plane nodes:
  - Number of Control Plane Nodes
  - Control plane CPU
  - Control plane memory
  - Control plane ephemeral disk size
  - Control plane persistent disk size

The screenshot shows the 'New Compute Profile' page in the management console. The left sidebar includes 'TKG Integrated Edition', 'Quotas', and 'Profiles'. Under 'ADMINISTRATION', there are links for 'Identity Management', 'Configuration' (selected), 'Deployment Metadata', 'TKGI Configuration', 'TKGI Instance Upgrade', and 'TKGI Component Patch'. The main form has the following fields:

- Name:** my-compute-profile
- Description:** Compute profile for development
- Availability Zones:** az1 (with an 'EDIT AVAILABILITY ZONES' link)
- Control plane AZs:** az1 (with a dropdown menu showing 'Use default')
- Number of Control Plane Notes:** 3
- Control plane CPU:** 2
- Control plane memory:** 8 GB
- Control plane ephemeral disk size:** 10 GB
- Control plane persistent disk size:** 10 MB
- Node Pool Groups:** A section with a right-pointing arrow button.

At the bottom are 'SAVE PROFILE' and 'CANCEL' buttons.

[View a larger version of this image](#)

8. Select **Node Pool Groups** to optionally override the values set in plans for the resources that are allocated to worker nodes, and to optionally add labels and taints to the nodes.

The screenshot shows the 'Node Pool Groups' configuration screen. The form includes the following fields:

- Name:** nodepool1
- Worker node AZs:** Use default
- Number of worker nodes:** (empty input field)
- Max worker instances:** (empty input field)
- CPU:** (empty input field)
- Memory:** (empty input field) MB
- Ephemeral disk size:** (empty input field) MB
- Persistent disk size:** (empty input field) MB
- Labels (optional):** ADD NODE LABEL
- Taints (optional):** ADD NODE TAINT

At the bottom are 'ADD NODE POOL GROUP' and 'REMOVE' buttons, followed by 'SAVE PROFILE' and 'CANCEL' buttons.

[View a larger version of this image](#)

9. Click **Save Profile**.

## Delete Compute Profile

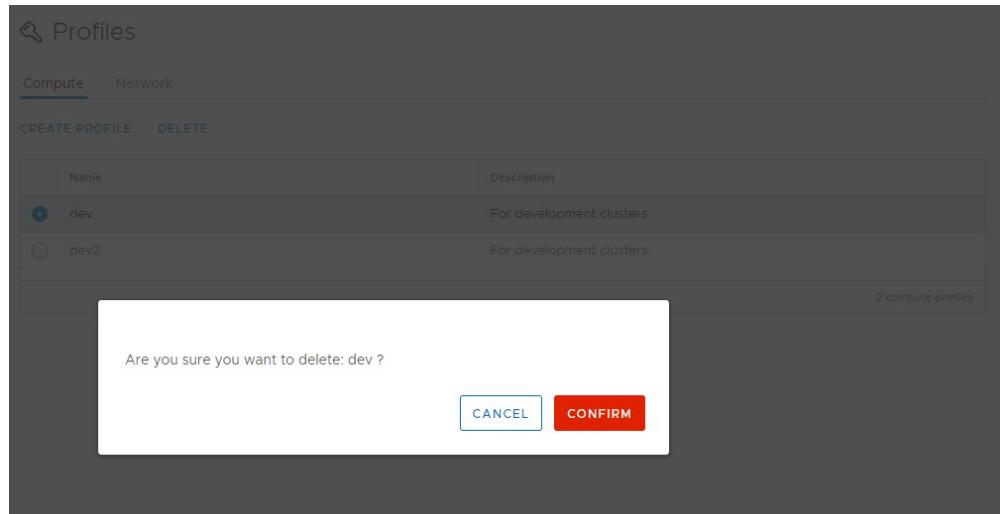
Use the Tanzu Kubernetes Grid Integrated Edition Management Console to delete

compute profile.



**NOTE:** You cannot delete a compute profile that is in use by a cluster.

1. Select **Profiles** and select the **Compute** tab.
2. Select the compute profile to remove.
3. Click **Delete**.
4. Confirm deletion.



[View a larger version of this image](#)

## Creating and Managing Compute Profiles with the CLI (vSphere)

This topic describes how to use the `tkgi` CLI to create and manage compute profiles for Linux- and Windows-based Kubernetes clusters on vSphere with NSX-T networking and for Linux-based Kubernetes clusters on vSphere without NSX-T networking.

For more information on how to use compute profiles, see [Using Compute Profiles \(vSphere\)](#).

TKGI versions prior to v1.9 used a different format for compute profiles, and these older profiles cannot be managed with the `tkgi` CLI. For information about pre-v1.9 compute profiles, see [Using Compute Profiles](#) in the TKGI v1.8 documentation.

## About Compute Profiles

A compute profile enables cluster administrators, `pks.clusters.admin`, to override the default settings defined by a plan.

Using a compute profile, cluster administrators can customize the following:

- Compute resources such as CPU, memory, ephemeral disk, and persistent disk for Kubernetes control plane and worker nodes
- vSphere resources for Kubernetes control plane and worker nodes



**Note:** A compute profile overrides only those CPU, memory, disk, and AZ settings that you define in the profile. If you do not define a setting in the profile, its configuration is inherited from the plan.

After you create a compute profile, cluster managers, `pks.clusters.manage`, can apply it to one or more Kubernetes clusters.

For more information, see [Compute Profiles vs. Plans](#) below.

## Create a Compute Profile

To create a compute profile in TKGI, a cluster administrator must:

1. Define a compute profile in a JSON configuration file. See [Compute Profile Format](#) and [Compute Profile Parameters](#) below.
2. Use the TKGI CLI to define the compute profile within TKGI. See [The create-compute-profile Command](#) below.

## Compute Profile Format

To create a compute profile, a cluster administrator first defines it as a JSON file. Every profile must include the `name`, `description`, and `parameters` properties. Then, depending on what compute resources you want to customize, define `azs`, `control_plane`, or `node_pools`. For example, you can define the following:

- Both `control_plane` and `node_pools`, with or without `azs`
- `control_plane` or `node_pools`, with or without `azs`

See the table below for examples of compute profiles.

| Example                              | Description                                                                      |
|--------------------------------------|----------------------------------------------------------------------------------|
| Custom Nodes                         | Define custom compute resources for Kubernetes control plane and worker nodes.   |
| Worker Node Pools                    | Define multiple pools of worker nodes.                                           |
| Custom AZs (vSphere with NSX-T Only) | Define AZs for Kubernetes control plane nodes and worker node pools dynamically. |

For information about all of the parameters that you can specify in a compute profile, see [Compute Profile Parameters](#) below.

### Custom Nodes

Cluster administrators can define custom compute resources for control plane nodes and/or worker nodes in a compute profile instead of updating plans. Then, cluster managers can apply the profile to an existing cluster to update the cluster with the new compute resources. As a result, the overall impact to the TKGI control plane and other clusters is smaller.

The example below defines compute resources for control plane nodes and one node

pool for workers:

```
{
 "name": "custom-nodes-compute-profile",
 "description": "custom-nodes-compute-profile",
 "parameters": {
 "cluster_customization": {
 "control_plane": {
 "cpu": 2,
 "memory_in_mb": 4096,
 "ephemeral_disk_in_mb": 16384,
 "persistent_disk_in_mb": 16384,
 "instances": 3
 },
 "node_pools": [
 {
 "cpu": 2,
 "memory_in_mb": 4096,
 "ephemeral_disk_in_mb": 16384,
 "persistent_disk_in_mb": 16384,
 "name": "tiny-1",
 "instances": 5,
 "max_worker_instances": 10
 }
]
 }
 }
}
```

## Worker Node Pools

Cluster administrators can define pools of worker nodes with different compute resources. Cluster managers then apply the compute profile to one or more clusters. This enables cluster managers to schedule workloads with different compute requirements on a single cluster.



**Warning:** If cluster update fails while applying a new compute profile with revised node pool names, do not reapply the previous compute profile: the cluster's worker nodes will be deleted. If you encounter this scenario, fix the new compute profile configuration without modifying the node pool names, and retry your cluster update. For more information, see [tkgi update-cluster compute profile failure](#) in the VMware Tanzu Knowledge Base.

The example below defines compute resources for control plane nodes and two worker node pools. The `control_plane` block in this example is optional.

```
{
 "name": "custom-node-pools-compute-profile",
 "description": "custom-node-pools-compute-profile",
 "parameters": {
 "cluster_customization": {
 "control_plane": {
 "cpu": 2,
 "memory_in_mb": 4096,
 "ephemeral_disk_in_mb": 16384,
 "persistent_disk_in_mb": 16384
 },
 "node_pools": [
 {
 "cpu": 2,
 "memory_in_mb": 4096,
 "ephemeral_disk_in_mb": 16384,
 "persistent_disk_in_mb": 16384,
 "name": "tiny-1",
 "instances": 5,
 "max_worker_instances": 10
 }
]
 }
 }
}
```

```

 "ephemeral_disk_in_mb": 16384,
 "instances": 3
 },
 "node_pools": [
 {
 "cpu": 2,
 "memory_in_mb": 4096,
 "ephemeral_disk_in_mb": 16384,
 "persistent_disk_in_mb": 16384,
 "name": "tiny-1",
 "instances": 5,
 "node_labels": "k1=v1,k2=v2",
 "node_taints": "k3=v3:PreferNoSchedule, k4=v4:PreferNoSchedule"
 },
 {
 "cpu": 4,
 "memory_in_mb": 4096,
 "ephemeral_disk_in_mb": 32768,
 "name": "medium-2",
 "instances": 1,
 "max_worker_instances": 5,
 "node_labels": "k1=v3,k2=v4",
 "node_taints": "k3=v3:NoSchedule, k4=v4:NoSchedule"
 }
]
}
}
}
}

```

## Custom AZs (vSphere with NSX-T Only)

Cluster administrators can define AZs in a compute profile instead of adding new AZs in the BOSH Director tile. Cluster managers then use it to specify AZs for a cluster dynamically. As a result, you do not need to make AZ changes to each TKGI plan and the overall impact to the TKGI control plane is smaller.



**Note:** You cannot use compute profiles to change the AZs of any nodes in an existing cluster. TKGI does not support changing the AZs of existing control plane nodes, but you can change the AZs of worker nodes by modifying their cluster's plan.

The example below defines three AZs, `cp-hg-az-1`, `cp-hg-az-2`, and `cp-hg-az-3`, in the `azs` block, which are then referenced in the `cluster_customization` block.

```
{
 "name": "azs-custom-compute-profile",
 "description": "Profile for customized AZs",
 "parameters": {
 "azs": [
 {
 "name": "cp-hg-az-1",
 "cpi": "ff8d93840299bd7474f5",
 "cloud_properties": {
 "datacenters": [
 {
 "name": "vSAN_Datacenter",
 "clusters": [
 {
 "vSAN Cluster": {

```

```

 "host_group": {
 "drs_rule": "MUST",
 "name": "CP-HG-AZ-1"
 }
 }
}
}
}

},
{
 "name": "cp-hg-az-2",
 "cpi": "ff8d93840299bd7474f5",
 "cloud_properties": {
 "datacenters": [
 {
 "name": "vSAN_Datacenter",
 "clusters": [
 {
 "vSAN_Cluster": {
 "host_group": {
 "drs_rule": "MUST",
 "name": "CP-HG-AZ-2"
 }
 }
 }
]
 }
]
 }
},
{
 "name": "cp-hg-az-3",
 "cpi": "ff8d93840299bd7474f5",
 "cloud_properties": {
 "datacenters": [
 {
 "name": "vSAN_Datacenter",
 "clusters": [
 {
 "vSAN_Cluster": {
 "host_group": {
 "drs_rule": "MUST",
 "name": "CP-HG-AZ-3"
 }
 }
 }
]
 }
]
 }
},
],
"cluster_customization": {
 "control_plane": {
 "cpu": 4,
 "memory_in_mb": 16384,
 "ephemeral_disk_in_mb": 32768,
 "az_names": ["cp-hg-az-1", "cp-hg-az-2", "cp-hg-az-3"],
 "instances": 3
 },
 "node_pools": [
 {
 "name": "x-large",
 "cpu": 4,
 "memory_in_mb": 8192,
 "ephemeral_disk_in_mb": 32768,
 "az_names": ["cp-hg-az-1", "cp-hg-az-2", "cp-hg-az-3"],
 "instances": 3
 }
]
}
]
```

```

 "max_worker_instances": 25,
 "node_labels": "k1=v1,k2=v2",
 "node_taints": "k3=v3:NoSchedule, k4=v4:NoSchedule"
]
}
}
}

```

## Compute Profile Parameters

The compute profile JSON configuration file includes the following top-level properties:

| Property                           | Type | Description                                                                                                                                                                        |
|------------------------------------|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>name</code>                  | S    | (Required) Name of the compute profile. You use this name when managing the compute profile or assigning the profile to a Kubernetes cluster through the TKGI CLI.                 |
| <code>description</code>           | S    | (Required) Description of the compute profile.                                                                                                                                     |
| <code>parameters</code>            | O    | (Required) Properties defining the main body of the compute profile such as <code>azs</code> and <code>cluster_customization</code> .                                              |
| <code>azs</code>                   | A    | (Optional) Properties defining one or more AZs, including the <code>name</code> , <code>cpi</code> , and <code>cloud_properties</code> settings. See <code>azs Block</code> below. |
| <code>cluster_customization</code> | O    | (Optional) Properties defining the <code>control_plane</code> and <code>node_pools</code> settings. See <code>control_plane Block</code> and <code>node_pools Block</code> below.  |

### `azs` Block (vSphere with NSX-T Only)

This optional block defines where Kubernetes control plane and worker nodes are created within your vSphere infrastructure. You can define one or more AZs.

If you define the `azs` block, do not specify the `persistent_disk_in_mb` property in `cluster_customization`. You can specify either `azs` or `persistent_disk_in_mb`, but not both.

Specify the properties below for each AZ. For more information about the `cloud_properties` schema, see [AZs](#) in the BOSH documentation.

| Property                      | Type | Description                                                                                                                                                                    |
|-------------------------------|------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>name</code>             | S    | Name for the AZ where you want to deploy Kubernetes cluster VMs.<br>For example, <code>cp-hg-az-1</code> .                                                                     |
| <code>cpi</code>              | S    | BOSH CPI ID of your TKGI deployment. For example, <code>abc012abc345abc567de</code> . For instructions on how to obtain the ID, see <a href="#">Retrieve the BOSH CPI ID</a> . |
| <code>cloud_properties</code> | O    | Properties defining vSphere <code>datacenters</code> for your Kubernetes cluster VMs.                                                                                          |
| <code>datacenters</code>      | A    | Array of datacenters. Define only one datacenter.                                                                                                                              |
| <code>name</code>             | S    | Name of your vSphere datacenter as it appears in Ops Manager and your cloud provider console. For example, <code>vSAN_Datacenter</code> .                                      |
| <code>clusters</code>         | A    | Array of clusters. Define only one cluster.                                                                                                                                    |
| <code>CLUSTER-NAME</code>     | S    | Name of your vSphere compute cluster. For example, <code>vSAN_Cluster</code> .<br>This section defines <code>host_group</code> .                                               |
| <code>host_group</code>       | O    | Properties of the host group that you want to use for your Kubernetes cluster VMs. This includes <code>name</code> and <code>drs_rule</code> .                                 |
| <code>name</code>             | S    | Name of the host group in vSphere.                                                                                                                                             |
| <code>drs_rule</code>         | S    | Specify <code>MUST</code> . If you use vSAN stretched clusters, specify <code>SHOULD</code> .                                                                                  |

## Retrieve the BOSH CPI ID

Use the following procedure to retrieve the BOSH CPI ID for your TKGI deployment.

1. Locate the credentials that were used to import the Ops Manager .ova or .ovf

file into your virtualization system. You configured these credentials when you installed Ops Manager.



**Note:** If you lose your credentials, you must shut down the Ops Manager VM in the vSphere UI and reset the password. See [vCenter Password Requirements and Lockout Behavior](#) in the vSphere documentation for more information.

2. From a command line, run the following command to SSH into the Ops Manager VM:

```
ssh ubuntu@OPS-MANAGER-FQDN
```

Where `OPS-MANAGER-FQDN` is the fully qualified domain name (FQDN) of Ops Manager.

3. When prompted, enter the password that you configured during the .ova deployment into vCenter.

For example:

```
$ ssh ubuntu@my-opsmanager-fqdn.example.com
Password: *****
```

4. Run `bosh cpi-config` to locate the Cloud Provider Interface (CPI) name for your deployment.

For example:

```
$ bosh cpi-config
Using environment 'BOSH-DIRECTOR-IP' as client 'ops_manager'
cpis:
- migrated_from:
 - name: ""
 name: YOUR-CPI-NAME
```

For more information about running BOSH commands in your Tanzu Kubernetes Grid Integrated Edition deployment, see [Using BOSH Diagnostic Commands in Tanzu Kubernetes Grid Integrated Edition](#).

### `control_plane` Block

This optional block defines properties for Kubernetes control plane node instances.

When defining the `control_plane` block, you must specify either `cpu`, `memory_in_mb`, and `ephemeral_disk_in_mb` or none of the three.

| Property | Ty | Description |
|----------|----|-------------|
|          | pe |             |

|                                    |    |                                                                                                                                                       |
|------------------------------------|----|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>cpu</code>                   | In | CPU count for control plane instances.                                                                                                                |
| <code>memory_in_mb</code>          | In | RAM for control plane instances.                                                                                                                      |
| <code>ephemeral_disk_in_mb</code>  | In | Ephemeral disk for control plane instances.                                                                                                           |
| <code>persistent_disk_in_mb</code> | In | Persistent disk for control plane instances. Do not specify this parameter if you intend to define the <code>azs</code> block in the compute profile. |
| <code>az_names</code>              | Ar | One or more AZs in which you want control plane instances to run. You defined these AZs in the <code>azs</code> block of the compute profile.         |
| <code>instances</code>             | In | Number of control plane instances. Specify <code>1</code> , <code>3</code> , or <code>5</code> .                                                      |

Do not assign the `name` property to this block.

### `node_pools` Block

This optional block defines properties for Kubernetes worker nodes. You can define one or more node pools.

When defining the `node_pools` block, you must specify either `cpu`, `memory_in_mb`, and `ephemeral_disk_in_mb` or none of the three.



**Note:** You must always configure at least one Node Pool group without a `node_taints` definition or with `PreferNoSchedule` taints. `kube-scheduler` will use this Node Pool group to schedule core Pods such as the `coredns` Pod. For more information about taints, see [Taints and Tolerations](#) in the Kubernetes documentation.

| Property          | T<br>y<br>p<br>e | Description            |
|-------------------|------------------|------------------------|
| <code>name</code> | S                | Name of the node pool. |

|                                    |    |                                                                                                                                                                                                             |
|------------------------------------|----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>cpu</code>                   | I  | CPU count for worker node instances.                                                                                                                                                                        |
|                                    | n  |                                                                                                                                                                                                             |
|                                    | t  |                                                                                                                                                                                                             |
|                                    | e  |                                                                                                                                                                                                             |
|                                    | g  |                                                                                                                                                                                                             |
|                                    | e  |                                                                                                                                                                                                             |
|                                    | r  |                                                                                                                                                                                                             |
| <code>memory_in_mb</code>          | I  | RAM for worker node instances.                                                                                                                                                                              |
|                                    | n  |                                                                                                                                                                                                             |
|                                    | t  |                                                                                                                                                                                                             |
|                                    | e  |                                                                                                                                                                                                             |
|                                    | g  |                                                                                                                                                                                                             |
|                                    | e  |                                                                                                                                                                                                             |
|                                    | r  |                                                                                                                                                                                                             |
| <code>ephemeral_disk_in_mb</code>  | I  | Ephemeral disk for worker node instances.                                                                                                                                                                   |
|                                    | n  |                                                                                                                                                                                                             |
|                                    | t  |                                                                                                                                                                                                             |
|                                    | e  |                                                                                                                                                                                                             |
|                                    | g  |                                                                                                                                                                                                             |
|                                    | e  |                                                                                                                                                                                                             |
|                                    | r  |                                                                                                                                                                                                             |
| <code>persistent_disk_in_mb</code> | I  | Persistent disk for worker node instances. Do not specify this parameter if you intend to define the <code>azs</code> block in the compute profile.                                                         |
|                                    | n  |                                                                                                                                                                                                             |
|                                    | t  |                                                                                                                                                                                                             |
|                                    | e  |                                                                                                                                                                                                             |
|                                    | g  |                                                                                                                                                                                                             |
|                                    | e  |                                                                                                                                                                                                             |
|                                    | r  |                                                                                                                                                                                                             |
| <code>az_names</code>              | A  | One or more AZs in which you want worker node instances to run.                                                                                                                                             |
|                                    | rr | You defined these AZs in the <code>azs</code> block of the compute profile.                                                                                                                                 |
|                                    | a  |                                                                                                                                                                                                             |
|                                    | y  |                                                                                                                                                                                                             |
| <code>instances</code>             | I  | Number of worker node instances.                                                                                                                                                                            |
|                                    | n  |                                                                                                                                                                                                             |
|                                    | t  |                                                                                                                                                                                                             |
|                                    | e  |                                                                                                                                                                                                             |
|                                    | g  |                                                                                                                                                                                                             |
|                                    | e  |                                                                                                                                                                                                             |
|                                    | r  |                                                                                                                                                                                                             |
| <code>max_worker_instances</code>  | I  | Maximum number of worker node instances for the node pool.                                                                                                                                                  |
|                                    | n  |                                                                                                                                                                                                             |
|                                    | t  |                                                                                                                                                                                                             |
|                                    | e  |                                                                                                                                                                                                             |
|                                    | g  |                                                                                                                                                                                                             |
|                                    | e  |                                                                                                                                                                                                             |
|                                    | r  |                                                                                                                                                                                                             |
| <code>node_labels</code>           | S  | One or more comma-delimited <code>key:value</code> pair labels you want to apply to worker node instances. For information on kubectl label syntax, see <a href="#">label</a> in the kubectl documentation. |
|                                    | tr |                                                                                                                                                                                                             |
|                                    | i  |                                                                                                                                                                                                             |
|                                    | n  |                                                                                                                                                                                                             |
|                                    | g  |                                                                                                                                                                                                             |

|                          |                                                                                                                                                                                                                                           |
|--------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>node_taints</code> | S One or more comma-delimited <code>key=value:effect</code> taints you want to apply to worker node instances. For information on <code>kubectl taint</code> syntax, see <a href="#">taint</a> in the <code>kubectl</code> documentation. |
| n<br>g                   |                                                                                                                                                                                                                                           |



**Warning:** If cluster update fails while applying a new compute profile with revised node pool names, do not reapply the previous compute profile: the cluster's worker nodes will be deleted. If you encounter this scenario, fix the new compute profile configuration without modifying the node pool names, and retry your cluster update. For more information, see [tkgi update-cluster compute profile failure](#) in the VMware Tanzu Knowledge Base.

## The create-compute-profile Command

After a compute profile is defined in a JSON file as described in [Compute Profile Format](#), a cluster administrator can create the compute profile by running the following TKGI CLI command:

```
tkgi create-compute-profile PATH-TO-YOUR-COMPUTE-PROFILE-CONFIGURATION
```

Where `PATH-TO-YOUR-COMPUTE-PROFILE-CONFIGURATION` is the path to the JSON file you created when defining the compute profile.

For example:

```
$ tkgi create-compute-profile dc-east-mixed.json
Compute profile dc-east-mixed successfully created
```

Only cluster administrators, `pks.clusters.admin`, can create compute profiles. If a cluster manager `pks.clusters.manage` or read-only admin `pks.clusters.admin-read-only` attempts to create a compute profile, the following error occurs:

You do not have enough privileges to perform this action. Please contact the TKGI administrator.

After an administrator creates a compute profile, cluster managers can create clusters with it or assign it to existing clusters. For more information, see the [Using Compute Profiles \(vSphere\)](#) topic.

## Manage Compute Profiles

TKGI administrators can delete compute profiles. Administrators can also perform the same operations that cluster managers use to list compute profiles and manage how clusters use them.



**Warning:** These commands do not work for compute profiles created

using the TKGI API in TKGI v1.8 or earlier.

## View a Compute Profile

To view details about a compute profile, run the following command:

```
tkgi compute-profile COMPUTE-PROFILE-NAME
```

Where `CPU-PROFILE-NAME` is the name of the compute profile you want to view.

For example:

```
tkgi compute-profile test-compute-profile

Name: test-compute-profile
Description: test-compute-profile
Parameters:
 Cluster Customization:
 Control Plane:
 Name:
 Instances: 3
 CPU: 2
 Memory (Mb): 4096
 Ephemeral Disk (Mb): 16384
 Node Pool:
 Name: tiny-1
 Instances: 5
 CPU: 2
 Memory (Mb): 4096
 Ephemeral Disk (Mb): 16384
 Node Pool:
 Name: medium-2
 Instances: 1
 CPU: 4
 Memory (Mb): 4096
 Ephemeral Disk (Mb): 32768
```

## Delete a Compute Profile

To delete a compute profile, run the following command:

```
tkgi delete-compute-profile COMPUTE-PROFILE-NAME
```

Where `CPU-PROFILE-NAME` is the name of the compute profile you want to delete.

For example:

```
tkgi delete-compute-profile test-compute-profile-8

Are you sure you want to delete the compute profile test-compute-profile-8?
(y/n): y
Deletion of test-compute-profile-8 completed
```

Limitations:

- You cannot delete a compute profile that is in use by a cluster.

- Only cluster administrators, `pks.clusters.admin`, can delete compute profiles. If a cluster manager `pks.clusters.manage` or read-only admin `pks.clusters.admin-read-only` attempts to delete a compute profile, the following error occurs:

You do not have enough privileges to perform this action. Please contact the TKGI administrator.

## Cluster Manager Operations

The following sections link to operations that both TKGI administrators and cluster managers can perform on compute profiles, documented in the [Using Compute Profiles \(vSphere\)](#) topic.

- [List Compute Profiles](#)
- [Create a Cluster with a Compute Profile](#)
- [Assign a Compute Profile to an Existing Cluster](#)
- [Resize a Cluster that Has an Existing Compute Profile](#)

## Compute Profiles vs. Plans

As with plans defined in TKGI tile **Plans** panes, compute profiles let TKGI administrators define cluster resource choices for developers using Kubernetes.

Compute profiles offer more granular control over cluster topology and node sizing than plans do. For example, compute profiles can define heterogenous clusters with different CPU, memory, ephemeral disk, or persistent disk settings for control plane nodes and worker nodes.

You can also apply a compute profile to specific clusters, overriding the default settings defined by their plan and possibly avoiding the need to create new plans.

You use the TKGI tile to manage plans and the TKGI CLI to manage compute profiles.

## Using Compute Profiles (vSphere)

This topic describes how to use compute profiles for Linux- and Windows-based Kubernetes clusters on vSphere with NSX-T networking and for Linux-based Kubernetes clusters on vSphere without NSX-T networking.

Compute profiles let you customize cluster resources parameters.

## How Compute Profiles are Created

TKGI cluster administrators can create and delete compute profiles, as described in the [Creating and Managing Compute Profiles with the CLI \(vSphere\)](#) topic.

After an administrator creates a compute profile, cluster managers

(`pks.cluster.manage`) can create clusters with it or assign it to existing clusters.

## List Compute Profiles

To list available compute profiles, run the following command:

```
tkgi compute-profiles
```

For example:

```
$ tkgi compute-profiles

Name Description
custom-nodes-compute-profile custom-nodes-compute-profile
custom-node-pools-compute-profile custom-node-pools-compute-profile
Dc-east-single-node-pool A profile for the east datacenter with a
 single node pool
dc-east-mixed A profile for the east datacenter with he
 terogeneous workers
```

## Create a Cluster with a Compute Profile

You can assign a compute profile to a Kubernetes cluster at the time of cluster creation.

To create an TKGI-provisioned Kubernetes cluster with a compute profile, run the following command:

```
tkgi create-cluster CLUSTER-NAME --external-hostname HOSTNAME --plan PLAN-NAME --compute-profile COMPUTE-PROFILE-NAME --node-pool-instances "NODE-POOL-NAME:INSTANCES"
```

Where:

- `CLUSTER-NAME` is a unique name for your cluster.



**Note:** Use only lowercase characters when naming your cluster if you manage your clusters with Tanzu Mission Control (TMC). Clusters with names that include an uppercase character cannot be attached to TMC.

- `HOSTNAME` is your external hostname used for accessing the Kubernetes API.
- `PLAN-NAME` is the name of the TKGI plan you want to use for your cluster.
- `COMPUTE-PROFILE-NAME` is the name of the compute profile you want to use for your cluster.
- (Optional) `--node-pool-instances` overrides the worker node instance counts

specified in the compute profile:

- `NODE-POOL-NAME` is the name of the node pool you want to specify the number of worker node instances for.
- `INSTANCES` is the number of worker nodes for the node pool name specified as `NODE-POOL-NAME`.

For example:

```
tkgi create-cluster custom-node-pools -e test.tkgi.shep.api.com --compute-profile custom-node-pools-compute-profile -p "small" --node-pool-instances "tiny-1:3"

TKGI Version: 1.9.0-build.11
Name: test
K8s Version: 1.18.5
Plan Name: small
UUID: <UUID of deployment>1
Last Action: CREATE
Last Action State: in progress
Last Action Description: Creating cluster
Kubernetes Master Host: test.tkgi.shep.api.com
Kubernetes Master Port: 8443
Worker Nodes: 4
Kubernetes Master IP(s): In Progress
Network Profile Name:
Kubernetes Profile Name:
Compute Profile Name: custom-node-pools-compute-profile
Tags:
```

## Assign a Compute Profile to an Existing Cluster

TKGI supports changing the compute profile for an already created cluster.

You can use this procedure to:

- Assign a compute profile to a cluster that does not have one.
- Change a cluster's existing profile to a new one.

To assign a compute profile to an existing cluster, complete the following:

1. [Compute Profile Update and Resize Validation](#).
2. [Assign a Compute Profile](#).

## Compute Profile Update and Resize Validation

Before assigning a compute profile to an existing cluster:

1. Review the following strict validation rules for the `tkgi update-cluster --compute-profile` command:

- You cannot use the compute profile to change Availability Zones of a cluster.
- If any optional field is unspecified at the time of cluster creation or cluster update, then the value of that field is set to the value of the corresponding field from the plan.
- Values passed to `--num-nodes` are ignored.
- Values passed to `--node-pool-instances` must match the labels of the node pools specified in the compute profile.
- Clusters created with a compute profile in TKGI 1.8 and earlier cannot be updated with a compute profile in TKGI v1.9 or later.

2. Review the following precautions and warnings:



**Warning:** Do not scale out or scale in existing control plane nodes by reconfiguring the TKGI tile or by using a compute profile. Reducing a cluster's number of control plane nodes might remove a control plane node and cause the cluster to become inactive.



**Warning:** If cluster update fails while applying a new compute profile with revised node pool names, do not reapply the previous compute profile: the cluster's worker nodes will be deleted. If you encounter this scenario, fix the new compute profile configuration without modifying the node pool names, and retry your cluster update. For more information, see [tkgi update-cluster compute profile failure](#) in the VMware Tanzu Knowledge Base.



**Note:** Compute profiles do not support moving a cluster's nodes to different Availability Zones.

3. Resolve any conditions matching the listed concerns, precautions or warnings.

## Assign a Compute Profile

To assign a compute profile to an existing cluster:

1. Select the compute profile to apply to the cluster:
  - Choose an existing compute profile: See [List Compute Profiles](#).
  - Create a new compute profile: Have a TKGI cluster administrator define and create a new compute profile as described in [Create a Compute Profile](#) in *Creating and Managing Compute Profiles with the CLI (vSphere)*.

- The name of the new compute profile must be unique and different from the previously assigned compute profile.
2. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
  3. Run the following command to update the cluster with the new compute profile:

```
tkgi update-cluster CLUSTER-NAME --compute-profile NEW-COMPUTE-PROFILE-NAME --node-pool-instances "NODE-POOL-NAME:INSTANCES"
```

Where:

- `CLUSTER-NAME` is the name of the existing Kubernetes cluster.
- `NEW-COMPUTE-PROFILE-NAME` is the name of the new compute profile you want to apply to the cluster.
- (Optional) `--node-pool-instances "NODE-POOL-NAME:INSTANCES"`. Use `--node-pool-instances` to update the number of worker nodes in a node pool. `NODE-POOL-NAME` is the name of the node pool in the new compute profile to be updated. `INSTANCES` is the new number of worker nodes in a node pool.

For example:

```
tkgi update-cluster test --compute-profile new-compute-profile

Update summary for cluster test:
Compute Profile Name: new-compute-profile
Are you sure you want to continue? (y/n): y
Use 'tkgi cluster test' to monitor the state of your cluster
```

## Resize a Cluster that Has an Existing Compute Profile

TKGI supports using the CLI to resize a cluster already created or assigned with a compute profile, without having to create a new compute profile.

To resize a cluster:

1. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
2. Run the `tkgi update-cluster` command with the `--node-pool-instances` option:
  - Pass in a comma-separated list that associates node pools defined in the compute profile with new instance counts. This changes the number of instances from each node pool.
  - The node pools must already be defined in the cluster's compute

profile.

For example, to resize a cluster `custom-node-pools-cluster` to use three nodes from the node pool named `tiny-1` and seven from the pool named `medium-2`:

```
tkgi update-cluster custom-node-pools-cluster --node-pool-instances "tiny-1:3,medium-2:7"
```

## Using vSphere Host Groups with Tanzu Kubernetes Grid Integrated Edition

### Topic provided by VMware

This topic describes how to use vSphere Host Groups with VMware Tanzu Kubernetes Grid Integrated Edition.

## About vSphere Host Groups

In vSphere, a cluster is a collection of ESXi servers that run virtual machines (VMs). A typical way to organize resources within a cluster is using resource pools. A resource pool is a collection of vSphere resources.

Another way to segment resources within a cluster is using host groups. This means that within a cluster object you can specify certain ESXi hosts to be part of a host group.

Tanzu Kubernetes Grid Integrated Edition users can define host groups in vSphere, then in the TKGI tile can specify the host group. Host groups align with the Availability Zone (AZ) construct in BOSH.

For more information on vSphere host groups, refer to the [vSphere documentation](#).

## Host Group Use Cases for Tanzu Kubernetes Grid Integrated Edition

This subsection describes use cases for using host groups with Tanzu Kubernetes Grid Integrated Edition.

### Enabling Support for vSAN Fault Domains

The vSAN fault domains feature instructs vSAN to spread redundancy components across the servers in separate computing racks. In this way, you can protect the environment from a rack-level failure such as loss of power or connectivity. For more information, see [Designing and Sizing vSAN Fault Domains](#) in the VMware documentation.

Fault domains map to host groups. If you have set up fault domains in your vSAN architecture, you can now leverage host groups with TKGI.

### Using Host Group as a New AZ in BOSH

Previously, the two types of AZs available with TKGI on vSphere were Datacenter and Datacenter plus Resource Pool. Host groups gives you a third option: Datacenter plus

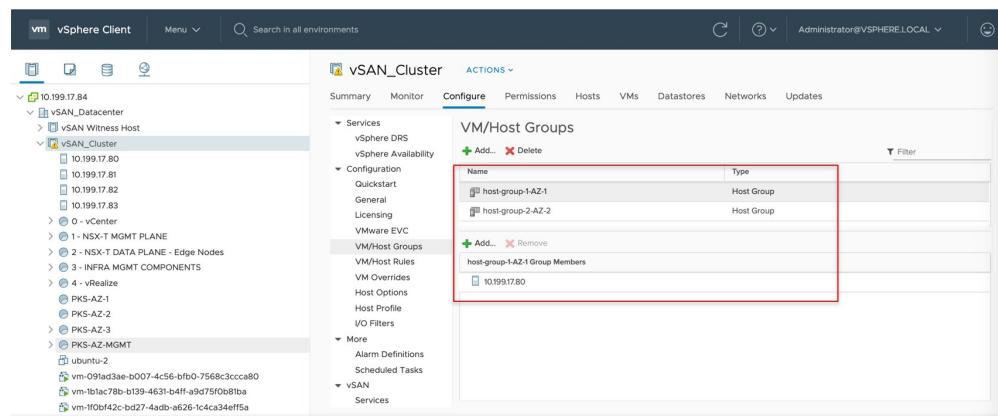
HostGroups.

In the case of multi-control plane node Kubernetes clusters, with the Datacenter and Datacenter plus Resource Pool AZs, there is no guarantee that control plane nodes will reside on separate ESXi hosts. With the Datacenter plus HostGroups AZ you can guarantee that Kubernetes control plane nodes will reside on separate ESXi hosts.

## Defining a Host Group in vSphere

To implement host groups with Tanzu Kubernetes Grid Integrated Edition, the first step is to define a host group in vSphere.

1. Log in to vCenter.
2. Select the compute **Cluster**.
3. Select the **Configure** tab.
4. Under **Configuration**, select **VM/Host Groups**.
5. Click **Add** and configure the host group as follows:
  - Name: Enter a name for the host group.
  - Type: Select **Host Group** from the drop down.
  - Click **Add** and select the ESXi host(s) to include in the host group.
  - Click **OK**.
6. Once done, you should see that the host group is configured.



## Using a Host Group with Tanzu Kubernetes Grid Integrated Edition

Once the host group is defined in vSphere, the next step is to declare this host group when defining the BOSH Availability Zone (AZ) for use with Tanzu Kubernetes Grid Integrated Edition.

1. Log in to Ops Manager.
2. Select the BOSH Director tile.
3. Select the **Create Availability Zones** tab.
4. Select the desired AZ, or create a new one.

5. In the **Clusters** section, enter the name of the **Host Group**.
6. (Optional) If you are using a host group with vSAN stretched clusters, set the **VM-Host Affinity Rule** dropdown to **SHOULD**. This setting maintains high availability by letting TKGI restart VMs in another host group if their AZ fails. TKGI ignores this setting if the vSAN cluster has no host group configured. For more information, see [Ability to Set the VM-Host Affinity Rule to “Should” for Clusters in vSphere] (<https://docs.pivotal.io/ops-manager/2-9/release-notes.html#vm-affinity-rule>) in the *Ops Manager v2.9 Release Notes*.
7. Click **Save**.

## Adding Infrastructure Password Changes to the Tanzu Kubernetes Grid Integrated Edition Tile

This topic describes how to manage VMware Tanzu Kubernetes Grid Integrated Edition after changing a BOSH Director or Tanzu Kubernetes Grid Integrated Edition service account password.

## Manage Your Service Account Passwords

When you installed Tanzu Kubernetes Grid Integrated Edition you created two service accounts:

- **BOSH/Ops Manager Service Account:** This service account is configured in the BOSH Director tile.
- **Master Node Service Account:** This service account is configured in the Tanzu Kubernetes Grid Integrated Edition tile.

You must update a tile’s copy of a service account password after changing the password on your network.

### Step 1: Update Your Service Account Passwords

To update BOSH Director with a new **BOSH/Ops Manager Service Account** password, perform the following steps:

1. Access the **Installation Dashboard** in Ops Manager.
2. Select the BOSH Director tile.
3. Select your IaaS' **Config** tab.
4. Click **Change**, the link beneath the IaaS **Password** field, to modify the password.

The screenshot shows the PCF Ops Manager interface with the 'BOSH Director for vSphere' configuration page. The 'vCenter Config' tab is active. On the left, a sidebar lists configuration steps: vCenter Config (selected), Director Config, Create Availability Zones, Create Networks, Assign AZs and Networks, Security, BOSH DNS Config, Syslog, and Resource Config. On the right, the 'vCenter Config' section includes fields for Name (vCenter-PA), vCenter Host (10.40.206.61), vCenter Username (administrator@vsphere.local), and vCenter Password\*. The 'vCenter Password\*' field is highlighted with a red border, and a tooltip indicates it is the password for the vCenter user specified above. A 'Change' link is visible below the password field.

5. Enter the new service account password.
6. Click **Save** to save the new password to the BOSH Director tile.

To update Tanzu Kubernetes Grid Integrated Edition with a new **Master Node Service Account** password, perform the following steps:

1. Access the **Installation Dashboard** in Ops Manager.
2. Select the Tanzu Kubernetes Grid Integrated Edition tile.
3. Select the **Kubernetes Cloud Provider** tab.
4. Click **Change**, the link beneath your IaaS' **Master Credentials** field, to modify the password.

vCenter Master Credentials \*

.....

Change

vCenter Host \*

Datacenter Name \*

Datastore Name \*

Stored VM Folder \*

Plan 4

Plan 5

Plan 6

Plan 7

Plan 8

Plan 9

Plan 10

Kubernetes Cloud Provider

Logging

Networking

UAA

Monitoring

Usage Data

AWS

Azure

Save

PCF Ops Manager v2.5.2-build.172; ©2013-2019 Pivotal Software, Inc; All Rights Reserved.

5. Enter the new control plane node service account password.
6. Click **Save** to save the new password to the Tanzu Kubernetes Grid Integrated Edition tile.

## Step 2: Deploy Your New Service Account Passwords

After updating an Ops Manager tile's service account password you must also deploy the new password.

To deploy a new password to BOSH Director and Tanzu Kubernetes Grid Integrated Edition, perform the following steps:

1. Access the **Installation Dashboard** in Ops Manager.
2. Click **Review Pending Changes**.
3. In the **Errands** section for Tanzu Kubernetes Grid Integrated Edition, select **Update all clusters errand**.
4. Click **Apply Changes** to update the Tanzu Kubernetes Grid Integrated Edition installation with the new password(s).



**Note:** The **Update all clusters errand** must be enabled to update the Kubernetes cloud provider password stored in Kubernetes clusters.

## Manage Your NSX Manager Password (vSphere and vSphere with NSX-T only)

If you are on vSphere or vSphere with NSX-T only, you also configured the **NSX Manager Account** and password when you installed Tanzu Kubernetes Grid Integrated Edition. This service account is configured in the BOSH Director tile.

After changing the password on your network, you must also update the BOSH Director tile's copy of the **NSX Manager Account** password.

To update the BOSH Director with the new NSX Manager password, perform the following steps:

1. Access the **Installation Dashboard** in Ops Manager.
2. Select the BOSH Director tile.
3. Select the **vCenter Config** tab.
4. Click **Change**, the link beneath the **NSX Username** field, to modify the password.

The screenshot shows the 'vCenter Config' tab in the BOSH Director tile. It includes fields for NSX Mode (set to NSX-T), NSX Address (10.40.206.5), NSX Username (admin), and NSX Password (which is currently set to '\*\*\*\*\*'). A 'Change' link is visible below the password field. The 'NSX CA Cert' field contains a large certificate block. Below the tile, there are fields for VM Folder (pks\_vms) and Template Folder, both of which are currently empty. A footer at the bottom of the tile displays the text: 'PCF Ops Manager v2.5.2-build.172; ©2013-2019 Pivotal Software, Inc; All Rights Reserved.'

5. Enter the new password.
6. Click **Save** to save the changes to the BOSH Director tile.
7. On the Ops Manager **Installation Dashboard**, select **Review Pending Changes**.
8. Click **Apply Changes**.

## Troubleshooting

### 'Failed to authenticate user' Error When Cluster Service Account Authenticates

#### Symptom

Your cluster control plane node does not authenticate with your vCenter even though the cluster's `/var/vcap/jobs/kube-controller-manager/config/cloud-provider.ini` file includes the correct vCenter credentials.

You see errors similar to the following in your logs:

- Service account errors in the TKGI logs:

```
error ... Failed to authenticate user ...
```

- Authentication errors in the BOSH tasks logs:

```
WARN -- [req_id ...]: Error running method 'Login'. Failed with message 'Cannot complete login due to an incorrect user name or password.'.
Rescued Unknown: Cannot complete login due to an incorrect user name or password..
backtrace: /var/vcap/data/packages/vsphere_cpi/.../lib/cloud/vsphere/retryer.rb:13:in `try'
```

Additionally, the cluster control plane repeatedly attempts to authenticate with vCenter, degrading the state of the STS service, and your vCenter becomes inaccessible.

## Solution

To resolve this issue:

1. Confirm that the correct credentials are included in your cluster's `/var/vcap/jobs/kube-controller-manager/config/cloud-provider.ini` file. If the credentials are incorrect, update the configuration with the valid credentials.
2. If the configured credentials are correct, review your cluster's actual vCenter authentication credentials:

```
bosh ssh -d DEPLOYMENT-NAME master/0 "sudo cat /var/vcap/jobs/csi-controller/config/csi-vsphere.conf"
```

Where `DEPLOYMENT-NAME` is the BOSH deployment name for the cluster that does not authenticate.

3. If the configured credentials and actual credentials are different, you must upgrade the cluster.

## Configuring VMware Tanzu Service Mesh by VMware NSX

This topic describes how to integrate VMware Tanzu Kubernetes Grid Integrated Edition with VMware Tanzu Service Mesh by VMware NSX.

Tanzu Service Mesh brings application-layer visibility, control, and security to microservices deployed on VMware Tanzu Kubernetes Grid Integrated Edition-managed Kubernetes clusters.

## About VMware Tanzu Service Mesh by VMware NSX

VMware Tanzu Service Mesh provides a service mesh solution for Kubernetes based on the NSX platform. Tanzu Service Mesh gives Kubernetes cluster users API-level visibility, control, and security over their clusters' services, data, and users.

In a Kubernetes cluster, Tanzu Service Mesh runs as a pod and is deployed using a YAML file.

For more information, see [NSX Service Mesh on VMware Tanzu: CONNECT & PROTECT Applications Across Your Kubernetes Clusters and Clouds](#) in the *VMware Network Virtualization* blog.

## Prerequisites

These instructions assume that:

- You have deployed VMware Tanzu Kubernetes Grid Integrated Edition.
- You have provisioned a target Kubernetes cluster for Tanzu Service Mesh.
- You have an account with VMware Cloud Services. If you do not already have an account, register as follows:
  1. Contact your Sales contact, or send an email to [driggs@vmware.com](mailto:driggs@vmware.com).
  2. Complete the registration process by following the emails you receive.

## Install VMware Tanzu Service Mesh in a Cluster

Install VMware Tanzu Service Mesh in a cluster as follows:

1. [Add the Tanzu Service Mesh Service](#)
2. [Onboard a Kubernetes Cluster to Tanzu Service Mesh](#)
3. [Install and Configure Istio](#)

### Add the Tanzu Service Mesh Service

1. Log in to the VMware Cloud Services console.
2. Select your organization or create a new one.
3. Select the Tanzu Service Mesh service offering and add your account to the service.

### Onboard a Kubernetes Cluster to Tanzu Service Mesh

Complete the following steps to install Tanzu Service Mesh onto a TKGI-provisioned Kubernetes cluster.

1. Sign in to the VMware Tanzu Service Mesh by VMware NSX console.
2. In the upper-left corner of the Tanzu Service Mesh Console, click **Add New > Onboard New Cluster** to open the **Onboard Clusters** panel. If this is the first cluster onboarded to Tanzu Service Mesh, the **Onboard Clusters** panel appears automatically when you finish signing up for Tanzu Service Mesh.
3. In the **Onboard Clusters** panel, enter a name for Tanzu Service Mesh to use to identify the target cluster.
  - VMware recommends that you enter the name of the cluster used in TKGI, but you can use a different name.
  - The cluster name must be unique within Tanzu Service Mesh.
4. Click **Generate Security Token** to generate a security token.

5. In the **Onboard Clusters** panel, click the copy icon to copy the `kubectl apply` command that applies the registration YAML file to the cluster.
6. Log in to your TKGI-provisioned Kubernetes cluster.
7. Run the `kubectl apply` command that you copied in a previous step to apply the registration YAML to the cluster. For example:

```
kubectl apply -f https://prod-1.servicemesh.biz/cluster-registration/k8s/v0.8.5
/k8s-registration.yaml
```

8. In the **Onboard Clusters** panel, click the copy icon to copy the `kubectl create secret` command that establishes a secure connection with NSX Service Mesh.
9. From your cluster, run the `kubectl create secret` command that you copied in the previous step. For example:

```
kubectl -n allspark create secret generic cluster-token --from-literal=token=eyJ
JhbGciOiJSUzI1NiIspInR5cCI6IkpxVCJ9.xxxxxxxxxxxxxxx
```

10. Click **Install NSX Service Mesh** button to install Tanzu Service Mesh on the cluster.
11. The YAML file deploys a pod to the target Kubernetes cluster that includes the Tanzu Service Mesh agent. If the target cluster is not discovered, click **Exit and Reload** and try again.

## Install and Configure Istio

Once the Tanzu Service Mesh agent is correctly started on a cluster:

1. Return to the Tanzu Service Mesh console and complete the on-boarding process by clicking on the **Install ISTIO** button in the on-boarding menu.

This operation installs the Istio components on the target cluster, including the Istio CNI plugin that lets Istio automatically inject its Envoy sidecar container whenever a new pod is started.

After you have onboarded clusters to Tanzu Service Mesh and installed Istio, they should appear in your Tanzu Service Mesh console.

## Known Issue: Timeout

When the `tkgi delete cluster` command is issued, the system runs an errand to clean up the pods currently running in the cluster. Istio installs a few pods that have a Pod Disruption Budget that conflict with the Tanzu Kubernetes Grid Integrated Edition cleanup errand. This might result in the errand running for an extended period of time.

Tanzu Kubernetes Grid Integrated Edition allows the user to select a timeout for Pod Disruption Budget, and the errand runs up to that timeout.

### Workaround

To avoid this problem, try to remove the on-boarded cluster as follows:

1. Log on to the Tanzu Service Mesh console and click on the name of cluster you want to remove.

2. Near the top right corner, click **Remove Cluster**.

- ◊ If this operation is successful, you can safely run the following command to delete the cluster:

```
tkgi delete-cluster
```

- ◊ If the operation is not successful, run the following command on the cluster before attempting to delete it with the `tkgi delete-cluster` command:

```
kubectl delete namespace istio-system
```

## Shutting Down and Restarting Tanzu Kubernetes Grid Integrated Edition

This topic lists and describes the shutdown and startup sequence for VMware Tanzu Kubernetes Grid Integrated Edition including Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes cluster nodes, TKGI components, and (vSphere only) vSphere hosts.

Many of these operations use your IaaS dashboard, such as vSphere Client, Azure Portal, AWS Management Console, or GCP Console.

## Shutdown Sequence and Tasks

To perform a graceful shutdown of all Kubernetes, Tanzu Kubernetes Grid Integrated Edition, and infrastructure components, complete the following tasks in sequence.

### Step 1: Deactivate BOSH Resurrection

If you have the **Enable VM Resurrector Plugin** checkbox selected in the BOSH Director tile > **Director Config** pane, you must turn BOSH resurrection off before restarting TKGI, to prevent BOSH from recreating VMs.

To do this, run the command `bosh update-resurrection off`.

### Step 2: Delete All PodDisruptionBudgets

To ensure that all workloads are drained as the worker nodes shutdown, remove all PodDisruptionBudgets before deleting your apps.

1. To confirm the names of your existing PodDisruptionBudgets:

```
kubectl get poddisruptionbudgets -A
```

2. Back up all PodDisruptionBudgets.

To back up a single PodDisruptionBudget:

```
kubectl get poddisruptionbudget PDB-NAME -o yaml > PDB-NAME.yaml
```

Where `PDB-NAME` is the name of one of your PodDisruptionBudgets.

3. Delete each PodDisruptionBudget until you have removed all PodDisruptionBudgets.

To delete a single PodDisruptionBudget:

```
kubectl edit poddisruptionbudget PDB-NAME
```

Where `PDB-NAME` is the name of one of your PodDisruptionBudgets.

## Step 3: Shut Down Customer Apps

Shut down all customer apps running on Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes clusters.



**Note:** This task is optional. Perform it after considering the types of apps you have deployed. For example, stateful, stateless, or legacy apps.

## Step 4: Shut Down Kubernetes Clusters

Shut down all Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes clusters following the procedure defined in the [How to shutdown and startup a Multi Control Plane Node TKGI cluster](#) knowledge base article.

For each Kubernetes cluster that you intend to shut down, do the following:

1. Using the BOSH CLI, retrieve the BOSH deployment name of your Tanzu Kubernetes Grid Integrated Edition clusters by running the following command:

```
bosh deployments
```

Kubernetes cluster deployment names begin with `service-instance_` and include a unique BOSH-generated hash.

2. Using the BOSH CLI, stop the Kubernetes worker nodes:

- ◊ For a Linux worker:

```
bosh -d service-instance_CLUSTER-UUID stop worker
```

- ◊ For a Windows worker:

```
bosh -d service-instance_CLUSTER-UUID stop windows-worker
```

Where `CLUSTER-UUID` is the BOSH deployment name of your Tanzu Kubernetes Grid Integrated Edition cluster.

For example:

```
$ bosh -d service-instance_aa1234567bc8de9f0alc stop worker
```



**Note:** When you use the BOSH `stop` command, all processes on the Kubernetes node are stopped. BOSH marks them stopped so that when the

VM is powered back on, the processes do not start automatically.

- Using the BOSH CLI, stop the Kubernetes control plane nodes by running the following command:

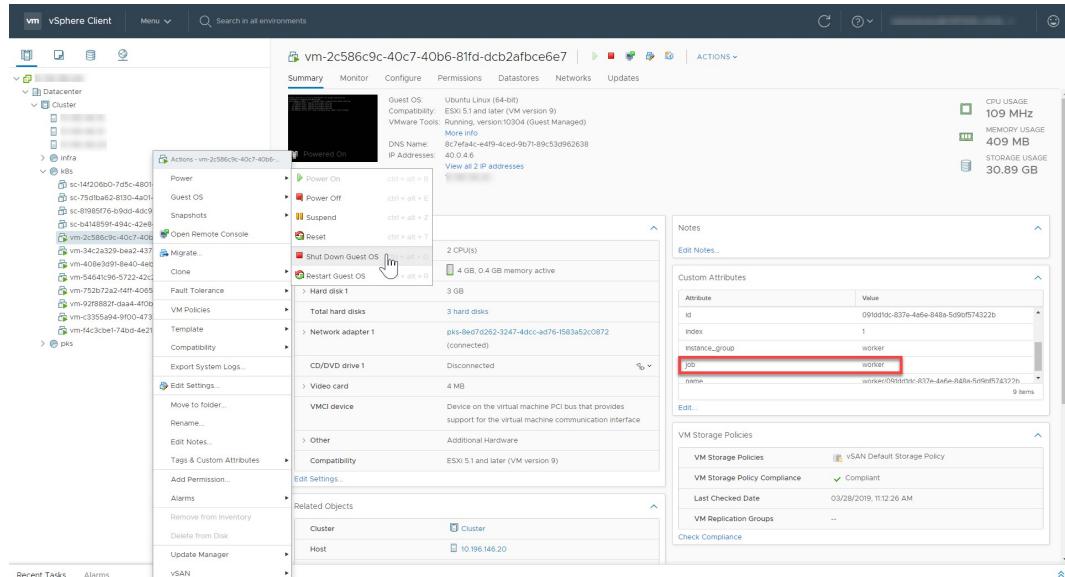
```
bosh -d service-instance_CLUSTER-UUID stop master
```

Where `CLUSTER-UUID` is the BOSH deployment name of your Tanzu Kubernetes Grid Integrated Edition cluster. For example:

```
$ bosh -d service-instance_aa1234567bc8de9f0alc stop master
```

- Using your IaaS dashboard, shut down all Kubernetes node VMs. To do this, perform the following steps:

- Verify the node type by checking the “job” name in the **Custom Attributes** pane.
- Perform a graceful shutdown by right-clicking the target VM and selecting **Power > Shut Down Guest OS**.



[View a larger version of this image.](#)

## Step 5: Stop the TKGI Control Plane

To shut down the TKGI control plane, stop and shut down the TKGI API and TKGI Database VMs as follows:

- Stop TKGI Control Plane Processes
- Shut Down the TKGI API and Database VMs

### Stop TKGI Control Plane Processes

To stop Tanzu Kubernetes Grid Integrated Edition control plane processes and services, do the following:

- Using the BOSH CLI, retrieve the BOSH deployment ID of your Tanzu Kubernetes Grid Integrated Edition deployment by running the following command:

```
bosh deployments
```

The Tanzu Kubernetes Grid Integrated Edition deployment ID is `pivotal-container-service-` followed by a unique BOSH-generated hash.

2. Stop the TKGI control plane VM by running the following command:

```
bosh -d pivotal-container-service-DEPLOYMENT-ID stop
```

Where `DEPLOYMENT-ID` is the BOSH-generated ID of your Tanzu Kubernetes Grid Integrated Edition deployment.

For example:

```
$ bosh -d pivotal-container-service-1bf7b02738056cdc37e6 stop
```

## Shut Down the TKGI API and Database VMs

To shut down the TKGI API and TKGI Database VMs, do the following:

1. Run the `bosh vms` command to list your Tanzu Kubernetes Grid Integrated Edition control plane VMs.

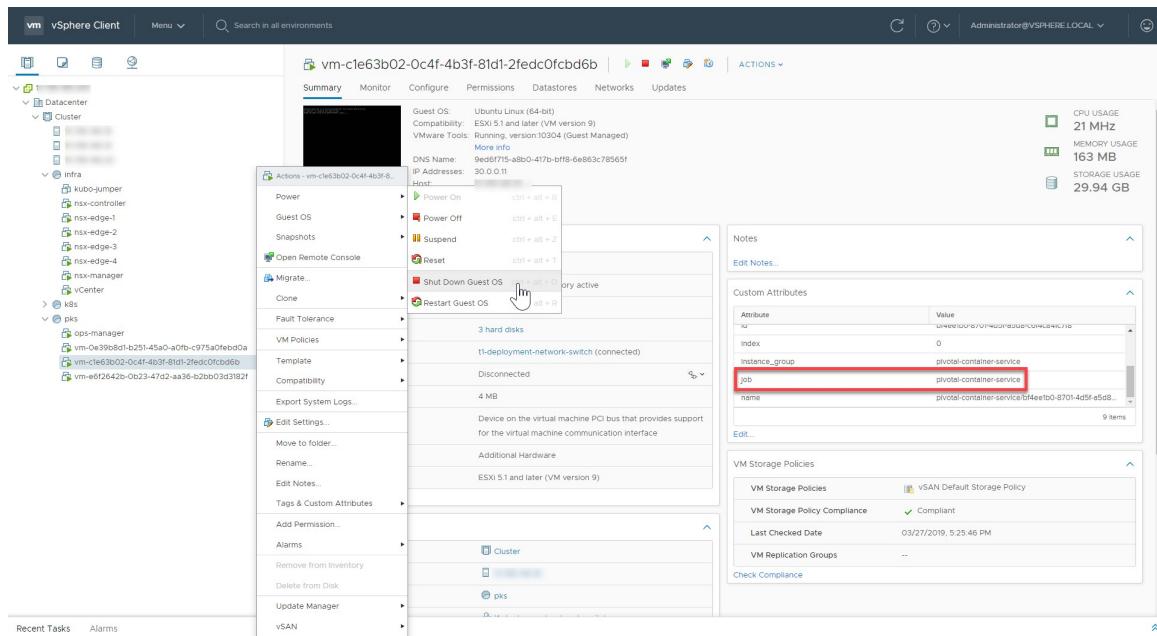
```
bosh -d pivotal-container-service-DEPLOYMENT-ID vms
```

Where `DEPLOYMENT-ID` is the BOSH-generated ID of your Tanzu Kubernetes Grid Integrated Edition deployment.

For example:

```
$ bosh -d pivotal-container-service-1bf7b02738056cdc37e6 vms
```

2. Review the `bosh vms` output:
  - Record the TKGI API VM name, listed under **Instances** as `pivotal-container-service/` followed by a unique BOSH-generated hash.
  - Record the TKGI Database VM name(s), listed under **Instances** as `pks-db/` followed by a unique BOSH-generated hash.
3. Using your IaaS dashboard, locate and gracefully shut down the TKGI control plane VMs:
  1. The TKGI API VMs.
  2. The TKGI Database VMs.



[View a larger version of this image.](#)

## Step 6: Shut Down VMware Harbor Registry (vSphere Only)

To shut down the Harbor Registry VM, do the following:

1. Using the BOSH CLI, retrieve the BOSH deployment ID of your Harbor Registry deployment by running the following command:

```
bosh deployments
```

Harbor Registry deployment names begin with `harbor-container-registry` and include a unique BOSH-generated hash.

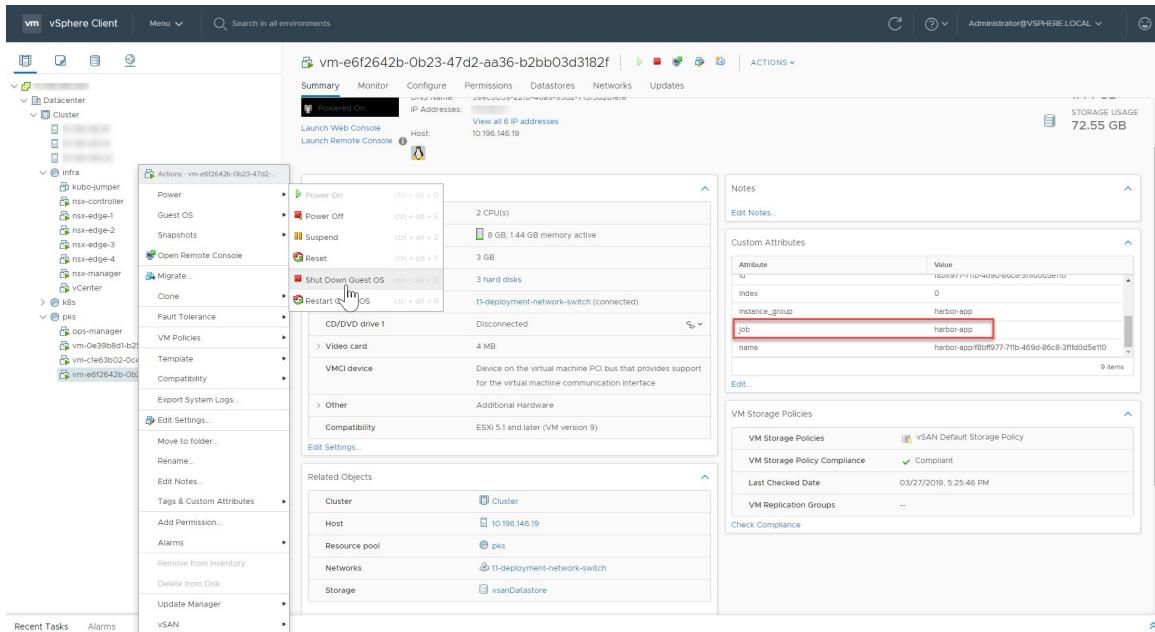
2. Using the BOSH CLI, stop the Harbor Registry VM by running the following command:

```
bosh -d harbor-container-registry-DEPLOYMENT-ID stop
```

Where `DEPLOYMENT-ID` is the BOSH-generated ID of your Harbor Registry deployment. For example:

```
$ bosh -d harbor-container-registry-b4023f6857207b237399 stop
```

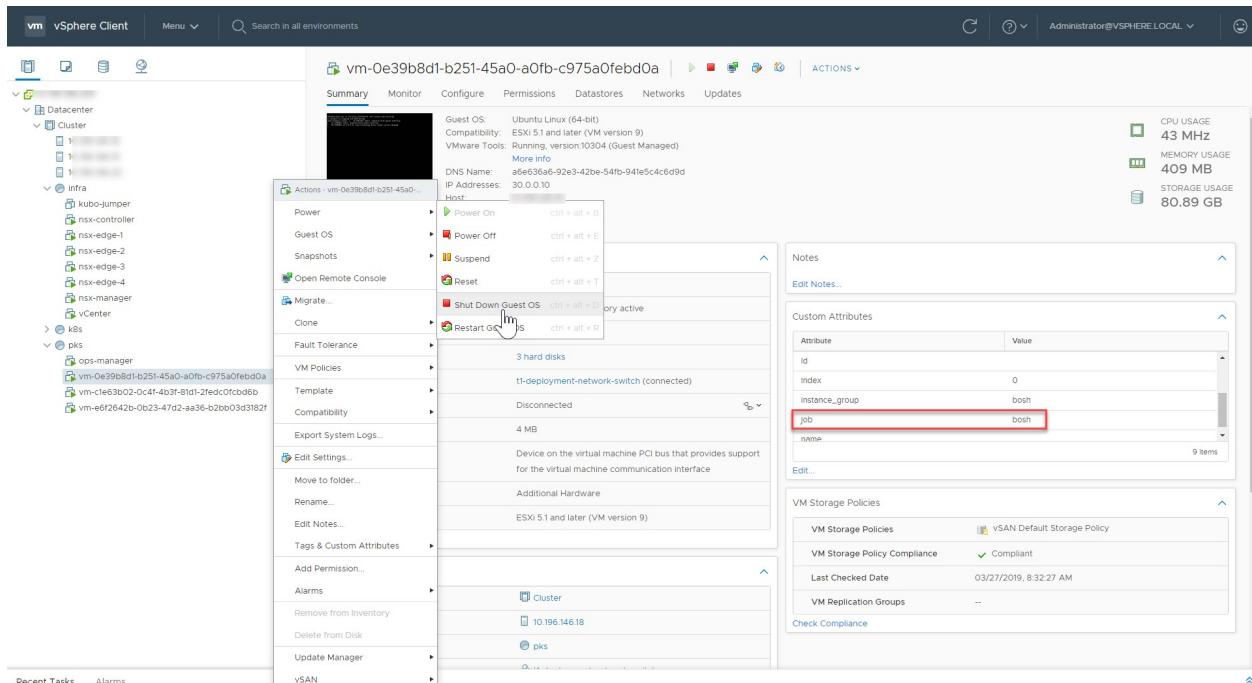
3. Using vCenter, locate and gracefully shut down the Harbor Registry VM.



[View a larger version of this image.](#)

## Step 7: Shut Down BOSH Director

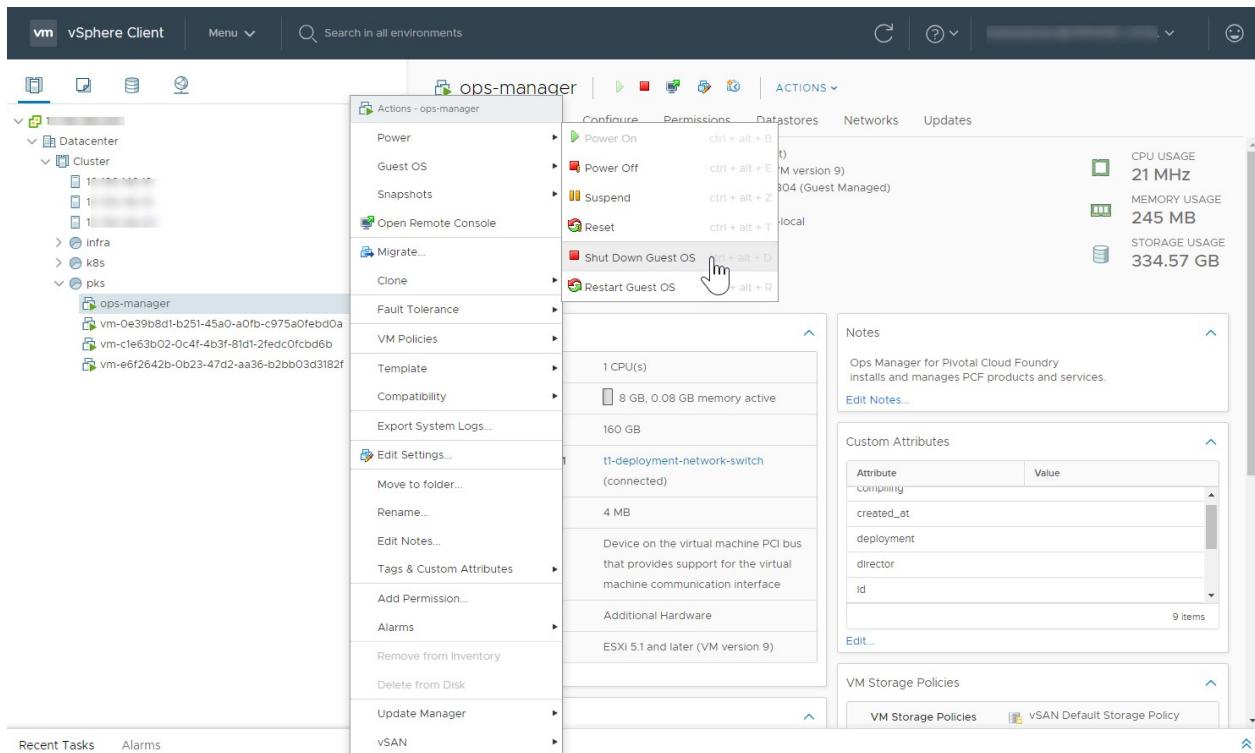
Using your IaaS dashboard, locate and gracefully shut down the BOSH Director VM.



[View a larger version of this image.](#)

## Step 8: Shut Down Ops Manager

Using your IaaS dashboard, locate and gracefully shut down the Ops Manager VM.

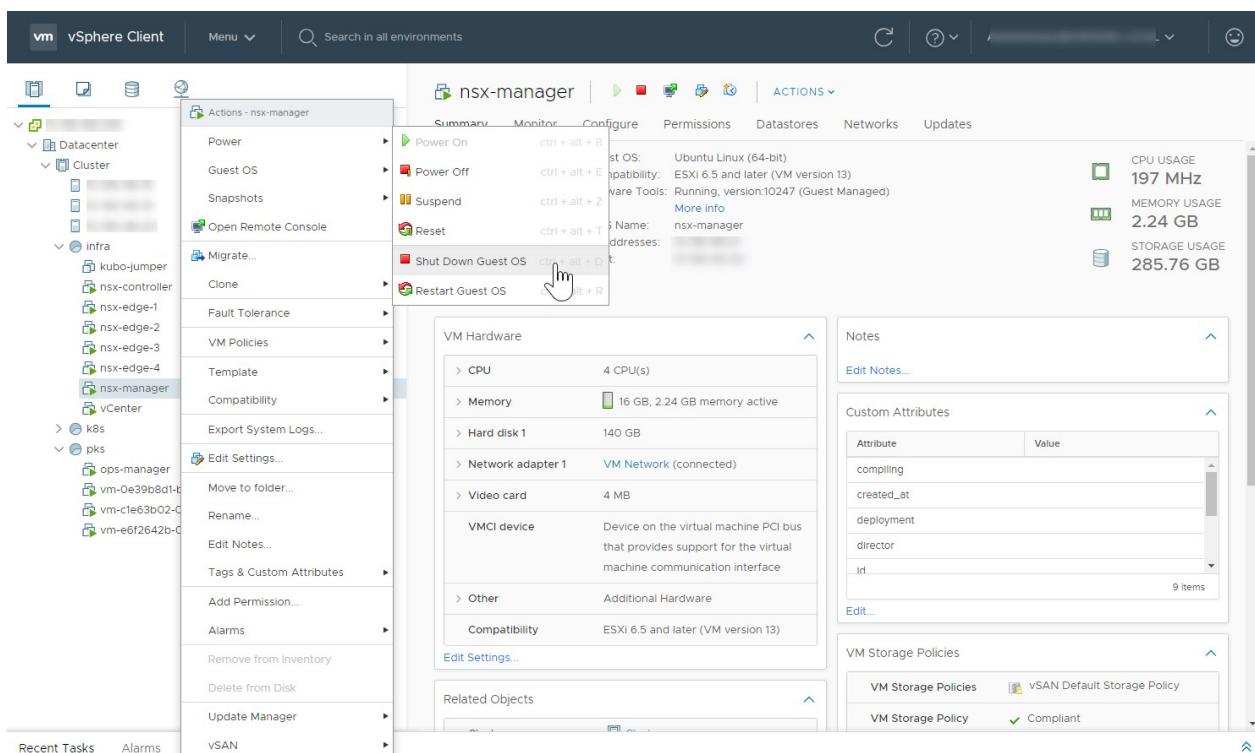


[View a larger version of this image.](#)

## Step 9: Shut Down NSX-T Components (vSphere NSX-T Only)

Using vCenter, gracefully shut down all NSX-T VMs in the following order:

1. NSX-T Manager
2. NSX-T Controllers
3. NSX-T Edge Nodes



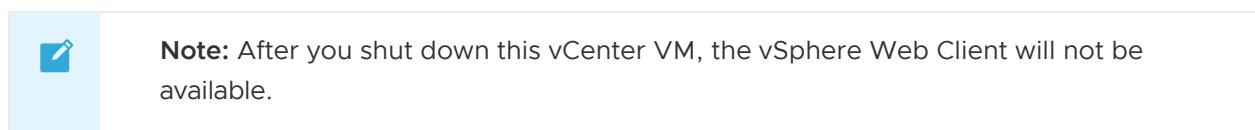
[View a larger version of this image.](#)

## Step 10: Shut Down vCenter Server (vSphere Only)

To shut down the vCenter Server VM, do the following:

1. Navigate to the vCenter Appliance Management Interface at <https://YOUR-VCENTER-HOSTNAME-OR-IP-ADDRESS:5480>, where `YOUR-VCENTER-HOSTNAME-OR-IP-ADDRESS` is the hostname or IP address that you use to connect to vCenter through the vSphere Web Client.
2. Log in as root.
3. Select **Actions > Shutdown** from the menu and confirm the operation.

For more information about how to shut down the vCenter Server VM, see [Reboot or Shut Down the vCenter Server Appliance](#) in the vSphere documentation and the [How to stop, start, or restart vCenter Server 6.x services](#) KB article.



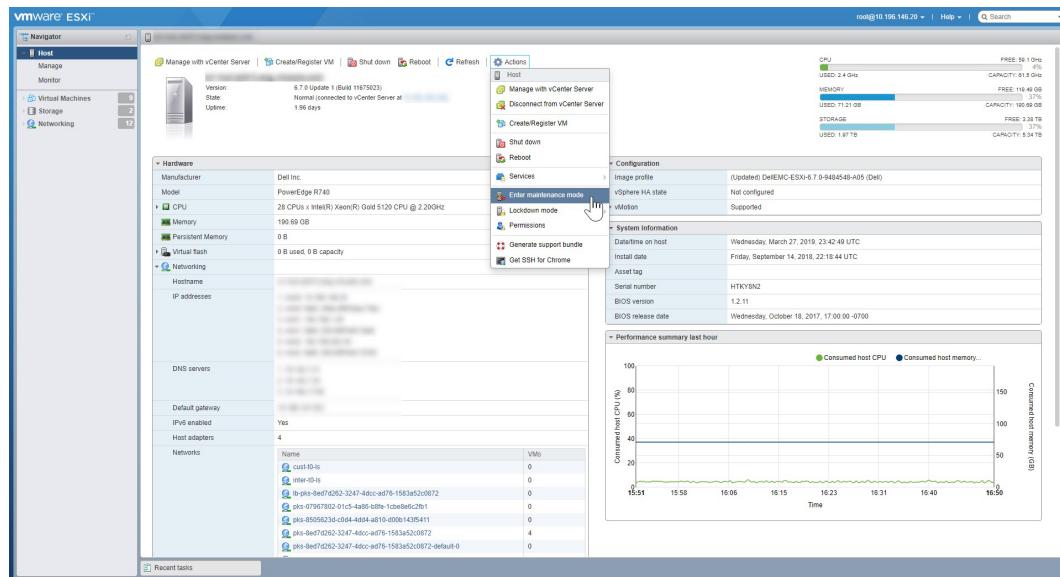
The screenshot shows the vCenter Appliance Management interface. On the left, there's a sidebar with various tabs like Summary, Monitor, Access, Networking, Firewall, Time, Services, Update, Administration, Syslog, and Backup. The main area displays the vCenter Server details: Hostname: photon-machine, Type: vCenter Server with an embedded Platform Services Controller, Product: VMware vCenter Server Appliance, Version: 6.7.0.21000, and Build number: 11726888. Below this is a Health Status table with rows for Overall Health, CPU, Memory, Database, Storage, and Swap, all showing Good status. To the right, there's a Single Sign-On table with Domain: vsphere.local and Status: Running. At the top right, there's a dropdown for English, Help, Actions (with Shutdown highlighted), and Logout. A note on the right says: "Note: After you shut down this vCenter VM, the vSphere Web Client will not be available."

[View a larger version of this image](#)

## Step 11: Shut Down ESXi Hosts (vSphere NSX-T Only)

To shut down each ESXi host in the vSphere cluster, do the following:

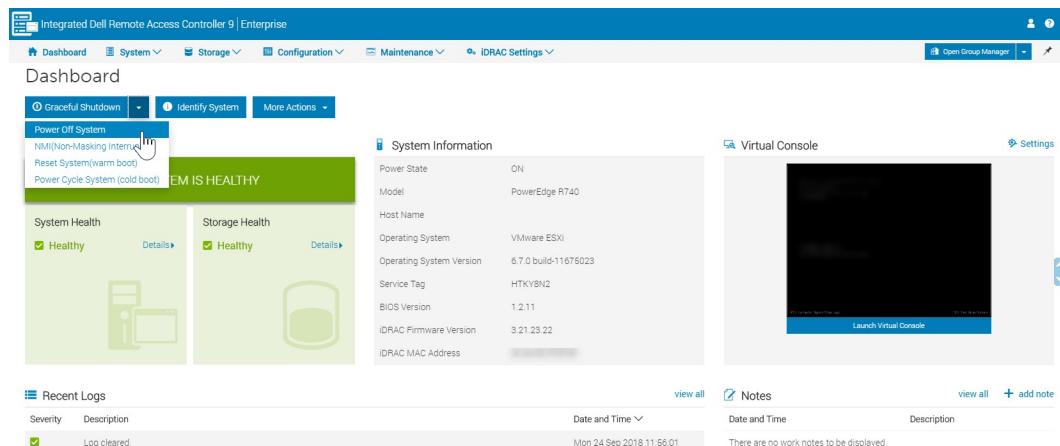
1. Put the ESXi host into maintenance mode by doing the following:
  1. Using a browser, navigate to the HTTPS IP address of the ESXi host, for example: <https://10.196.146.20/>.
  2. Log in using vSphere administrative credentials.
  3. Put the ESXi host in maintenance mode by selecting **Actions > Enter maintenance mode**.



[View a larger version of this image.](#)

- Power off the ESXi host. To do this, you have two options:

- Use the EXSi web interface and select **Actions > Shut down**.
- Use the remote management console for the host, such as Dell IDRAC or HP iLO.



[View a larger version of this image.](#)

## Startup Sequence and Tasks

To restart all Kubernetes, Tanzu Kubernetes Grid Integrated Edition, and infrastructure components, complete the following tasks in the sequence presented.

### Step 1: Start ESXi Hosts (vSphere NSX-T Only)

To start the ESXi hosts, do the following:

- Using the remote management console, such as Dell IDRAC or HP iLO, power on each ESXi host.
- Connect to the web interface of each ESXi host and exit maintenance mode.

## Step 2: Start vCenter (vSphere Only)

Connect to the web interface of the ESXi server that hosts the vCenter VM. Select the vCenter VM, and click **Power On**.

## Step 3: Start NSX-T Components (vSphere NSX-T Only)

To start the NSX-T components, perform the following steps:

1. Log into vCenter using the vSphere Client.
2. Power on the following VMs in the following order:
  1. NSX-T Manager
  2. NSX-T Controllers
  3. NSX-T Edge Nodes

## Step 4: Start Ops Manager

1. Using your IaaS dashboard, power on the Ops Manager VM.
2. Using a browser, go to the Ops Manager URL.
3. Enter the Ops Manager passphrase.
4. Log in to the Ops Manager UI.

## Step 5: Start the BOSH Director

Using your IaaS dashboard, power on the BOSH Director VM.



**Note:** BOSH is aware that all the VMs under its control were stopped. BOSH does not attempt to resurrect any VMs, which is the desired behavior.

It might take approximately 90 minutes for BOSH to start properly.

To speed up the BOSH startup process:

1. Obtain the BOSH Director VM Credentials from Ops Manager. For information about doing this, see [Retrieving Credentials from Your Deployment](#) in the Ops Manager documentation.
2. SSH to the BOSH Director VM.
3. On the BOSH Director VM, run the following commands:

```
sudo -i
monit summary
```

4. If you see messages such as `Process uaa Connection failed` and `Process credhub not monitored`, then run the following command:

```
monit restart uaa
```

5. After a few minutes, run the following command again:

```
monit summary
```

You should see that the `uaa` and `credhub` processes are now running. At this point, the BOSH Director should be fully up and running.

## Step 6: Start the TKGI Control Plane

To start the TKGI Control Plane, do the following:

1. Using your IaaS dashboard:
  1. Power on the TKGI Database VMs.
  2. Power on the TKGI API VMs.
2. Restart the TKGI Database deployments. The procedure to follow depends on whether the TKGI Database is scaled at `1` or `3`:
  - ◊ **TKGI Database Scaled at `1`:**  
To start the TKGI Database deployment:

```
bosh -d DEPLOYMENT-ID start pks-db
```

Where `DEPLOYMENT-ID` is the BOSH-generated ID of the Tanzu Kubernetes Grid Integrated Edition deployment.

- ◊ **TKGI Database Scaled at `3`:**

Run the `bootstrap` errand:

```
bosh -d tkgi-db-DEPLOYMENT-ID run-errand bootstrap
```

Where `DEPLOYMENT-ID` is the BOSH-generated ID of the Tanzu Kubernetes Grid Integrated Edition deployment.



**Note:** For more information about the `bootstrap` errand, see [Run the Bootstrap Errand](#) in the VMware Tanzu SQL with MySQL for VMs documentation.

For more information on TKGI Database scaling, see [Stop the TKGI Control Plane](#).

3. To restart the TKGI API deployment:

```
bosh -d DEPLOYMENT-ID start pivotal-container-service
```

Where `DEPLOYMENT-ID` is the BOSH-generated ID of the Tanzu Kubernetes Grid Integrated Edition deployment.

## Step 7: Start Harbor Registry (vSphere Only)

To start Harbor Registry, do the following:

1. Using vCenter, power on the Harbor VM.
2. Using the BOSH CLI, start the Harbor process on the VM by running the following command:

```
bosh -d harbor-container-registry-DEPLOYMENT-ID start
```

Where `DEPLOYMENT-ID` is the BOSH-generated ID of your Harbor Registry deployment. For example:

```
$ bosh -d harbor-container-registry-b4023f6857207b237399 start
```

## Step 8: Start the Kubernetes Clusters

For each Kubernetes cluster that you intend to start:

1. Using your IaaS dashboard, power on the cluster VMs.
2. Follow the procedure appropriate for the number of control plane nodes in the cluster:
  - ◊ [Start a Cluster with Three Control Plane Nodes](#)
  - ◊ [Start a Cluster with Five Control Plane Nodes](#)

### Start a Cluster with Three Control Plane Nodes

Use the BOSH CLI to run the commands below. For more information about the BOSH CLI, see the [BOSH CLI Documentation](#).

1. Retrieve the BOSH deployment name of your Tanzu Kubernetes Grid Integrated Edition cluster:

```
bosh deployments
```

Tanzu Kubernetes Grid Integrated Edition cluster deployment names begin with `service-instance_` followed by the BOSH deployment name.

2. Start etcd on the `master/0` node. A cluster with three members must have at least two members running to satisfy quorum, so you must first start one etcd instance.

```
bosh -d DEPLOYMENT-NAME ssh master/0 "sudo monit start etcd"
```

Where `DEPLOYMENT-NAME` is the BOSH deployment name retrieved in the previous step.

3. View the status of etcd on the `master/0` node:

```
bosh -d DEPLOYMENT-NAME ssh master/0 "sudo monit summary"
```

Where `DEPLOYMENT-NAME` is the BOSH deployment name retrieved in the previous step. Wait until etcd on `master/0` is running before executing the next step.

4. Start the `master/1` node:

```
bosh -d DEPLOYMENT-NAME start master/1
```

Where `DEPLOYMENT-NAME` is the BOSH deployment name retrieved in the previous step. Wait until `master/1` has status **Ready** before executing the next step.

5. Start the `master/2` node:

```
bosh -d DEPLOYMENT-NAME start master/2
```

Where `DEPLOYMENT-NAME` is the BOSH deployment name retrieved in the previous step. Wait until `master/2` has status **Ready** before executing the next step.

6. Stop etcd on the `master/0` node:

```
bosh -d DEPLOYMENT-NAME ssh master/0 "sudo monit stop etcd"
```

Wait until etcd on `master/0` has stopped and has status **not monitored** before executing the next step.

7. Start the `master/0` node:

```
bosh -d DEPLOYMENT-NAME start master/0
```

Wait until `master/0` has status **Ready** before executing the next step.

8. Start the Kubernetes worker nodes:

```
bosh -d DEPLOYMENT-NAME start worker
```

## Start a Cluster with Five Control Plane Nodes

Use the BOSH CLI to run the commands below. For more information about the BOSH CLI, see the [BOSH CLI Documentation](#).

1. Retrieve the BOSH deployment name of your Tanzu Kubernetes Grid Integrated Edition cluster:

```
bosh deployments
```

Tanzu Kubernetes Grid Integrated Edition cluster deployment names begin with `service-instance_` followed by the BOSH deployment name.

2. Start etcd on the `master/0` and `master/1` nodes. A cluster with five members must have at least three members running to satisfy quorum, so you must first start two etcd instance.

```
bosh -d DEPLOYMENT-NAME ssh master/0 "sudo monit start etcd"
bosh -d DEPLOYMENT-NAME ssh master/1 "sudo monit start etcd"
```

Where `DEPLOYMENT-NAME` is the BOSH deployment name retrieved in the previous step.

3. View the status of etcd on the `master/0` and `master/1` nodes:

```
bosh -d DEPLOYMENT-NAME ssh master/0 "sudo monit summary"
bosh -d DEPLOYMENT-NAME ssh master/1 "sudo monit summary"
```

Where `DEPLOYMENT-NAME` is the BOSH deployment name retrieved in the previous step. Wait until etcd on `master/0` and `master/1` is running before executing the next step.

4. Start the `master/2` node:

```
bosh -d DEPLOYMENT-NAME start master/2
```

Where `DEPLOYMENT-NAME` is the BOSH deployment name retrieved in the previous step. Wait until `master/2` has status **Ready** before executing the next step.

- Start the `master/3` node:

```
bosh -d DEPLOYMENT-NAME start master/3
```

Wait until `master/3` has status **Ready** before executing the next step.

- Start the `master/4` node:

```
bosh -d DEPLOYMENT-NAME start master/4
```

Wait until `master/4` has status **Ready** before executing the next step.

- Stop etcd on the `master/1` node:

```
bosh -d DEPLOYMENT-NAME ssh master/1 "sudo monit stop etcd"
```

Wait until etcd on `master/1` has stopped and has status **not monitored** before executing the next step.

- Start the `master/1` node:

```
bosh -d DEPLOYMENT-NAME start master/1
```

Wait until `master/1` has status **Ready** before executing the next step.

- Stop etcd on the `master/0` node:

```
bosh -d DEPLOYMENT-NAME ssh master/0 "sudo monit stop etcd"
```

Wait until etcd on `master/0` has stopped and has status **not monitored** before executing the next step.

- Start the `master/0` node:

```
bosh -d DEPLOYMENT-NAME start master/0
```

Wait until `master/0` has status **Ready** before executing the next step.

- Start the Kubernetes worker nodes:

```
bosh -d DEPLOYMENT-NAME start worker
```

## Step 9: Start Customer Apps

Start all apps running on the Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes clusters.

## Step 10: Restore All PodDisruptionBudgets

After you have restarted your apps, restore your PodDisruptionBudgets.

1. To restore your PodDisruptionBudgets, re-create each of the PodDisruptionBudgets you backed up before stopping your apps.  
To re-create a single PodDisruptionBudget:

```
kubectl apply -f PDB-CONFIG-NAME
```

Where `PDB-CONFIG-NAME` is the name of one of the backup PodDisruptionBudget YAML configuration files that you created before stopping your apps.

## Step 11: Re-enable BOSH Resurrection

Turn BOSH resurrection back on by running the command `bosh update-resurrection on`.

## Deleting Tanzu Kubernetes Grid Integrated Edition

This topic explains how to delete the Tanzu Kubernetes Grid Integrated Edition (TKGI) tile.

## Delete the Tanzu Kubernetes Grid Integrated Edition Tile

To delete the TKGI tile, perform the following steps:

1. Navigate to the Ops Manager Installation Dashboard.
2. Click the trash can icon on the TKGI tile.
3. Click **Confirm**.
4. Click **Review Pending Changes**.
5. (Optional) To preserve clusters created by TKGI, click **Errands** and deactivate the **Delete all clusters errand** checkbox under **Pre-Delete Errands**.
  - By default, the **Delete all clusters** errand is activated, which deletes all TKGI clusters before Ops Manager deletes the TKGI tile.
6. Click **Apply Changes**.

# Managing Kubernetes Clusters and Workloads

This section describes how to manage VMware Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes clusters and workloads.

See the following topics:

- [Create and Manage Clusters in the Management Console](#)
- [Managing Clusters with the CLI](#)
- [Supporting Clusters with the CLI](#)
- [Deploying Workloads with the CLI](#)
- [Load Balancing and Ingress](#)
- [Load Balancing and Ingress with NSX-T](#)

## Create and Manage Clusters in the Management Console

This section describes how to create clusters in the VMware Tanzu Kubernetes Grid Integrated Edition Management Console on vSphere. Cluster creation in the management console involves the following actions:

- Add users and user groups to VMware Tanzu Kubernetes Grid Integrated Edition, and assign roles to them so that they can create and manage clusters. For information about user management, see [Identity Management in the Management Console](#).
- Set resource quotas to limit the amount of compute power and memory that users can consume. For information about quotas, see [Assign Resource Quotas to Users](#).
- Create network and compute profiles, so that you can customize the networking, CPU, memory, and storage for different types of cluster. For information about network and compute profiles, see [Creating and Managing Network Profiles in the Management Console](#) and [Creating and Managing Compute Profiles in the Management Console](#).
- When you have set up users and configured network and compute profiles, you can easily [Create Clusters in the Management Console](#), applying network and Kubernetes profiles to the clusters.
- After you create clusters, you can [Monitor and Manage Clusters, Nodes, and Namespaces in the Management Console](#).
- If a new version of Kubernetes is available, you can [Upgrade Clusters to a New Version of Kubernetes](#).

You can create Kubernetes clusters in your VMware Tanzu Kubernetes Grid Integrated Edition

deployment by using the VMware Tanzu Kubernetes Grid Integrated Edition CLI or by using the VMware Tanzu Kubernetes Grid Integrated Edition Management Console. For information about using the VMware Tanzu Kubernetes Grid Integrated Edition CLI to create clusters, see [Creating Clusters](#).

For information about how to deploy the management console and install Tanzu Kubernetes Grid Integrated Edition, see [Install on vSphere with the Management Console](#).

## Create Clusters in the Management Console

You can deploy Kubernetes clusters to Tanzu Kubernetes Grid Integrated Edition directly from Tanzu Kubernetes Grid Integrated Edition Management Console on vSphere.

When you deploy Kubernetes clusters from the management console, you select the following pre-existing resources to configure your cluster:

- A plan from the list of plans that were defined when the management console was deployed.
- An optional network profile. Network profiles allow cluster administrators and cluster managers to customize the networking for different types of Kubernetes cluster. For information about how to create network profiles in the management console, see [Working with Network Profiles](#).
- An optional Kubernetes profile. Kubernetes profiles enable cluster administrators and cluster managers to customize Kubernetes component settings for any clusters that they provision. You create Kubernetes profiles outside of the management console, by using the TKG CLI to define the Kubernetes profile in your Tanzu Kubernetes Grid Integrated Edition instance.

## Create Clusters

1. Go to the **TKG Integrated Edition** view of the management console.
2. Select the **Clusters** tab and click **Create Cluster**.



[View a larger version of this image](#)

3. Use the **Plan** drop-down menu to select one of the plans that were configured during the deployment of Tanzu Kubernetes Grid Integrated Edition Management Console.

The plan defines the set of resources that the Kubernetes cluster will use. A summary of the selected plan appears as you hover over each option.

## Create Cluster

Add a new Kubernetes cluster by first choosing a plan and then specify the rest of the attributes

|                                                                                                                                                                                                                                  |                                                                                                                                                                                                                                                                                                                                     |                                                                                                     |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------|
| <b>Plan</b><br><div style="border: 1px solid #ccc; padding: 5px; width: fit-content; margin-left: auto; margin-right: 0;">           Select a plan ▾<br/>           Small<br/> <b>Medium</b><br/>           Large         </div> | This is Medium plan<br><div style="border: 1px solid #ccc; padding: 5px; width: fit-content; margin-left: auto; margin-right: 0;"> <b>Master/ETCD Nodes</b><br/>           CPU: 4; RAM: 32768 MB<br/>           Persistent Disk: 160 MB<br/>           CPU: 2; RAM: 10000 MB<br/>           Persistent Disk: 1024 MB         </div> | <b>Worker Nodes</b><br>Disk: 1048576 MB<br>AZs: az1,az2,az3<br>Disk: 1048576 MB<br>AZs: az1,az2,az3 |
| <a href="#">SHOW MORE</a>                                                                                                                                                                                                        |                                                                                                                                                                                                                                                                                                                                     |                                                                                                     |

- Enter a name and a host name for the cluster, and specify the number of worker nodes to create.

|                                                                                                                                                                                                                      |                                                      |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------|
| <b>Name</b>                                                                                                                                                                                                          | <input type="text" value="mytestcluster"/>           |
| <b>Hostname</b>                                                                                                                                                                                                      | <input type="text" value="mytestcluster.example.c"/> |
| <b>Worker Nodes</b>                                                                                                                                                                                                  | <input type="text" value="2"/>                       |
| <b>Note:</b> Use only lowercase characters when naming your cluster if you manage your clusters with Tanzu Mission Control (TMC). Clusters with names that include an uppercase character cannot be attached to TMC. |                                                      |

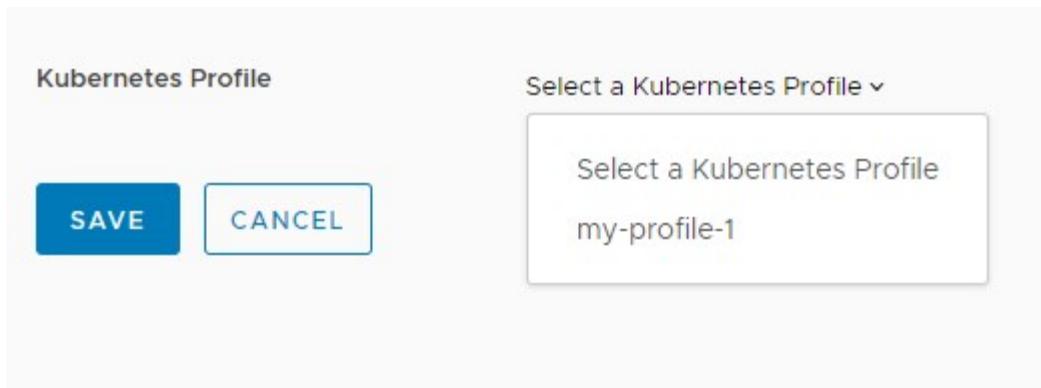
- Use the **Network Profile** and **Compute Profile** drop-down menus to select existing network and compute profiles for the cluster to use.

|                                                                                                                                                              |                     |                                                                                                                                                                                                      |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Network Profile</b>                                                                                                                                       | network-profile-1 ▾ | <b>Network Profile Parameters</b>                                                                                                                                                                    |
| Name: network-profile-1<br>Description: Example Network Profile with All Available Parameters -- FOR ILLUSTRATION PURPOSES ONLY<br>Load Balancer Size: large |                     |                                                                                                                                                                                                      |
| <a href="#">SHOW MORE</a>                                                                                                                                    |                     |                                                                                                                                                                                                      |
| <b>Compute Profile</b>                                                                                                                                       |                     | dev ▾                                                                                                                                                                                                |
| <a href="#">SHOW MORE</a>                                                                                                                                    |                     | <div style="border: 1px solid #ccc; padding: 5px; width: fit-content; margin-left: auto; margin-right: 0;">           Select a Compute Profile<br/>           dev<br/>           dev2         </div> |
| <b>Kubernetes Profile</b>                                                                                                                                    |                     |                                                                                                                                                                                                      |
| <a href="#">CREATE</a> <a href="#">CANCEL</a>                                                                                                                |                     |                                                                                                                                                                                                      |

If you have not created any network or compute profiles, the management console uses the default profiles. In this case, Tanzu Kubernetes Grid Integrated Edition Management Console configures networking and compute for you, based on the plan that you selected.

- Optionally use the Kubernetes Profile drop-down menu to select an existing Kubernetes

profile for the cluster to use.



If you have not created any Kubernetes profiles, the management console uses the default Kubernetes profile. In this case, Tanzu Kubernetes Grid Integrated Edition Management Console configures the cluster for you, based on the plan that you selected.

7. Click **Create** to deploy your cluster.

You can follow the progress of the deployment of your cluster in the **Clusters** tab.

| Name                        | Status   | Endpoint Address              | # Masters | # Workers | K8s Version |
|-----------------------------|----------|-------------------------------|-----------|-----------|-------------|
| shared-auth-services        | Running  | https://10.10.17.200:8443/api | 1         | 2         | 1.11.2      |
| shared-api-gateway-services | Creating |                               |           |           | 1.11.2      |

## Update Cluster Configuration

After you have deployed clusters, you can modify their configuration when they are in the Running state.

1. Select a cluster in the **Clusters** tab and click **Update**.

| Name                 | Status  | Endpoint Address              | # Masters | # Workers | K8s Version |
|----------------------|---------|-------------------------------|-----------|-----------|-------------|
| shared-auth-services | Running | https://10.10.17.200:8443/api | 1         | 2         | 1.11.2      |

2. Use the drop-down menus to optionally change the network, compute, or Kubernetes profiles for the cluster.
3. Optionally click **Modify Worker Nodes** to update the number of worker nodes.

## Update Cluster | shared-auth-services

Update the attributes of the cluster

|                                                                                                                                |                                                              |                                                                                                                                                                                                                                |
|--------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Network Profile</b><br>network-profile-1▼                                                                                   | <b>Compute Profile</b><br>dev ▼<br><a href="#">SHOW MORE</a> | <b>Network Profile Parameters</b><br>Name: network-profile-1<br>Description: Example Network Profile with All Available Parameters -- FOR ILLUSTRATION PURPOSES ONLY<br>Load Balancer Size: large<br><a href="#">SHOW MORE</a> |
|                                                                                                                                |                                                              | <a href="#">MODIFY WORKER NODES</a>                                                                                                                                                                                            |
| <b>Kubernetes Profile</b><br>my-profile-1▼                                                                                     |                                                              |                                                                                                                                                                                                                                |
| <b>Kubernetes Profile Parameters</b><br>Name: my-profile-1<br>Description: My profile description<br><a href="#">SHOW MORE</a> |                                                              |                                                                                                                                                                                                                                |

4. Optionally expand **Advanced Settings** to update the settings for node drain and the pod shutdown grace period.

**Advanced Settings**

|                                                                                                                                                                               |     |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| Node Drain Timeout (minutes, min: 0, max: 1440) <small> ⓘ</small>                                                                                                             | 50  |
| Pod Shutdown Grace Period (seconds, min: -1, max: 86400) <small> ⓘ</small>                                                                                                    | 150 |
| <input type="checkbox"/> Force node to drain even if it has running pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or Stateful Set <small> ⓘ</small> |     |
| <input type="checkbox"/> Force node to drain even if it has running DaemonSet managed pods <small> ⓘ</small>                                                                  |     |
| <input checked="" type="checkbox"/> Force node to drain even if it has running pods using emptyDir <small> ⓘ</small>                                                          |     |
| <input checked="" type="checkbox"/> Force node to drain even if pods are still running after timeout <small> ⓘ</small>                                                        |     |

[SAVE](#) [CANCEL](#)

## Upgrade Clusters to a New Version of Kubernetes

If you make a new version of Kubernetes available by upgrading Tanzu Kubernetes Grid Integrated Edition Management Console, you can upgrade your existing clusters in the management console.

1. Go to the **TKG Integrated Edition** view of the management console.
2. Select the **Clusters** tab.
3. Select one or more clusters and click **Upgrade**.

Clusters must be in the running state for upgrade to be possible.

| Name                        | Status        | Endpoint Address              | # Masters | # Workers | K8s Version |
|-----------------------------|---------------|-------------------------------|-----------|-----------|-------------|
| shared-auth-services        | Running       | https://10.10.17.200:8443/api | 1         | 2         | 1.11.2      |
| shared-api-gateway-services | Creating      |                               |           |           | 1.11.2      |
| app-dev                     | Deleting      |                               |           |           | 1.11.2      |
| app-concourse-pipeline      | Reconfiguring |                               |           |           | 1.11.2      |
| app-prod                    | Running       | https://10.10.152.77:8443/api | 3         | 10        | 1.11.2      |
| shared-auth-services        | Running       | https://10.10.17.200:8443/api | 1         | 2         | 1.11.2      |

[View a larger version of this image](#)

- Click **Upgrade** to confirm.

Do you want to upgrade the following clusters? This operation cannot be reversed.

- shared-auth-services ( v1.11.2 --> v1.17.5 )
- app-prod ( v1.11.2 --> v1.17.5 )
- shared-auth-services ( v1.11.2 --> v1.17.5 )

[View a larger version of this image](#)

## Delete Clusters

You can delete a cluster that you no longer require.

To avoid an incomplete deletion, prepare the cluster for deletion before deleting it:

- If the cluster is configured with a PodDisruptionBudget (PDB), remove the PDB from the cluster.
- Remove all static and dynamic PVCs from the cluster.
- Remove all PVs with a reclaimPolicy of `Retain` from the cluster.



**Note:** Before deleting a cluster, remove the PVs and PVCs from the cluster to avoid making orphan disks of the cluster's attached disks.

To delete a cluster:

1. Go to the **TKG Integrated Edition** view of the management console.
2. Select the **Clusters** tab.
3. Select the cluster to be deleted.
4. Click **Delete**.

| Name                 | Status  | Endpoint Address              | # Masters | # Workers | K8s Version |
|----------------------|---------|-------------------------------|-----------|-----------|-------------|
| shared-auth-services | Running | https://10.10.17.200:8443/api | 1         | 2         | 1.11.2      |

## Next Steps

- Monitor and Manage Clusters, Nodes, and Namespaces in the Management Console
- Connect to Clusters with kubectl

## Monitor and Manage Clusters, Nodes, and Namespaces in the Management Console

You can find general information about your deployment on vSphere, and information about all of the clusters and nodes running it, in the **TKG Integrated Edition** view of Tanzu Kubernetes Grid Integrated Edition Management Console.

## Obtain Cluster Information

1. Go to the **TKG Integrated Edition** view of the management console.
2. Select the **Clusters** tab to see detailed information about all of the clusters running in this instance.

| Name                        | Status   | Endpoint Address              | # Masters | # Workers | K8s Version |
|-----------------------------|----------|-------------------------------|-----------|-----------|-------------|
| shared-auth-services        | Running  | https://10.10.17.200:8443/api | 1         | 2         | 1.11.2      |
| shared-api-gateway-services | Creating |                               |           |           |             |

[View a larger version of this image](#)

3. Select a cluster.

On the **Summary** tab for the cluster, you see general information about that cluster, as well as networking, and the nodes in that cluster.

- ◊ In the Cluster Overview panel, select the Availability Zone links to be taken the vSphere cluster, host group, or resource pool that contains the cluster.
- ◊ In the Networking panel, select the links to be taken to each of the different components that comprise the network stack for the cluster.
- ◊ In the Storage panel, expand **Persistent Volume Claims** to see the volumes that your cluster is using.
- ◊ In the Nodes panel, expand **Masters** and **Workers** and select the VM links to go to those VMs in the vSphere inventory.

The screenshot shows the Tanzu Kubernetes Grid Integrated Edition interface. At the top, there's a header with a back arrow, the title 'TANZU KUBERNETES GRID INTEGRATED EDITION', a cluster name 'k8s1' with a 'RUNNING' status, and an 'ACCESS CLUSTER' button. Below the header are three tabs: 'Summary' (selected), 'Nodes', and 'Namespaces'. The 'Summary' tab contains four main sections: 'Cluster Overview', 'Networking', 'Storage', and 'Nodes'.

- Cluster Overview:** Displays cluster details like Created (Tue Apr 10 2018 13:36:26 GMT-0500 (CDT)), K8s Version (1.9.3), Linux Stemcell Version (ubuntu-xenial - Version: 456.30), Windows Stemcell Version (windows - Version: 1.1.0), API Endpoint (<https://1.1.1.18443/api>), Node VM Size (master) (medium.disk (2 vCPU 4 GB memory)), Node VM Size (worker) (large (2 vCPU 8 GB memory)), Availability Zone (az1 ( cr1, cr2, cr3 ), az2 ( cr3 )), Namespaces (4), Plan Name (small), Kubernetes Profile Name (kube profile 1), and Kubernetes Master IP(s) (10.33.22.11, 10.192.222.22).
- Networking:** Shows network configuration details such as Stack (NSX-T Data Center), Network profile (Default network profile), Tier-0 Router (ce7af943-f0f1-4b92-b783-5d96ff50c233), IP Pool (3), Load Balancer Size (MEDIUM), Load Balancer (8d92fe3c-84e9-48da-8f44-2ea143fa6962), Node IP Mode (NATED / ROUTABLE), Node IP Block (3), Pod IP Block (3), Node Network Logical Switch (ce7af943-f0f1-4b92-b783-5d96ff50c233), and Node Network Logical Router (b2710064-ae9d-4a78-9c52-88dcbeb2cf50).
- Storage:** Shows Persistent Volume Claims (3).
- Nodes:** Shows two sections: Masters (8 nodes) and Workers (10 of 16 nodes). Each section has a 'View all' link.

4. Select the **Nodes** tab to see details of all of the nodes that are running in that cluster.

The screenshot shows the 'Nodes' tab from the Tanzu Kubernetes Grid Integrated Edition interface. It displays a table of running nodes across various clusters and availability zones.

| Name           | Status  | Type           | Kubernetes Clusters         | VM Name      | VM IP        | Availability Zone |
|----------------|---------|----------------|-----------------------------|--------------|--------------|-------------------|
| Auth Node 1    | Running | Master         | shared-auth-services        | Auth VM 1    | 10.10.100.1  | US West           |
| Auth Node 2    | Running | windows-worker | shared-auth-services        | Auth VM 2    | 10.10.100.16 | US West           |
| Auth Node 3    | Running | Worker         | shared-auth-services        | Auth VM 3    | 10.10.100.23 | US West           |
| Gateway Node 1 | Running | Master         | shared-api-gateway-services | Gateway VM 1 | 10.10.101.1  | US East           |

5. Select the **Namespaces** tab to see the status and networking details of all of the namespaces that are running in that cluster.

The screenshot shows the Tanzu Kubernetes Grid Integrated Edition management console. At the top, it says "TANZU KUBERNETES GRID INTEGRATED EDITION". Below that, it shows a cluster named "k8s1" which is "RUNNING". There are three tabs: "Summary", "Nodes", and "Namespaces", with "Namespaces" being the active tab. The main area displays a table of namespaces:

| Name         | Status  | Age | Pod Network                          | Tier-1 Router                        |
|--------------|---------|-----|--------------------------------------|--------------------------------------|
| auth-default | Running |     | fb6e4578-6b07-4d96-8aaa-372041c1acbc | -                                    |
| auth-test    | Running |     | b18e1931-b1d7-4c89-9499-15b79e4942cc | 49ad5187-9378-4601-b012-a9c4f2c98459 |
| auth-system  | Running |     | dd4561ff-f4f6-4c31-991b-7633246e17d8 | 2a8c2bc8-cd3c-4851-b953-c48c984582fb |
| auth-dev     | Running |     | 91ea02a8-6cd1-4be8-b5cf-a59ae23dfcd4 | b64fe8c8-731d-447f-80d4-e8e82b1d4c09 |

At the bottom right of the table, there are buttons for "Namespaces per page" (set to 20) and "1 - 4 of 4 Namespaces".

## Connect to Clusters with `kubectl`

1. Go to the Tanzu Kubernetes Grid Integrated Edition view of the management console.
2. Select the **Clusters** tab for your Tanzu Kubernetes Grid Integrated Edition instance.
3. Select a cluster.
4. Select **Access Cluster** for instructions about how to access the cluster by using `kubectl`.

The screenshot shows the Tanzu Kubernetes Grid Integrated Edition management console with the "Clusters" tab selected. On the left, there's a sidebar with "TKG Integrated Edition" selected, and under it, "Quotas" and "Network Profiles". The main area shows the cluster "k8s1" which is "RUNNING". There are three tabs: "Summary", "Nodes", and "Namespaces", with "Namespaces" being the active tab. At the top right, there's a "ACCESS CLUSTER" button, which is highlighted with a red box.

5. Click **Email** to send the instructions to users who need to use `kubectl` to connect to this cluster.

## Obtain Node Information

1. Go to the Tanzu Kubernetes Grid Integrated Edition view of the management console.
2. Select the **Nodes** tab to see detailed information about all of the nodes running in this instance.

This tab shows the general status, type, name, IP, and availability zone for all of the nodes that are running in your Tanzu Kubernetes Grid Integrated Edition instance.

| Name           | Status  | Type           | Kubernetes Clusters         | VM Name      | VM IP        | Availability Zone |
|----------------|---------|----------------|-----------------------------|--------------|--------------|-------------------|
| Auth Node 1    | Running | Master         | shared-auth-services        | Auth VM 1    | 10.10.100.1  | US West           |
| Auth Node 2    | Running | windows-worker | shared-auth-services        | Auth VM 2    | 10.10.100.16 | US West           |
| Auth Node 3    | Running | Worker         | shared-auth-services        | Auth VM 3    | 10.10.100.23 | US West           |
| Gateway Node 1 | Running | Master         | shared-api-gateway-services | Gateway VM 1 | 10.10.101.1  | US East           |
| Gateway Node 2 | Running | Master         | shared-api-gateway-services | Gateway VM 2 | 10.10.101.2  | US East           |

3. Click the links in the Kubernetes Clusters columns to go to the **Summary** tab of that cluster.

| Name        | Status  | Type           | Kubernetes Clusters  | VM Name   | VM IP        | Availability Zone |
|-------------|---------|----------------|----------------------|-----------|--------------|-------------------|
| Auth Node 1 | Running | Master         | shared-auth-services | Auth VM 1 | 10.10.100.1  | US West           |
| Auth Node 2 | Running | windows-worker | shared-auth-services | Auth VM 2 | 10.10.100.16 | US West           |

4. Click the links in the VM Name column to be taken to the node VMs in the vSphere inventory.

| Name        | Status  | Type           | Kubernetes Clusters  | VM Name   | VM IP        | Availability Zone |
|-------------|---------|----------------|----------------------|-----------|--------------|-------------------|
| Auth Node 1 | Running | Master         | shared-auth-services | Auth VM 1 | 10.10.100.1  | US West           |
| Auth Node 2 | Running | windows-worker | shared-auth-services | Auth VM 2 | 10.10.100.16 | US West           |

## Managing Clusters with the CLI

This section describes how to use the CLI to manage Kubernetes clusters in VMware Tanzu Kubernetes Grid Integrated Edition.

See the following topics:

- [Logging in to Tanzu Kubernetes Grid Integrated Edition](#)
- [Creating Clusters](#)
- [Using Kubernetes Profiles](#)
- [Using Network Profiles](#)
- [Viewing Cluster Lists](#)
- [Viewing Cluster Details](#)
- [Viewing Cluster Plans](#)

- [Scaling Existing Clusters](#)
- [Upgrading Clusters](#)
- [Rotating Cluster Certificates](#)
- [Deleting Clusters](#)

## Logging in to Tanzu Kubernetes Grid Integrated Edition

This topic describes how to log in to VMware Tanzu Kubernetes Grid Integrated Edition.

### Overview

To manage Tanzu Kubernetes Grid Integrated Edition-deployed clusters, you use the TKGI Command Line Interface (TKGI CLI). When you log in to Tanzu Kubernetes Grid Integrated Edition successfully for the first time, the TKGI CLI generates a local `creds.yml` file that contains the API endpoint, refresh token, access token, and CA certificate, if applicable.

By default, `creds.yml` is saved in the `~/.pks` directory on your local system. You can use the `TKGI_HOME` environment variable to override this location and store `creds.yml` in any directory on your system.

### Prerequisites

Before you can log in to Tanzu Kubernetes Grid Integrated Edition, you must have the following:

- A running Tanzu Kubernetes Grid Integrated Edition environment, including an external load balancer configured to forward traffic to the TKGI API endpoint. See the *Installing Tanzu Kubernetes Grid Integrated Edition* section for your cloud provider.
- A username and password that has access to the TKGI API. See [Managing Tanzu Kubernetes Grid Integrated Edition Users with UAA](#).
- The TKGI CLI installed on your local system. See [Installing the TKGI CLI](#).

### Log in to the TKGI CLI

Use the command in this section to log in as an individual user. The login procedure is the same for users created in UAA or users from external LDAP groups.

On the command line, run the following command in your terminal to log in to the TKGI CLI:

```
tkgi login -a TKGI-API -u USERNAME -p PASSWORD -ca-cert CERT-PATH
```

Replace the placeholder values in the command as follows:

- `TKGI-API` is the domain name for the TKGI API that you entered in **Ops Manager > Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN)**. For example, `api.tkgi.example.com`.
- `USERNAME` and `PASSWORD` belong to the account you created in the **Grant Tanzu Kubernetes Grid Integrated Edition Access to an Individual User** section of [Managing Tanzu Kubernetes Grid Integrated Edition Users with UAA](#). If you do not use `-p` to provide a password, the TKGI

CLI prompts for the password interactively. VMware recommends running the login command without the `-p` flag for added security.

- `CERT-PATH` is the path to your root CA certificate. Provide the certificate to validate the TKGI API certificate with SSL.

For example:

```
$ tkgi login -a api.tkgi.example.com -u alana
--ca-cert /var/tempest/workspaces/default/root_ca_certificate
```

If you are logging in to a trusted environment, you can use `-k` to skip SSL verification instead of `--ca-cert CERT-PATH`.

For example:

```
$ tkgi login -a api.tkgi.example.com -u alana -k
```



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

## Log in to the TKGI CLI as an Automated Client

To log in to the TKGI CLI as an automated client for a script or service, run the following command:

```
tkgi login -a TKGI-API --client-name CLIENT-NAME --client-secret CLIENT-SECRET --ca-cert CERTIFICATE-PATH
```

Where:

- `TKGI-API` is the domain name for the TKGI API that you entered in **Ops Manager > Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN)**. For example, `api.tkgi.example.com`.
- `CLIENT-NAME` is an OAuth client ID for either:
  - A UAA admin client created with `--authorities "pks.clusters.admin"`
  - The default admin client **Pks Uaa Management Admin Client**
- `CLIENT-SECRET` the OAuth client secret for the `--client-name` value above.
- `CERTIFICATE-PATH` is the path to your root CA certificate. Provide the certificate to validate the TKGI API certificate with SSL.

For example:

```
$ tkgi login -a api.tkgi.example.com
```

```
-client-name automated-client
-client-secret randomly-generated-secret
-ca-cert /var/tempest/workspaces/default/root_ca_certificate
```

For information on how to create a UAA client, see [Grant Tanzu Kubernetes Grid Integrated Edition Access to a Client](#) in *Managing Tanzu Kubernetes Grid Integrated Edition Users with UAA*.

## Export TKGI API Access Token

This procedure stores a TKGI API access token as an environment variable that you can use when executing TKGI API calls on the command line.

1. To export your access token into an environment variable, run the following command:

```
tkgi login -a TKGI-API -u USER-ID -p 'PASSWORD' -k; \
export YOUR-ACCESS-TOKEN=$(bosh int ~/.pks/creds.yml --path /access_token)
```

Where:

- TKGI-API is the FQDN of your TKGI API endpoint. For example, `api.tkgi.example.com`.
- USER-ID is your Tanzu Kubernetes Grid Integrated Edition user ID.
- PASSWORD is your Tanzu Kubernetes Grid Integrated Edition password.
- YOUR-ACCESS-TOKEN is the name of your access token environment variable.

For example:

```
$ tkgi login -a tkgi.my.lab -u alana -p 'psswrdabc123...!' -k;
export my_token=$(bosh int ~/.pks/creds.yml --path /access_token)
```



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

## Creating Clusters

This topic describes how to create a Kubernetes cluster with VMware Tanzu Kubernetes Grid Integrated Edition using the TKGI Command Line Interface (TKGI CLI).

## Overview

Use the TKGI CLI to create Kubernetes clusters in your Tanzu Kubernetes Grid Integrated Edition environment.

To create an Tanzu Kubernetes Grid Integrated Edition Kubernetes cluster, do the following:

- [Configure Cluster Access](#)
- [Create a Kubernetes Cluster](#)
- [Identify Kubernetes Cluster Control Plane VMs](#)

The `tkgi create-cluster` command creates a Kubernetes cluster with TKGI compatibility matching the TKGI version of the current TKGI control plane.

## Configure Cluster Access

Cluster access configuration differs by the type of Tanzu Kubernetes Grid Integrated Edition deployment.

### vSphere with NSX-T

Tanzu Kubernetes Grid Integrated Edition deploys a load balancer automatically when clusters are created. The load balancer is configured automatically when workloads are being deployed on these Kubernetes clusters. For more information, see [Load Balancers in Tanzu Kubernetes Grid Integrated Edition Deployments with NSX-T](#).



**Note:** For a complete list of the objects that Tanzu Kubernetes Grid Integrated Edition creates by default when you create a Kubernetes cluster on vSphere with NSX-T, see [vSphere with NSX-T Cluster Objects](#).

### GCP, AWS, Azure, or vSphere without NSX-T

When you create a Kubernetes cluster, you must configure external access to the cluster by creating an external TCP or HTTPS load balancer. This load balancer allows you to run TKGI CLI commands on the cluster from your local workstation. For more information, see [Load Balancers in Tanzu Kubernetes Grid Integrated Edition Deployments without NSX-T](#).

You can configure any load balancer of your choice. If you use GCP, AWS, Azure, or vSphere without NSX-T, you can create a load balancer using your cloud provider console.

For more information about configuring a Tanzu Kubernetes Grid Integrated Edition cluster load balancer, see the following:

- [Creating and Configuring a GCP Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters](#)
- [Creating and Configuring an AWS Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters](#)
- [Creating and Configuring an Azure Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters](#)

Create the Tanzu Kubernetes Grid Integrated Edition cluster load balancer before you create the cluster. Use the load balancer IP address as the external hostname, and then point the load balancer

to the IP address of the control plane virtual machine (VM) after cluster creation. If the cluster has multiple control plane nodes, you must configure the load balancer to point to all control plane VMs for the cluster.

If you are creating a cluster in a non-production environment, you can choose to create a cluster without a load balancer. Create a DNS entry that points to the IP address of the cluster's control plane VM after cluster creation.

To locate the IP addresses and VM IDs of the control plane VMs, see [Identify the Kubernetes Cluster Control Plane VM](#) below.

## Create a Kubernetes Cluster

Perform the following steps:

1. Grant cluster access to a new or existing user in UAA. For more information, see the [Grant Tanzu Kubernetes Grid Integrated Edition Access to an Individual User](#) section of *Managing Tanzu Kubernetes Grid Integrated Edition Users with UAA*.
2. On the command line, run the following command to log in:

```
tkgi login -a TKGI-API -u USERNAME -k
```

Where:

- ◊ **TKGI-API** is the domain name for the TKGI API that you entered in **Ops Manager > Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN)**. For example, [api.tkgi.example.com](http://api.tkgi.example.com).
- ◊ **USERNAME** is your user name.

See [Logging in to Tanzu Kubernetes Grid Integrated Edition](#) for more information about the `tkgi login` command.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

3. (Optional) To configure any of the following for a cluster, create a config file:
  - ◊ **Custom CAs:** For more information, see [Using a Custom CA for Kubernetes Clusters](#).
  - ◊ **VM Extensions:** For more information, see [Using BOSH VM Extensions](#).
  - ◊ **Proxy settings:** For more information, see [Configure Cluster Proxies](#).
  - ◊ **group Managed Service Account (gMSA) settings:** For more information, see [Authenticate Windows Clusters with Active Directory](#).
4. To create a cluster, run the following command:

```
tkgi create-cluster CLUSTER-NAME \
--external-hostname HOSTNAME \
--plan PLAN-NAME \
[--num-nodes WORKER-NODES] \
[--network-profile NETWORK-PROFILE-NAME] \
[--kubernetes-profile KUBERNETES-PROFILE-NAME] \
[--config-file CONFIG-FILE-NAME] \
[--tags TAGS]
```

Where:

- ◊ **CLUSTER-NAME** is your unique name for your cluster.



**Note:** The **CLUSTER-NAME** must not contain special characters such as &. The TKGI CLI does not validate the presence of special characters in the **CLUSTER-NAME** string, but cluster creation fails if one or more special characters are present.

Use only lowercase characters when naming your cluster if you manage your clusters with Tanzu Mission Control (TMC). Clusters with names that include an uppercase character cannot be attached to TMC.

- ◊ **HOSTNAME** is your external hostname for your cluster. You can use any fully qualified domain name (FQDN) or IP address you own. For example, [my-cluster.example.com](#) or [10.0.0.1](#). If you created an external load balancer, use its DNS hostname. If you are using NSX-T, you can pre-provision the IP address to use for the Kubernetes API server load balancer using an available IP address from the floating IP pool and define a network profile to perform DNS lookup, or specify the IP address to use for load balancer on the command line. See [Defining Network Profile for DNS Lookup of Pre-Provisioned IP Addresses](#) for details.
- ◊ **PLAN-NAME** is the plan for your cluster. Run `tkgi plans` to list your available plans.
- ◊ (Optional) **WORKER-NODES** is the number of worker nodes for the cluster.
- ◊ (Optional) (NSX-T only) **NETWORK-PROFILE-NAME** is the network profile to use for the cluster. See [Using Network Profiles \(NSX-T Only\)](#) for more information.
- ◊ (Optional) **KUBERNETES-PROFILE-NAME** is the Kubernetes profile to use for the cluster. See [Using Kubernetes Profiles](#) for more information.
- ◊ (Optional) **CONFIG-FILE-NAME** is the configuration file to use for the cluster.
- ◊ (Optional) (Azure and vSphere only) **TAGS** are the labels and metadata values to apply to the VMs created in the cluster. Specify the tags as `key:value` pairs. For more information about tagging see [Tagging Clusters](#).

For example:

```
$ tkgi create-cluster my-cluster \
-external-hostname my-cluster.example.com \
```

```
-plan large -num-nodes 3
```



**Note:** It can take up to 30 minutes to create a cluster.

For high availability, create clusters with a minimum of three worker nodes, or two per AZ if you intend to use PersistentVolumes (PVs). For example, if you deploy across three AZs, you should have six worker nodes. For more information about PVs, see [PersistentVolumes](#) in *Maintaining Workload Uptime*. Provisioning a minimum of three worker nodes, or two nodes per AZ is also recommended for stateless workloads.

The maximum value you can specify is configured in the **Plans** pane of the Tanzu Kubernetes Grid Integrated Edition tile. If you do not specify a number of worker nodes, the cluster is deployed with the default number, which is also configured in the **Plans** pane. For more information, see the *Installing Tanzu Kubernetes Grid Integrated Edition* topic for your IaaS, such as [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#).

- To track cluster creation, run the following command:

```
tkgi cluster CLUSTER-NAME
```

Where **CLUSTER-NAME** is the unique name for your cluster.

For example:

```
$ tkgi cluster my-cluster

Name: my-cluster
Plan Name: large
UUID: 01a234bc-d56e-7f89-01a2-3b4cde5f6789
Last Action: CREATE
Last Action State: succeeded
Last Action Description: Instance provisioning completed
Kubernetes Master Host: my-cluster.example.com
Kubernetes Master Port: 8443
Worker Instances: 3
Kubernetes Master IP(s): 192.168.20.7
```

- If the **Last Action State** value is **error**, troubleshoot by performing the following procedure:

- Log in to the BOSH Director.

2. Run the following command:

```
bosh tasks
```

For more information, see [Advanced Troubleshooting with the BOSH CLI](#).

7. Depending on your deployment:

- ◊ For **vSphere with NSX-T**, choose one of the following:
  - Specify the hostname or FQDN and register the FQDN with the IP provided by Tanzu Kubernetes Grid Integrated Edition after cluster deployment. You can do this using `resolv.conf` or via DNS registration.
  - Specify a temporary placeholder value for FQDN, then replace the FQDN in the `kubeconfig` with the IP address assigned to the load balancer dedicated to the cluster.

To retrieve the IP address to access the Kubernetes API and UI services, use the `tkgi cluster CLUSTER-NAME` command.
- ◊ For **vSphere without NSX-T, AWS, and Azure**, configure external access to the cluster's control plane nodes using either DNS records or an external load balancer. Use the output from the `tkgi cluster` command to locate the control plane node IP addresses and ports.
- ◊ For **GCP**, use the output from the `tkgi cluster` command to locate the control plane node IP addresses and ports, and then continue to [Creating and Configuring a GCP Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters](#) in [Configuring a GCP Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters](#).



**Note:** For clusters with multiple control plane node VMs, health checks on port 8443 are recommended.

8. To access your cluster, run the following command:

```
tkgi get-credentials CLUSTER-NAME
```

Where `CLUSTER-NAME` is the unique name for your cluster.

For example:

```
$ tkgi get-credentials tkgi-example-cluster

Fetching credentials for cluster tkgi-example-cluster.

Context set for cluster tkgi-example-cluster.

You can now switch between clusters by using:
```

```
$ kubectl config use-context <cluster-name>
```

The `tkgi get-credentials` command creates a local `kubeconfig` that allows you to manage the cluster. For more information about the `tkgi get-credentials` command, see [Retrieving Cluster Credentials and Configuration](#).



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

- To confirm you can access your cluster using the Kubernetes CLI, run the following command:

```
kubectl cluster-info
```

See [Managing Tanzu Kubernetes Grid Integrated Edition](#) for information about checking cluster health and viewing cluster logs.

- To review the status, container runtime or other information about the nodes in your cluster, run the following command:

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

## Identify Kubernetes Cluster Control Plane VMs



**Note:** This section applies only to Tanzu Kubernetes Grid Integrated Edition deployments on GCP or on vSphere without NSX-T. Skip this section if your Tanzu Kubernetes Grid Integrated Edition deployment is on vSphere with NSX-T, AWS, or Azure. For more information, see [Load Balancers in Tanzu Kubernetes Grid Integrated Edition](#).

To reconfigure the load balancer or DNS record for an existing cluster, you might need to locate VM ID and IP address information for the cluster's control plane VMs. Use the information you locate in this procedure when configuring your load balancer backend.

To locate the IP addresses and VM IDs for the control plane VMs of an existing cluster, do the following:

- On the command line, run the following command to log in:

```
tkgi login -a TKGI-API -u USERNAME -k
```

Where:

- TKGI-API is the domain name for the TKGI API that you entered in [Ops Manager >](#)

**Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN).**

For example, `api.tkgi.example.com`.

- ◊ `USERNAME` is your user name.

See [Logging in to Tanzu Kubernetes Grid Integrated Edition](#) for more information about the `tkgi login` command.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

2. To locate the cluster ID and control plane node IP addresses, run the following command:

```
tkgi cluster CLUSTER-NAME
```

Where `CLUSTER-NAME` is the unique name for your cluster.

From the output of this command, record the following items: \* **UUID**: This value is your cluster ID. \* **Kubernetes Master IP(s)**: This value lists the IP addresses of all control plane nodes in the cluster.

3. Gather credential and IP address information for your BOSH Director.
4. To log in to the BOSH Director, perform the following:
  1. SSH into the Ops Manager VM.
  2. Log in to the BOSH Director by using the BOSH CLI from the Ops Manager VM.

For information on how to complete these steps, see [Advanced Troubleshooting with the BOSH CLI](#).

5. To identify the name of your cluster deployment, run the following command:

```
bosh -e tkgi deployments
```

Your cluster deployment name begins with `service-instance` and includes the UUID you located in a previous step.

6. To identify the control plane VM IDs by listing the VMs in your cluster, run the following command:

```
bosh -e tkgi -d CLUSTER-SI-ID vms
```

Where `CLUSTER-SI-ID` is your cluster service instance ID which begins with `service-instance` and includes the `UUID` you previously located.

For example:

```
$ bosh -e tkgi -d service-instance-aa1234567bc8de9f0alc vms
```

Your control plane VM IDs are displayed in the **VM CID** column.

7. Use the control plane VM IDs and other information you gathered in this procedure to configure your load balancer backend. For example, if you use GCP, use the control plane VM IDs retrieved during the previous step in [Creating and Configuring a GCP Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters](#).

## Next Steps

If you did not tag your new cluster during creation, tag your cluster's VMs now. If your Tanzu Kubernetes Grid Integrated Edition deployment is on:

- **AWS:** Tag your subnets with your new cluster's unique identifier before adding the subnets to the Tanzu Kubernetes Grid Integrated Edition workload load balancer. After you complete the [Create a Kubernetes Cluster](#) procedure above, follow the instructions in [AWS Prerequisites in Deploying and Exposing Basic Linux Workloads](#).
- **Azure, vSphere, or vSphere with NSX-T:** You can use the TKGI CLI to tag clusters by following the steps in [Tagging Clusters](#).
- **GCP:** You can tag your clusters using your IaaS-provided management console.

## Using Kubernetes Profiles

This topic describes how VMware Tanzu Kubernetes Grid Integrated Edition administrators and cluster managers create and use Kubernetes profiles for Kubernetes clusters provisioned by Tanzu Kubernetes Grid Integrated Edition.

This topic also lists verified use cases for Kubernetes profiles on Tanzu Kubernetes Grid Integrated Edition.

## Overview

Kubernetes profiles enable cluster administrators and cluster managers to customize Kubernetes component settings for any clusters that they provision.

To use Kubernetes profiles, Tanzu Kubernetes Grid Integrated Edition users:

1. Create JSON configuration files that set configuration options for any Kubernetes components, such as `kube-apiserver` on the control plane or the `kubelet` on each node.
2. Use the TKGI CLI to create the Kubernetes profile in Tanzu Kubernetes Grid Integrated Edition.
3. Use the TKGI CLI to apply the profile to clusters.

Verified uses for Kubernetes profiles include encrypting secrets in an etcd database, adding an OIDC provider, and using a `ResourceQuota` admission control plugin.

## Who Creates and Manages Kubernetes Profiles

Users with the `pks.clusters.admin` or `pks.clusters.manage` roles can create and use Kubernetes

profiles.

If user with the `pks.clusters.admin-read-only` role attempts to create a Kubernetes profile, they see the following error:

You do not have enough privileges to perform this action. Please contact the TKGI administrator.

## Validated vs Experimental Customizations

A Kubernetes profile configures settings for Kubernetes components in two JSON code blocks, `customizations` and `experimental_customizations`. See [Kubernetes Profile Format](#) for details. The code blocks differ as follows:

- `customizations` block:
  - TKGI checks the validity of configurations in this block. If you run `tkgi create-k8s-profile` on a profile with invalid configurations in `customizations`, the command returns an error.
  - The TKGI team supports clusters configured with tested, validated parameters in this block.
- `experimental_customizations` block:



**Warning:** Experimental customizations are not validated or supported.

- TKGI imposes no restrictions on the contents of this block.
- Configurations in this block are neither tested nor supported.
- If a customer wants to use an unsupported configuration, they should contact the TKGI team with the parameters that they want tested, validated, and supported.

## “k8s” to “kubernetes” Alias in TKGI CLI

In the TKGI CLI, all commands that include `k8s-profile` are aliased to also use `kubernetes-profile`. For example, the `tkgi k8s-profiles` and `tkgi kubernetes-profiles` commands are equivalent.

For brevity, this documentation uses the `k8s-` versions.

## Create a Kubernetes Profile

To create a Kubernetes profile in Tanzu Kubernetes Grid Integrated Edition, you:

1. Define a Kubernetes profile in a JSON configuration file, following the [Kubernetes Profile Format](#) below.
2. Use the TKGI CLI to define the Kubernetes profile within Tanzu Kubernetes Grid Integrated Edition, as described in [The create-network-profile Command](#), below.

## Kubernetes Profile Format

To create a Kubernetes profile, you must first define it as a JSON file that specifies network

parameters, listed in [Kubernetes Profile Parameters](#) below.

Here is the basic structure of a Kubernetes profile.

```
{
 "name": "my-profile-1",
 "description": "My profile description",
 "customizations": [
 {
 "component": "k8s-component-name",
 "arguments": {
 "key": "value,value,value"
 },
 "file-arguments": {
 "key": "local file path"
 }
 }
],
 "experimental_customizations": [
 {
 }
]
}
```



**Note:** This example Kubernetes profile is for illustration purposes only. It is not intended to be used as a template for Kubernetes profile definition.

## Kubernetes Profile Parameters

The Kubernetes profile JSON can include the following parameters:

| Parameter                                | Type   | Description                                                                                                                                                             |
|------------------------------------------|--------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>name</code> *                      | String | Name of the Kubernetes profile.                                                                                                                                         |
| <code>description</code>                 | String | Description of the Kubernetes profile.                                                                                                                                  |
| <code>customizations</code> *            | Map    | A block that defines supported, validated K8s options using the <code>component</code> , <code>arguments</code> , and <code>file-arguments</code> parameters below.     |
| <code>experimental_customizations</code> | Map    | A block that defines unsupported, unvalidated K8s options using the <code>component</code> , <code>arguments</code> , and <code>file-arguments</code> parameters below. |
| <code>component</code> *                 | String | The name of a K8s component, e.g. <code>kubelet</code> , <code>kube-apiserver</code> .                                                                                  |
| <code>arguments</code> *                 | Map    | Parameters for each component, as one or more Key:Value pairs. Multiple values must be separated by commas, without spaces.                                             |
| <code>file-arguments</code> *            | Map    | Parameters whose values are stored as files on the local machine.                                                                                                       |

\*Parameters marked with an \* are mandatory.



**Note:** If you specify the same parameter in both `customizations` and `experimental_customizations`, the one in `customization` takes precedence.

## The `create-k8s-profile` Command

After defining a Kubernetes profile in a JSON file as described in [Kubernetes Profile Format](#), a cluster administrator or manager creates the Kubernetes profile by running the following TKGI CLI command:

```
tkgi create-k8s-profile PATH-TO-YOUR-KUBERNETES-PROFILE-CONFIGURATION
```

Where `PATH-TO-YOUR-KUBERNETES-PROFILE-CONFIGURATION` is the path to the JSON file you created when defining the Kubernetes profile.

For example:

```
cat profile3-docs.json

{
 "name": "my-profile3",
 "description": "My profile description",
 "customizations": [
 {
 "component": "kube-apiserver",
 "arguments": {
 "service-node-port-range": "30000-40000"
 }
 }
],
 "experimental_customizations": [
 {
 "component": "kubelet",
 "arguments": {
 "maximum-dead-containers": "1000",
 "feature-gates": "APIListChunking=true,AttachVolumeLimit=false"
 }
 }
]
}
```

```

}

}

]

}

user ~/workspace: ./tkgi create-k8s-profile k8s-profile3.json

Kubernetes profile my-profile3 successfully created

```

## Manage Kubernetes Profiles

Tanzu Kubernetes Grid Integrated Edition cluster administrators and managers can perform the following operations on Kubernetes profiles and the clusters that use them.

### List Kubernetes Profiles

To list available Kubernetes profiles, run the following command:

```
tkgi k8s-profiles
```

For example:

```
$ tkgi k8s-profiles

K8s-profile Owner Created Date

Basic-k8s-profile Alana Tue, 05 Nov 2019 16:28:15 PST
```

The command output differs by user role:

- `pks.cluster.admin` see a list of Kubernetes profiles that all users created.
- `pks.cluster.manage` see a list of only the Kubernetes profiles that they created.

### Delete a Kubernetes Profile

To delete a Kubernetes profile, run the following command:

```
tkgi delete-k8s-profile KUBERNETES-PROFILE-NAME
```

Where `KUBERNETES-PROFILE-NAME` is the name of the Kubernetes profile you want to delete.

For example:

```
$ user ~/workspace: ./tkgi delete-k8s-profile my-profile3

Are you sure you want to delete the kubernetes profile my-profile3? (y/n): y

Deletion of my-profile3 completed
```



**Note:** You cannot delete a Kubernetes profile that is in use. Before deleting a Kubernetes profile, you must disassociate it from all clusters or delete all clusters it is associated with.

Both `pks.clusters.admin` and `pks.clusters.manage` users can delete Kubernetes profiles. If a `pks.clusters.admin-read-only` user attempts to delete a Kubernetes profile, they see the following error:

```
You do not have enough privileges to perform this action. Please contact the TKGI administrator.
```

## View Kubernetes Profile Details

To view details of a Kubernetes profile, run the following command:

```
tkgi k8s-profile KUBERNETES-PROFILE-NAME
```

Where `KUBERNETES-PROFILE-NAME` is the name of the Kubernetes profile you want to view.

For example:

```
tkgi k8s-profile Basic-k8s-profile

Name: Basic-k8s-profile

Owner: Alana

Created Date: Tue, 05 Nov 2019 16:28:15 PST

Description: Kubernetes profile for customer A

...
<KEY> : <VALUE>
```

- Users with the `pks.cluster.admin` can view the details of any Kubernetes profile; users with the `pks.cluster.manage` role can view details of Kubernetes profiles that they created.
- Once you create or update a cluster with an encryption profile, you cannot assign any other Kubernetes profiles to that cluster. Because decryption is not straightforward, applying another profile can have nondeterministic outcome.

## Create a Cluster with a Kubernetes Profile

You can assign a Kubernetes profile to a Kubernetes cluster at the time of cluster creation.

To create an Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes cluster with a Kubernetes profile, run the following command:

```
tkgi create-cluster CLUSTER-NAME --external-hostname HOSTNAME --plan PLAN-NAME --kubernetes-profile KUBERNETES-PROFILE-NAME
```

Where:

- **CLUSTER-NAME** is a unique name for your cluster.



**Note:** Use only lowercase characters when naming your cluster if you manage your clusters with Tanzu Mission Control (TMC). Clusters with names that include an uppercase character cannot be attached to TMC.

- **HOSTNAME** is your external hostname used for accessing the Kubernetes API.
- **PLAN-NAME** is the name of the Tanzu Kubernetes Grid Integrated Edition plan you want to use for your cluster.
- **KUBERNETES-PROFILE-NAME** is the name of the Kubernetes profile you want to use for your cluster.

## Assign a Kubernetes Profile to an Existing Cluster

TKGI supports changing the Kubernetes profile for an already created cluster. You can use this procedure to:

- assign a Kubernetes profile to a cluster that does not have one, or
- change a cluster's existing profile to a new one

This is the procedure to change a cluster's Kubernetes profile:

1. Do one of the following
  - ◊ Choose a new Kubernetes profile for the cluster. See [List Kubernetes profiles](#).
  - ◊ Define and create a new Kubernetes profile as described in [Create a Kubernetes Profile](#).
    - The name of the new Kubernetes profile must be unique and different from the previously assigned Kubernetes profile.
2. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
3. Run the following command to update the cluster with the new Kubernetes profile:

```
tkgi update-cluster CLUSTER-NAME --kubernetes-profile NEW-KUBERNETES-PROFILE-NAME
```

Where:

- ◊ **CLUSTER-NAME** is the name of the existing Kubernetes cluster.
- ◊ **NEW-KUBERNETES-PROFILE-NAME** is the name of the new Kubernetes profile you want to apply to the cluster.

## Kubernetes Profile Use Cases

Kubernetes profiles let you customize Kubernetes configuration parameters at the time of cluster creation. Use cases for Kubernetes profiles include:

| Topic                               | Description                                                                                                      |
|-------------------------------------|------------------------------------------------------------------------------------------------------------------|
| Encrypt Secrets in an etcd Database | Use an encryption provider to encrypt secrets in a cluster's etcd database.                                      |
| Admission Control: ResourceQuota    | Use the <code>ResourceQuota</code> admission control plugin to restrict incoming requests by resource usage.     |
| Set Service Node Port Range         | Use <code>service-node-port-range</code> to specify an IP range for <code>NodePort</code> services.              |
| Adding an OIDC Provider             | Customize a cluster's OIDC provider by deploying a <code>dex</code> connector or other OIDC provider to its pod. |
| Restrict Request Header Names       | Set <code>requestheader-allowed-names</code> for Apiserver client authentication.                                |
| Modify the Service Cluster IP Range | Change the service cluster IP range.                                                                             |

## Admission Control: ResourceQuota

To create a Kubernetes profile that includes the `ResourceQuota` admission control plugin:

- Follow the [Create a Kubernetes Profile](#) instructions.
- Include the following `customizations` in your profile configuration file:

```
"customizations": [
 {
 "component": "kube-apiserver",
 "arguments": {
 "enable-admission-plugins": PLUGINS-LIST
 }
 }
],
```

Where:

- `PLUGINS-LIST` is one of the following:
  - The string `"ResourceQuota"`.
  - A comma-delimited string list of validated plugins that includes `ResourceQuota`.

For more information, see [ResourceQuota](#) in the Kubernetes documentation.

## Set Service Node Port Range

To create a Kubernetes profile that uses `service-node-port-range` for `NodePort` type services:

- Follow the [Create a Kubernetes Profile](#) instructions.
- Include the following `customizations` in your profile configuration file:

```
"customizations": [
 {
```

```

 "component": "kube-apiserver",
 "arguments": {
 "service-node-port-range": PORT-RANGE
 }
 },
],

```

Where `PORT-RANGE` is a CIDR notation IP range from which to assign service cluster IPs, such as `30000-40000`.

If the specified `PORT-RANGE` is not valid, the `tkgi create-k8s-profile` command returns an error `invalid value for service-node-port-range`.

For more information, see [Type NodePort](#) in the Kubernetes documentation.

## Restrict Request Header Names

To create a Kubernetes profile that uses `requestheader-allowed-names` for Apiserver client authentication:

- Follow the [Create a Kubernetes Profile](#) instructions.
- Include the following `customizations` in your profile configuration file:

```

"customizations": [
{
 "component": "kube-apiserver",
 "arguments": {
 "requestheader-allowed-names": COMMON-NAMES
 }
},
],

```

Where `COMMON-NAMES` is a string list of valid Common Name values in the signed client certificate, such as `"cn1.com,c2.com"`.

For more information, see [Kubernetes Apiserver Client Authentication](#) in the Kubernetes documentation.

## Modify the Service Cluster IP Range

To create a Kubernetes profile that modifies the service cluster IP range:

- Follow the [Create a Kubernetes Profile](#) instructions.
- Include the following `customizations` in your profile configuration file:

```

"customizations": [
{
 "component": "kube-apiserver",
 "arguments": {
 "service-cluster-ip-range": IP-RANGE
 }
},
],

```

Where `IP-RANGE` is a CIDR notation IP range from which to assign service cluster IPs. The IP

range can be a maximum of two dual-stack CIDRs and must not overlap with any IP ranges assigned to nodes or pods.

For more information, see [kube-apiserver Options](#) in the Kubernetes documentation.

## Using Network Profiles (NSX-T Only)

This topic describes how VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) cluster managers can manage and use network profiles with Kubernetes clusters provisioned by TKGI on vSphere with NSX-T integration.

Network profiles let you customize NSX-T configuration parameters.

## Prerequisite

TKGI supports network profiles on TKGI on vSphere with NSX-T only.

To work with TKGI network profiles you must be either a cluster manager or cluster administrator:

- To create or delete a network profile, you must be a cluster administrator: `pks.clusters.admin`.
- To use a network profile, you must be a cluster manager: `pks.clusters.manage` or a cluster administrator: `pks.clusters.admin`.

## Overview

You can use network profiles to customize your TKGI Kubernetes clusters on vSphere with NSX-T. For information on when to use network profiles, see [Network Profile Use Cases](#) below.

TKGI cluster managers can apply network profiles to clusters:

To list the available network profiles:

- [List Network Profiles](#)

To apply a network profile to a cluster:

- [List Network Profiles](#)
- [Create a Cluster with a Network Profile](#)
- [Assign a Network Profile to an Existing Cluster](#)
- [Update an Existing Network Profile](#)

TKGI cluster administrators can create and manage network profiles. To create or manage network profiles see the following in [Creating and Managing Network Profiles](#):

- [Create a Network Profile](#)
- [Update an Existing Network Profile](#)
- [Delete a Network Profile](#)

## List Network Profiles

To list available network profiles:

- Run the following command:

```
tkgi network-profiles
```

For example:

```
$ tkgi network-profiles
```

| Name               | Description                                                            |
|--------------------|------------------------------------------------------------------------|
| lb-profile-medium  | Network profile for medium size NSX-T load balancer                    |
| small-routable-pod | Network profile with small load balancer and two routable pod networks |

## Create a Cluster with a Network Profile

You can assign a network profile to a TKGI-provisioned Kubernetes cluster at the time of cluster creation.

To create a Kubernetes cluster with a network profile:

- If you do not have a network profile with the desired configuration, have a TKGI cluster administrator define and create a new network profile. For more information, see [Create a Network Profile](#) in *Creating and Managing Network Profiles*.
- Choose a network profile for the cluster. See [List Network Profiles](#).
- To create the cluster, run the following command:

```
tkgi create-cluster CLUSTER-NAME --external-hostname HOSTNAME --plan PLAN-NAME
--network-profile NETWORK-PROFILE-NAME
```

Where:

- CLUSTER-NAME** is a unique name for your cluster.



**Note:** Use only lowercase characters when naming your cluster if you manage your clusters with Tanzu Mission Control (TMC). Clusters with names that include an uppercase character cannot be attached to TMC.

- HOSTNAME** is your external hostname used for accessing the Kubernetes API.
- PLAN-NAME** is the name of the TKGI plan you want to use for your cluster.
- NETWORK-PROFILE-NAME** is the name of the network profile you want to use for your cluster.

## Assign a Network Profile to an Existing Cluster

TKGI supports assigning a network profile to an existing cluster.

To assign a network profile to a cluster that does not have a network profile already applied:

1. If you do not have a network profile with the desired configuration, have a TKGI cluster administrator define and create a new network profile. For more information, see [Create a Network Profile](#) in *Creating and Managing Network Profiles*.
2. Choose a network profile for the cluster. See [List Network Profiles](#).
3. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
4. To apply the network profile to the cluster, run the following command:

```
tkgi update-cluster CLUSTER-NAME --network-profile NEW-NETWORK-PROFILE-NAME
```

Where:

- ◊ `CLUSTER-NAME` is the name of the existing Kubernetes cluster.
- ◊ `NEW-NETWORK-PROFILE-NAME` is the name of the new network profile you want to apply to the cluster.



**Note:** When you use `tkgi update-cluster` to update an existing cluster, the attached network-profile must consist of only updatable settings.

## Update an Existing Network Profile

The use cases for updating an existing network profile are limited to adding to or changing the order of Pod IP Blocks on your existing cluster. For more information, see [Customizing Pod Networks](#).

Only TKGI cluster administrators can modify an existing network profile. For information on updating an existing network profile, see [Update an Existing Network Profile](#) in *Creating and Deleting Network Profiles*.

## Network Profile Use Cases

Network profiles let you customize configuration parameters for Kubernetes clusters provisioned by TKGI on vSphere with NSX-T.

You can apply a network profile to a Kubernetes cluster for the following scenarios:

| Topic                                    | Description                                                                                                     |
|------------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| <a href="#">Size a Load Balancer</a>     | Customize the size of the NSX-T load balancer service that is created when a Kubernetes cluster is provisioned. |
| <a href="#">Customizing Pod Networks</a> | Customize Kubernetes Pod Networks, including adding pod IP addresses, subnet size, and routability.             |
| <a href="#">Customize Node Networks</a>  | Customize Kubernetes Node Networks, including the IP addresses, subnet size, and routability.                   |

| Topic                                           | Description                                                                                                                  |
|-------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------|
| Customize Floating IP Pools                     | Specify a custom floating IP pool.                                                                                           |
| Configure Bootstrap NSGroups                    | Specify an NSX-T Namespace Group where the Kubernetes control plane nodes will be added to during cluster creation.          |
| Configure Edge Router Selection                 | Specify the NSX-T Tier-0 router where Kubernetes node and Pod networks will be connected to.                                 |
| Specify Nodes DNS Servers                       | Specify one or more DNS servers for Kubernetes clusters.                                                                     |
| Configure DNS for Pre-Provisioned IPs           | Configure DNS lookup of the Kubernetes API load balancer or ingress controller.                                              |
| Configure the TCP Layer 4 Load Balancer         | Configure layer 4 TCP load balancer settings; use a third-party load balancer.                                               |
| Configure the HTTP/S Layer 7 Ingress Controller | Configure layer 7 HTTP/S ingress controller settings; use third-party ingress controller.                                    |
| Define DFW Section Markers                      | Configure top or bottom section markers for explicit DFW rule placement.                                                     |
| Configure NCP Logging                           | Configure NCP logging.                                                                                                       |
| Dedicated Tier-1 Topology                       | Use dedicated Tier-1 routers, rather than a shared router, for each cluster's Kube node, Namespace, and NSX-T load balancer. |

## Using BOSH VM Extensions

This topic describes how to configure Kubernetes clusters with BOSH VM extensions using VMware Tanzu Kubernetes Grid Integrated Edition (TKGI).

### Overview

BOSH VM extensions are VM configurations stored in the BOSH cloud config. BOSH VM extensions allow you to specify IaaS-specific configurations for your VMs such as custom security group and load balancer configurations.

TKGI supports configuring Kubernetes clusters using BOSH VM extensions. You can use BOSH VM extensions to configure TKGI-provisioned Linux and Windows clusters.

For information about BOSH VM Extensions, see [VM Extensions Block](#) in *Usage in the Cloud Foundry BOSH documentation*.



**Warning:** Configure VM Extensions only if you are already familiar with BOSH VM Extensions. If you use VM extensions, you might accidentally override more settings than you intend. For example, if you use a VM extension to add tags, the default tags are removed from all instance groups.

## Create a Cluster Using VM Extensions

To create a new Kubernetes cluster configured with VM extensions:

1. Create a VM extensions configuration file. For more information, see [Create a Cluster](#)

Configuration File for BOSH VM Extensions below.

- To create a cluster using a VM extensions file:

```
tkgi create-cluster CLUSTER-NAME --config-file CONFIG-Filename
```

Where:

- CLUSTER-NAME is the name of the cluster to create.
- CONFIG-Filename is the name of the VM extension configuration file created above.

## Configure a Cluster Using VM Extensions

To configure an existing Kubernetes cluster with VM extensions:

- If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
- Create a VM extensions configuration file. For more information, see [Create a Cluster Configuration File for BOSH VM Extensions](#) below.
- To modify a cluster using a VM extensions file:

```
tkgi update-cluster CLUSTER-NAME --config-file CONFIG-Filename
```

Where:

- CLUSTER-NAME is the name of the cluster to modify.
- CONFIG-Filename is the name of the VM extension configuration file created above.

## Create a Cluster Configuration File for BOSH VM Extensions

To create a VM extensions configuration file in JSON format:

- Create a new configuration file containing the following content:

- vSphere VM Extension Configuration File Template:**

```
{
 "instance_groups": [
 {
 "name": "master",
 "vm_extension": {
 "vmx_options": {
 "disk.enableUUID": "1"
 },
 "nsxt": {
 "ns_groups": [NSX-NS-GROUPS]
 }
 }
 },
 {
 "name": "WORKER-NAME",
 "vm_extension": {
 "vmx_options": {
 "disk.enableUUID": "1"
 }
 }
 }
]
}
```

```

 }
 },
 {
 "name": "NODE-POOL-NAME",
 "vm_extension": {
 "vmx_options": {
 "disk.enableUUID": "1"
 }
 }
 }
]
}

```

- ❖ **Public Cloud VM Extension Configuration File Template:**

```

{
 "instance_groups": [
 {
 "name": "master",
 "vm_extension": {
 "vmx_options": {
 "disk.enableUUID": "1"
 }
 }
 },
 {
 "name": "WORKER-NAME",
 "vm_extension": {
 "vmx_options": {
 "disk.enableUUID": "1"
 }
 }
 }
]
}

```

Where:

- ❖ **NSX-NS-GROUPS** (vSphere with VMware NSX Only) is a comma-separated list of NS Group names that the instances should belong to.
  - ❖ **NODE-POOL-NAME** (vSphere Only) is the instance group name for the node pool VM extensions. The node pool must be configured in the cluster's compute profile. For Linux clusters, the node pool name must be prefixed `worker-` and for Windows Workers `windows-worker-`. For example: `worker-tiny-1` and `windows-worker-tiny-1`.
  - ❖ **WORKER-NAME** is `worker` for Linux clusters and `windows-worker` for Windows Worker clusters.
2. Update the `vm_extension` and `vmx_options` parameters of each instance group with the custom BOSH VM extensions and VMX options to apply to the Kubernetes cluster.



**Warning:** Volumes will not attach to your nodes if you do not include the `disk.enableUUID vmx_options` parameter in your configuration.

For example:

```
{
 "instance_groups": [
 {
 "name": "master",
 "vm_extension": {
 "cpu_hot_add_enabled": "true",
 "nsxt": {
 "ns_groups": ["master-2"]
 },
 "vmx_options": {
 "disk.enableUUID": "1"
 }
 }
 },
 {
 "name": "windows-worker",
 "vm_extension": {
 "cpu_hot_add_enabled": "true",
 "vmx_options": {
 "disk.enableUUID": "1"
 }
 }
 },
 {
 "name": "windows-worker-tiny-1",
 "vm_extension": {
 "cpu_hot_add_enabled": "true",
 "vmx_options": {
 "ctkEnabled": "TRUE",
 "disk.enableUUID": "1"
 }
 }
 }
]
}
```

The supported BOSH VM extensions are specific to each IaaS. For the names of the cloud properties you can use in your VM extension configurations, see the following topics in *Usage* in the Cloud Foundry BOSH documentation:

- ◊ For AWS, see: [VM Types / VM extensions](#).
- ◊ For Azure, see: [VM Types / VM extensions](#).
- ◊ For GCP, see: [VM Types / VM extensions](#).
- ◊ For vSphere, see: [VM Types / VM extensions](#).

## Remove BOSH VM Extensions From a Cluster

To remove a VM extensions instance group from a cluster:

1. Create a VM extension configuration file containing the following:

```
{
 "instance_groups": [
```

```
{
 "name": "INSTANCE-GROUP",
 "vm_extension": {
 }
}
]
```

Where:

- ◊ **INSTANCE-GROUP** is the name of the instance group VM extensions to be removed. Specify either `master`, `worker`, `windows-worker` or a node pool VM extension instance group name.

2. To remove the VM extensions instance group using the CLI:

```
tkgi update-cluster CLUSTER-NAME --config-file CONFIG-FILENAME
```

Where:

- ◊ **CLUSTER-NAME** is the name of the cluster to remove VM extensions from.
- ◊ **CONFIG-FILENAME** is the name of the VM extension configuration file created above.

## Viewing Cluster Lists

Follow the steps below to view the list of deployed Kubernetes cluster with the TKGI CLI.

1. On the command line, run the following command to log in:

```
tkgi login -a TKGI-API -u USERNAME -k
```

Where:

- ◊ **TKGI-API** is the domain name for the TKGI API that you entered in **Ops Manager > Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN)**. For example, `api.tkgi.example.com`.
- ◊ **USERNAME** is your user name.

See [Logging in to Tanzu Kubernetes Grid Integrated Edition](#) for more information about the `tkgi login` command.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

2. Run the following command to view the list of deployed clusters, including cluster names and status:

```
tkgi clusters
```

For example:

```
$ tkgi clusters
```

| Name                                 | Plan Name          | UUID               |
|--------------------------------------|--------------------|--------------------|
| Status                               | Action cluster-one | small              |
| -hea6-3h7g1o04kh0e                   | succeeded          | CREATE cluster-two |
| 951547d1-67kg-9631-bju8-7h9s3o98br0q | succeeded          | CREATE             |

## Viewing Cluster Details

Follow the steps below to view the details of an individual cluster using the TKGI CLI.

1. On the command line, run the following command to log in:

```
tkgi login -a TKGI-API -u USERNAME -k
```

Where:

- TKGI-API is the domain name for the TKGI API that you entered in **Ops Manager > Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN)**. For example, [api.tkgi.example.com](http://api.tkgi.example.com).
- USERNAME is your user name.

See [Logging in to Tanzu Kubernetes Grid Integrated Edition](#) for more information about the `tkgi login` command.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

2. Run the following command to view the details of an individual cluster:

```
tkgi cluster CLUSTER-NAME
```

Where `CLUSTER-NAME` is the unique name for your cluster.

3. Run the following command to view additional details of an individual cluster, including NSX-T network details and Kubernetes settings details:

```
tkgi cluster CLUSTER-NAME -details
```

Where `CLUSTER-NAME` is the unique name for your cluster.

For example:

```
$ tkgi cluster my-cluster --details TKGI Version: 1.9.0-b
 uid.1 Name: my-cluster K8s Version: 1.18.8
 Plan Name: small UUID: 4b1a9
 b6d-3594-4cad-ad0f-22043fb26480 Last Action: CREATE Last
 Action State: succeeded Last Action Description: Instance provisioning completed
 Kubernetes Master Host: example-hostname Kubernetes Master Port: 8443 Worker Nodes: 3 Kubernetes Master IP(s): 10.197.100.130 Network Profile Name:

NSXT Network Details: Load Balancer Size (lb_size)
 e: "small" Nodes DNS Setting (nodes_dns):
 [no-nat] Node IP addresses are routable [node_routable]: false Nodes subnet prefix
 (node_subnet_prefix): 24 POD IP addresses are routable
 e [no-nat] (pod_routable): false PODs subnet prefix
 (pod_subnet_prefix): 24 NS Group ID of master VMs
 (master_vms_nsgroup_id): "" Tier 0 Router identifier
 (t0_router_id): "1e8371ac-1718-4617-8734-3975c6cd373b" Floating IP Pool identifiers (fip_pool_ids):
 ["901341c7-2e14-49d0-a3c1-66748664a062"] Node IP block identifiers (node_ip_block_ids):
 ["c5f0eb13-9691-4170-a9cd-c988f336ebd2"] POD IP block identifiers (pod_ip_block_ids):
 ["fead2c9a-96e8-4c5f-98a3-e797f06bc8d4"] Kubernetes Settings Details: Set by Plan: Kubelet Node Drain timeout (mins) (kubelet-drain-timeout): 0 Kubelet Node Drain grace-period (seconds) (kubelet-drain-grace-period): 10 Kubelet Node Drain force (kubelet-drain-force): true Kubelet Node Drain force-node (kubelet-drain-force-node): false Kubelet Node Drain ignore-daemonsets (kubelet-drain-ignore-daemonsets): true Kubelet Node Drain delete-local-data (kubernetes-drain-delete-local-data): true
```

The following image shows another example of `tkgi cluster CLUSTER-NAME --details` output with NSX-T details:

```
Name: some-cluster-name
Plan Name: some-plan-name
UUID: 7a161098-986a-4a3c-8b02-d21e2a523463
Last Action: CREATE
Last Action State: in progress
Last Action Description: Instance update in progress
Kubernetes Master Host: 85da06e7-bd08-43b8-82bf-6fa946339356.internal
Kubernetes Master Port: 8443
Worker Nodes: 2
Kubernetes Master IP(s): 10.11.12.13, 10.20.30.40
Network Profile Name:

NSXT Network Details:
 Load Balancer Size (lb_size): "small"
 Nodes DNS Setting (nodes_dns): ["192.168.115.1"]
 POD IP addresses are routable [no-nat] (node_routable): false
 (pod_subnet_prefix): 24
 (master_vms_nsgroup_id): ""
 (t0_router_id): "1b0014c1-fdaa-49b2-b2c2-e5a7f6215200"
 Floating IP Pool identifiers (fip_pool_ids): ["21a65cd7-2162-4dc1-ad51-020e617f3595", "d33ec8ca-04ed-4e8e-9a05-af4a2b48d906"]
 Node IP block identifiers (node_ip_block_ids): ["59bc3d66-d5ca-4c02-b509-8b09c950f52c", "59bc3d66-d5ca-4c02-b509-8b09c950f52c"]
 POD IP block identifiers (pod_ip_block_ids): ["d33ec8ca-04ed-4e8e-9a05-af4a2b48d906"]

[Ubuntu-1c(46041) Outgoing]$
```

## Viewing Cluster Plans

Follow the steps below to view information about the available plans for deploying a cluster using the TKGI CLI.

1. On the command line, run the following command to log in:

```
tkgi login -a TKGI-API -u USERNAME -k
```

Where:

- TKGI-API is the domain name for the TKGI API that you entered in **Ops Manager > Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN)**. For example, [api.tkgi.example.com](http://api.tkgi.example.com).
- USERNAME is your user name.

See [Logging in to Tanzu Kubernetes Grid Integrated Edition](#) for more information about the `tkgi login` command.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

2. Run the following command to view information about the available plans for deploying a cluster:

```
$ tkgi plans
```

The response lists details about the available plans, including plan names and descriptions:

| Name | ID | Description | default | Default plan for K8s cluster |
|------|----|-------------|---------|------------------------------|
|------|----|-------------|---------|------------------------------|

## Scaling Existing Clusters

This topic explains how to scale an existing cluster horizontally by adding worker nodes and vertically by changing the size of the node VMs.



**WARNING:** Resize only TKGI clusters that have been upgraded to the current TKGI version.

To change the default number of worker nodes created in new clusters, change your plan's **Worker Node Instances** setting. For more information, see [Plans](#) in the *Installing TKGI* topic for your IaaS.



**WARNING:** Do not change the number of control plane/etcdb nodes for any plan that was used to create currently-running clusters. Tanzu Kubernetes Grid Integrated Edition does not support changing the number of control plane/etcdb nodes for plans with existing clusters.

## Scale Horizontally by Changing the Number of Worker Nodes Using the TKGI CLI

You can use the TKGI CLI to scale an existing cluster by increasing or decreasing the number of worker nodes in the cluster.

To increase or decrease the number of worker nodes on a cluster:

1. On the command line, run the following command to log in:

```
tkgi login -a TKGI-API -u USERNAME -k
```

Where:

- TKGI-API is the domain name for the TKGI API that you entered in **Ops Manager > Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN)**. For example, [api.tkgi.example.com](http://api.tkgi.example.com).
- USERNAME is your user name.

See [Logging in to Tanzu Kubernetes Grid Integrated Edition](#) for more information about the `tkgi login` command.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

2. To view the current number of worker nodes in your cluster, run the following command:

```
tkgi cluster CLUSTER-NAME
```

Where CLUSTER-NAME is the name of your cluster.

3. Run the following command:

```
tkgi update-cluster CLUSTER-NAME --num-nodes NUMBER-OF-WORKER-NODES
```

Where:

- CLUSTER-NAME is the name of your cluster.
- NUMBER-OF-WORKER-NODES is the number of worker nodes that you want to set for the cluster.
  - To scale down your existing cluster, enter a number lower than the current number of worker nodes.
  - To scale up your existing cluster, enter a number higher than the current number of worker nodes. The maximum number of worker nodes you can set is configured in the **Plan** pane of the Tanzu Kubernetes Grid Integrated

Edition tile in Ops Manager.

 **Note:** VMware recommends that you avoid using the `tkgi resize` command to perform resizing operations.

For example:

```
$ tkgi update-cluster my-cluster -num-nodes 5
```

 **Note:** This command might roll additional virtual machines in the cluster, which can affect workloads if the worker nodes are at capacity.

## Scale Vertically by Changing Cluster Node VM Sizes in the TKGI Tile

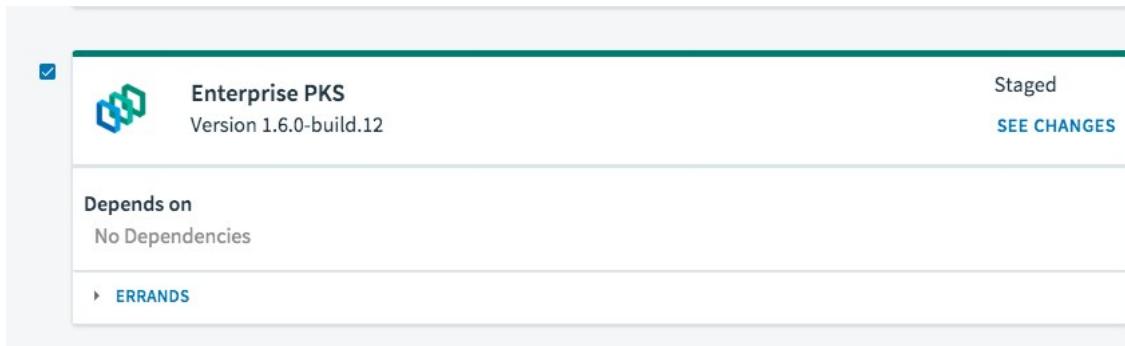
You can scale an existing cluster vertically by changing the size of the control plane or worker node VMs. When you do this, BOSH recreates the VMs sequentially, one cluster at a time, and one node after another within the cluster. For more information, see [VM Sizing for TKGI Clusters](#).

To change the size of a Kubernetes cluster node VM, complete the following steps:

1. Log in to Ops Manager.
2. Select the TKGI tile.
3. Select the plan that is in use by the cluster(s) you want to resize.
4. To change the VM size:
  - For Control Plane nodes, select the desired VM size from the **Master/ETCD VM Type** menu.
  - For Worker nodes, select the desired VM size from the **Worker VM Type** menu.

 **Note:** See [Customize Control Plane and Worker Node VM Size and Type](#) for information on creating a custom VM size for use with a TKGI cluster.

5. Click **Save** to preserve tile changes.
6. At the **Installation Dashboard**, click **Review Pending Changes**.



7. For the TKGI tile, expand the **ERRANDS** list.
8. Select the **Update all clusters errand** if it is not already selected. You must ensure that **Update all clusters errand** is selected so that the cluster deployment manifest is regenerated after the plan is updated.
9. Click **Apply Changes**.

## Tagging Clusters

This topic explains how to tag new and existing clusters using the TKGI CLI.

### Overview

IaaSes provide the ability for customers to “tag” VMs, databases, and other resources with custom labels and metadata values. Apply one or more tags to your clusters to simplify organizing, managing, searching for, and filtering resources within your IaaS-provided management console and other tools:

You can use the TKGI CLI to tag clusters by following the steps in [Tag Your Clusters as They Are Created](#) below.



**Note:** Tanzu Kubernetes Grid Integrated Edition Cluster tagging requires Ops Manager v2.8.0 or later.

## Tag Your Clusters as They Are Created

To apply tags to your cluster’s VMs, include the `--tags` parameter in your `tkgi create-cluster` command line, and specify the desired tags as a comma-delimited list of `key:value` pairs.

```
tkgi create-cluster CLUSTER-NAME --tags "TAGS"
```

Where:

- `CLUSTER-NAME` is the name of the cluster to create.



**Note:** Use only lowercase characters when naming your cluster if you manage your clusters with Tanzu Mission Control (TMC). Clusters with names that include an uppercase character cannot be attached to TMC.

- `TAGS` is a comma-delimited list of `key:value` pairs to apply to the cluster.

For example:

```
$ tkgi create-cluster my-cluster -tags "status:billable"
$ tkgi create-cluster my-cluster -tags "status:non-billable,region:northwest"
$ tkgi create-cluster my-cluster -tags "client:example.com, costcenter:petty"
```

```
cash"
```

## Tag Your Existing Clusters

You can use the TKGI CLI to tag an existing cluster.

To apply tags to your existing cluster's VMs:

1. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
2. Run the `tkgi update-cluster` command line, and specify the `--tags` parameter and a comma-delimited list of `key:value` pairs of the tags to apply to the cluster:

```
tkgi update-cluster CLUSTER-NAME --tags "TAGS"
```

Where:

- ◊ `CLUSTER-NAME` is the name of the cluster to tag.
- ◊ `TAGS` is a comma-delimited list of `key:value` pairs.

For example:

```
$ tkgi update-cluster my-cluster -tags "client:tinymegacorp"
$ tkgi update-cluster my-cluster -tags "client:example.com,costcenter:pettycash"
$ tkgi update-cluster my-cluster -tags "status:non-billable, region:northwest"
```

## Modify Cluster Tags

You can also use `tkgi update-cluster` to modify your cluster's existing tags. When you modify cluster tags you completely replace all of the cluster's existing tags with the specified tags.



**Note:** On Azure, `tkgi update-cluster` cannot remove tags from your IaaS. For more information, see [Azure-Specific Tagging Limitations](#) below.

## Modify Your Existing Tags

To modify your cluster's existing tags do the following:

1. Retrieve the cluster's existing tags list string by running `tkgi cluster`. For information on [tkgi cluster Review Your Tags](#) below.
2. Modify the tags list string by doing one of the following:
  - ◊ To add a new tag, append it to existing tags list string.
  - ◊ To modify an existing tag, modify it within the tags list string.

- To remove an existing tag, delete it from within the tags list string.
3. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
  4. Run the following command:

```
tkgi update-cluster CLUSTER-NAME --tags "TAGS"
```

Where `TAGS` is a comma-delimited list of revised `key:value` pairs.

## Remove All Tags From Your Cluster

To remove all of your cluster's existing tags do the following:

1. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
2. Run the following `tkgi update-cluster --tags` on your command line:

```
tkgi update-cluster CLUSTER-NAME --tags ""
```

Where `CLUSTER-NAME` is the cluster to remove tags from.

## Review Your Tags

To review the tags applied to a cluster, run `tkgi cluster`.

For example:

```
$ tkgi cluster my-cluster Name: my-cluster
Plan Name: large
UUID: 01a234bc-d56e-7f89-01a2-3b4cde5f6789
Last Action: CREATE
Last Action State: succeeded
Last Action Description: Instance provisioning completed
Kubernetes Master Host: my-cluster.example.com
Kubernetes Master Port: 8443
Worker Instances: 3
Kubernetes Master IP(s): 192.168.20.7
Tags: client:tinymegacorp, costcenter:pettycash
```

The `tkgi cluster` function returns only the custom tags you've applied to the cluster using the TKGI

CLI. To display all of the tags applied to your cluster VMs use your IaaS-provided management console.



**Note:** Do not use the IaaS-provided management console to modify your custom tags. Custom tag alterations you've applied via the management console will be overwritten when you next run `tkgi update-cluster`.

## Tagging Rules

The tagging you apply must adhere to the following rules:

- Tag names and values must not include any of the following symbols: ", :, ,.
- Surrounding double quotes are required if there are one or more spaces in your tag list, such as a space after a comma delimiter.
- Tag names and values must adhere to the tagging rules of the IaaS hosting your Tanzu Kubernetes Grid Integrated Edition environment.

For information about IaaS-specific tagging rules see the following:

- Azure: See [Use tags to organize your Azure resources] (<https://docs.microsoft.com/en-us/azure/azure-resource-manager/management/tag-resources>) in the Azure documentation.
- vSphere: See [vSphere Tags and Attributes] (<https://docs.vmware.com/en/VMware-vSphere/6.7/com.vmware.vsphere.vcenterhost.doc/GUID-E8E854DD-AA97-4E0C-8419-CE84F93C4058.html>) in the vSphere documentation.

## Tagging Limitations

Cluster tagging has the following limitations:

### Tags Reserved for BOSH

The BOSH layer applies 10 system-level metadata tags to each cluster, including `deployment`, `director`, `id`, `index`, `instance_group`, `job`, and `name`. These reserved tags impose the following limitations:

- The maximum number of custom tags you can apply to a cluster is 10 less than the maximum number of tags supported by your IaaS.
  - For example: Azure limits tagging to a maximum of 50 tags per entity. Therefore, if your Tanzu Kubernetes Grid Integrated Edition environment is hosted on Azure, apply fewer than 40 custom tags to your clusters.
- You cannot set or change BOSH system-level tags using the TKGI CLI. If you use the TKGI CLI to create tags with those reserved names, they are ignored.
  - When you try to change system-level tags with the TKGI CLI, the output of `tkgi cluster` shows the tag values passed into the `create-cluster` or `update-cluster` command, but BOSH overrides and ignores these settings.

## AWS-Specific Tagging Limitations

For tagging limitations on Amazon Web Services (AWS), see [Tag restrictions in the AWS documentation](#).

## Azure-Specific Tagging Limitations

The following are known tagging limitations specific to Microsoft Azure:

- `tkgi update-cluster` cannot remove tags from your Azure clusters. This limitation is due to an issue in the Azure CPI for BOSH which is used by `tkgi cli` for Azure IaaS tagging.
  - ◊ To remove an IaaS tag from an Azure cluster do the following:
    1. Perform the removal steps described in [Modify Existing Tags](#) above.
    2. Remove unwanted tags through the Azure portal.

For information about additional Azure-specific tagging limitations see [Use tags to organize your Azure resources] (<https://docs.microsoft.com/en-us/azure/azure-resource-manager/management/tag-resources>) in the Azure documentation.

## GCP-Specific Tagging Limitations

Google Cloud Platform (GCP) refers to tags as “labels.” The following are known label limitations specific to Google Cloud Platform (GCP):

- You can assign up to 64 labels (tags) to each resource.
- Label keys and values must:
  - ◊ Be 63 characters or shorter.
  - ◊ Contain only:
    - Lowercase letters (International characters are allowed)
    - Numeric characters
    - Underscores
    - Hyphens
- Label keys must start with a lowercase letter.
- Label keys cannot be empty.

## vSphere-Specific Tagging Limitations

The following are known tagging limitations specific to vSphere:

- `tkgi update-cluster` applies tagging to vSphere entities as vSphere Custom Attributes. This limitation is due to an issue in the vSphere CPI which is used by `tkgi cli` for vSphere IaaS tagging.
  - ◊ vSphere Custom Attribute tagging is applied to VMs only. Disks and other resources are not tagged.
  - ◊ A vSphere Custom Attribute applied to a single VM is also visible on all other VMs,

but as an empty property.

For information about additional vSphere-specific tagging limitations see [vSphere Tags and Attributes] (<https://docs.vmware.com/en/VMware-vSphere/6.7/com.vmware.vsphere.vcenterhost.doc/GUID-E8E854DD-AA97-4EOC-8419-CE84F93C4058.html>) in the vSphere documentation.

## Upgrading Clusters

This topic describes how to upgrade Kubernetes clusters provisioned by VMware Tanzu Kubernetes Grid Integrated Edition (Tanzu Kubernetes Grid Integrated Edition) through the Tanzu Kubernetes Grid Integrated Edition Command Line Interface (TKGI CLI).

For information about how to upgrade TKGI-provisioned clusters through the Tanzu Kubernetes Grid Integrated Edition tile, see *Verify Errand Configuration* in one of the following topics:

- [Upgrading Tanzu Kubernetes Grid Integrated Edition \(Antrea and Flannel Networking\)](#)
- [Upgrading Tanzu Kubernetes Grid Integrated Edition \(NSX-T Networking\)](#)

For conceptual information about Tanzu Kubernetes Grid Integrated Edition upgrades, see [About Tanzu Kubernetes Grid Integrated Edition Upgrades](#).

## Overview

Upgrading a TKGI-provisioned Kubernetes cluster updates the Tanzu Kubernetes Grid Integrated Edition version and the Kubernetes version of the cluster.

TKGI-provisioned Kubernetes clusters upgrade when:

- You upgrade Tanzu Kubernetes Grid Integrated Edition with the **Upgrade all clusters errand** enabled in the **Tanzu Kubernetes Grid Integrated Edition** tile > **Errands**.
- You run `tkgi upgrade-cluster` or `tkgi upgrade-clusters` as described in [Upgrade Clusters](#) below.

For example, running `tkgi upgrade-cluster` upgrades the cluster you specify to your current version of Tanzu Kubernetes Grid Integrated Edition and to the version of Kubernetes that is included with your current version of Tanzu Kubernetes Grid Integrated Edition.



**WARNING:** Do not change the number of control plane/etcdb nodes for any plan that was used to create currently-running clusters. Tanzu Kubernetes Grid Integrated Edition does not support changing the number of control plane/etcdb nodes for plans with existing clusters.

## Prerequisites

Before upgrading TKGI-provisioned Kubernetes clusters:

1. If you have not already done so, install the TKGI CLI for the current TKGI version. For information, see [Installing the TKGI CLI](#).
2. Verify the cluster you are upgrading supports upgrading. For information, see [Verify Your](#)

[Clusters Support Upgrading](#) in the *Upgrade Preparation Checklist for Tanzu Kubernetes Grid Integrated Edition*.

3. Verify that your Kubernetes environment is healthy. For information, see [Verifying Deployment Health](#).
4. If you are upgrading a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
5. Log in to Tanzu Kubernetes Grid Integrated Edition using `tkgi login`. For more information, see [Logging in to Tanzu Kubernetes Grid Integrated Edition](#).

## Upgrade Clusters

You can use the TKGI CLI to upgrade an existing cluster to the current version of TKGI.

To upgrade the TKGI version on individual or multiple clusters:

- [Upgrade a Single Kubernetes Cluster](#)
- [Upgrade Multiple Kubernetes Clusters](#)

To monitor or stop a cluster upgrade, follow the procedures in [Manage Your Kubernetes Cluster Upgrade Job](#) below.

### Upgrade a Single Cluster

The Tanzu Kubernetes Grid Integrated Edition CLI provides `upgrade-cluster` for upgrading an individual Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes cluster.

To upgrade an individual Kubernetes cluster:

1. Run the following command:

```
tkgi upgrade-cluster CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of the Kubernetes cluster you want to upgrade.

For more information about the `tkgi upgrade-cluster` command, see [tkgi upgrade-cluster](#) in the *TKGI CLI* documentation.



**Note:** The nodes in an upgrading cluster are processed serially.

To upgrade multiple clusters, see [Upgrade Multiple Kubernetes Clusters](#) below.

### Upgrade Multiple Clusters

The Tanzu Kubernetes Grid Integrated Edition CLI provides `upgrade-clusters` for upgrading multiple Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes clusters. You can upgrade clusters serially, serially with some clusters designated as canary clusters, or entirely in parallel.

To upgrade multiple Kubernetes clusters:

1. Run the following command:

```
tkgi upgrade-clusters CLUSTER-NAMES
```

Where `CLUSTER-NAMES` is a list of names of the Kubernetes clusters that you want to upgrade.

For more information about the `tkgi upgrade-clusters` command, see [tkgi upgrade-clusters](#) in the *TKGI CLI* documentation.



**Note:** The nodes in an upgrading cluster are always processed serially.

To upgrade a single cluster, see [Upgrade a Single Kubernetes Cluster](#) above.

## Upgrade Clusters in Parallel

To upgrade multiple Kubernetes clusters:

1. Run the following command:

```
tkgi upgrade-clusters --clusters CLUSTER-NAMES --max-in-flight CLUSTER-COUNT --wait
```

Where:

- `CLUSTER-NAMES` is a comma-delimited list of the names of the Kubernetes clusters you want to upgrade.
- `CLUSTER-COUNT` is the maximum number of clusters to upgrade in parallel within an AZ. The `CLUSTER-COUNT` must be less than your TKGI tile > TKGI API Service > **Worker VM Max in Flight** value. For example, if your TKGI tile **Worker VM Max in Flight** value remains the default of `4`, run `upgrade-clusters` with a `-max-in-flight` argument value less than `4`.

Considerations when running `tkgi upgrade-clusters`:

- If an upgrade for a cluster in the `--clusters` list fails, the `tkgi upgrade-clusters` job continues to a subsequent cluster in the list.
- Clusters are upgraded serially if `--max-in-flight` is not set.
- If the count of names in the `--clusters` list is more than the `--max-in-flight` value, the first set of clusters are upgraded in parallel and subsequent clusters are queued. As the initial cluster upgrades complete, the remaining clusters are pulled from the queue and upgraded in parallel.
- To run the cluster upgrade job as a background task, remove the `--wait` argument.



**Note:** The nodes in an upgrading cluster are always processed serially.

For example:

```
$ tkgi upgrade-clusters --clusters k8-cluster-000,k8-cluster-001,k8-cluster-002 --max-in-flight 2 --wait
```

```
You are about to upgrade k8-cluster-000, k8-cluster-001 and k8-cluster-002.
Warning: This operation might be long running and might block further operations on th
```

```
e cluster(s) until complete

Continue? (y/n):y
Your taskID for the upgrade task is: d772aba0-2670-4fba-b26c-044b19d6ab60
Started upgrading cluster: k8-cluster-000
Started upgrading cluster: k8-cluster-001
Finished upgrading cluster: k8-cluster-000
Started upgrading cluster: k8-cluster-002
Finished upgrading cluster: k8-cluster-001
Finished upgrading cluster: k8-cluster-002
Upgrade task d772aba0-2670-4fba-b26c-044b19d6ab60 is done.
```

## Upgrade Clusters With Canaries

To upgrade multiple clusters and automatically stop upgrading clusters if a cluster upgrade fails, specify your cluster list as canary clusters. You can specify one or more clusters as canary clusters.

To upgrade multiple clusters with one or more canary clusters:

1. Run the following command:

```
tkgi upgrade-clusters --canaries CANARY-CLUSTER-NAMES --clusters CLUSTER-NAMES
--wait
```

Where:

- **CANARY-CLUSTER-NAMES** is a comma-delimited list of the names of the Kubernetes clusters you want to upgrade as canary clusters.
- **CLUSTER-NAMES** is a comma-delimited list of Kubernetes clusters to upgrade if all canary clusters successfully upgrade.



**Note:** The `-clusters` argument is required.

Considerations when running `tkgi upgrade-clusters` with a `--canaries` list:

- The clusters specified in the `--canaries` list are upgraded prior to upgrading the clusters in your `--clusters` list.
- If a canary cluster upgrade fails, the entire `tkgi upgrade-clusters` job stops.
- If a `--clusters` list cluster upgrade fails, the `tkgi upgrade-clusters` job continues to a subsequent cluster in the list.
- To configure `tkgi upgrade-clusters` to stop for any cluster upgrade failure, specify only one cluster in your `-clusters` list and the remaining clusters in your `-canaries` list.
- Canary clusters are always upgraded serially. To upgrade clusters in the `--clusters` list in parallel, see [Upgrade Clusters in Parallel](#) above.
- To run the cluster upgrade job as a background task, remove the `--wait` argument.



**Note:** The nodes in an upgrading cluster are always processed serially.

For example:

```
$ tkgi upgrade-clusters --canaries k8-cluster-dev,k8-cluster-000,k8-cluster-001 --clusters k8-cluster-002 --wait

You are about to upgrade k8-cluster-dev k8-cluster-000, k8-cluster-001 and k8-cluster-002.
Warning: This operation might be long running and might block further operations on the cluster(s) until complete

Continue? (y/n):y
Your taskID for the upgrade task is: ce31a1bb-380a-453f-afa0-835ffalce6ac
Started upgrading cluster: k8-cluster-000
Upgrading cluster succeeded: k8-cluster-000
Started upgrading cluster: k8-cluster-001
Upgrading cluster succeeded: k8-cluster-001
Started upgrading cluster: k8-cluster-dev
Upgrading cluster failed: k8-cluster-dev
Upgrade task ce31a1bb-380a-453f-afa0-835ffalce6ac is done.
```

## Manage Your Cluster Upgrade Job

You can use the TKGI CLI to monitor and manage your Tanzu Kubernetes Grid Integrated Edition-provisioned Kubernetes cluster upgrade jobs.

### Monitor Your Clusters

To review the status of the actions being performed on your clusters, run the following command:

```
tkgi clusters
```

For example:

```
$ tkgi clusters

Upgrade is available to TKGI Version: 1.9.0-build.1

TKGI Version Name k8s Version Plan Name UUID
 Status Action
1.9.0-build.1 k8-cluster-000 1.18.8 small 9527ebaa-e2fa-422
f-a52b-de3c3f0e39a4 succeeded UPGRADE

1.9.0-build.1 k8-cluster-001 1.18.8 small 9527ebaa-e2fa-422
f-a52b-de3c3f0e39a4 failed UPGRADE

1.9.0-build.1 k8-cluster-002 1.18.8 small 9527ebaa-e2fa-422
f-a52b-de3c3f0e39a4 in progress UPGRADE

1.9.0-build.1 k8-cluster-003 1.18.8 small 9527ebaa-e2fa-422
f-a52b-de3c3f0e39a4 queued UPGRADE
```

## Monitor Your Cluster Upgrade Job

To review the status of your `upgrade-clusters` job, run the following command:

```
tkgi task TASKID
```

Where `TASKID` is the ID of the task that was returned when you ran `tkgi upgrade-clusters`.

For example:

```
$ tkgi task ce31a1bb-380a-453f-afa0-835ffalce6ac
Your upgrade task is: done
Name Status Start time End time
 isCanary

k8-cluster-000 succeeded Mon, 14 Oct 2019 12:00:00 PDT Mon, 14 Oct 2019 12
:19:54 PDT true

k8-cluster-001 failed Mon, 14 Oct 2019 12:20:00 PDT -
 true
```

## Stop Your Cluster Upgrade Job

To cancel a running `upgrade-clusters` job, run the following TKGI CLI command:

```
tkgi cancel-task TASKID
```

Where `TASKID` is the ID of the task that was returned when you ran `tkgi upgrade-clusters`.



**Warning:** `tkgi cancel-task` does not cancel cluster upgrades currently in progress. This command only cancels a job's pending cluster upgrades.

## After Upgrading Clusters

Complete the following optional procedures after you have upgraded your cluster:

- [Upgrade Velero](#)
- [Restore Cluster Sizing](#)

### Upgrade Velero

TKGI v1.14 uses Velero v1.8.1. You must upgrade Velero to v1.8.1 on all of your existing clusters.

To upgrade Velero:

1. Download [Velero v1.8.1](#) from VMware Customer Connect.
2. Complete the steps in [Upgrading to Velero 1.8](#) in the Velero documentation.



**Warning:** Ensure you have updated the Velero custom resource definitions on your

clusters as described in [Instructions in Upgrading to Velero 1.8](#) in the Velero documentation.

## (Optional) Restore Cluster Sizing

If you scaled your cluster up for the upgrade and you prefer to restore your cluster to its original sizing, you can now scale the cluster back down to its previous configuration. VMware recommends that you not scale down your clusters and continue to run them with recommended configurations, reducing the chance of a future outage.

## Adding an OIDC Provider

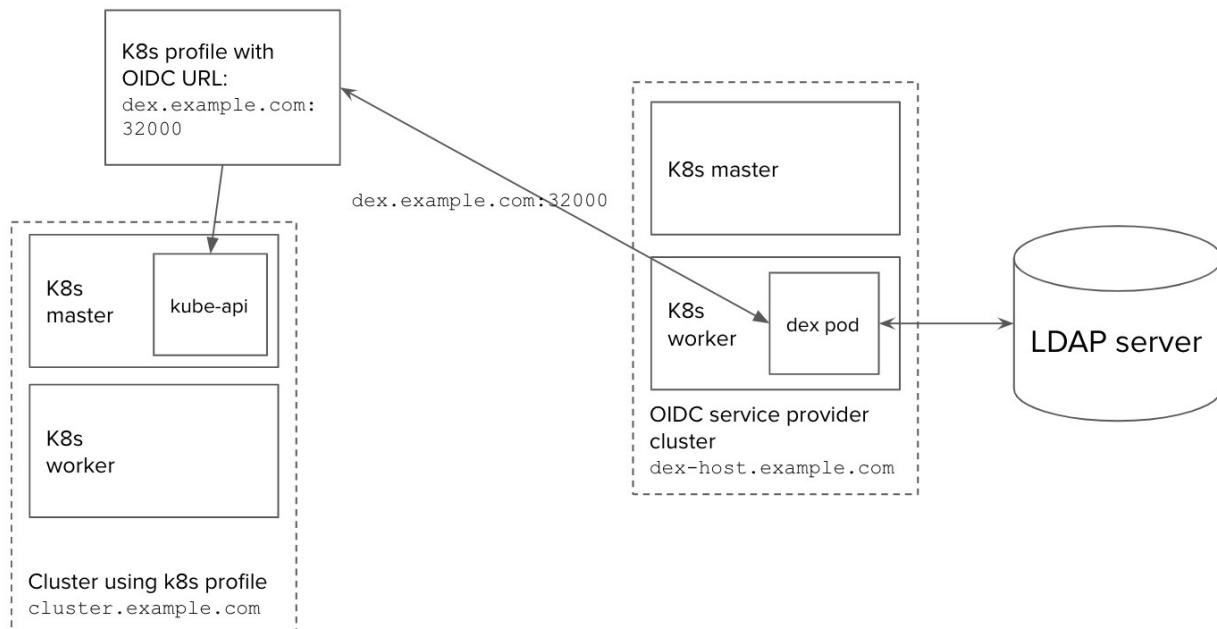
This topic explains how you can use a Kubernetes profile in Tanzu Kubernetes Grid Integrated Edition (TKGI) to override the default Identity Provider (IDP).

## Overview

The TKGI **UAA** pane configures a default IDP for all the clusters that TKGI creates. You can use a Kubernetes profile to override this default IDP.

The Kubernetes profile applies a custom OIDC-compatible IDP to a cluster by deploying an OIDC connector as a service pod on the cluster.

The following diagram provides an overview of how this configuration works:



- **OIDC service provider cluster** with external hostname `dex-host.example.com`
  - Hosts a dex pod that accesses the LDAP server
  - Publishes its OIDC service to `dex.example.com:32000`
- **OIDC Kubernetes profile** has the URL of the OIDC service
- **Host cluster** with external hostname `cluster.example.com`
  - Uses the OIDC Kubernetes profile

- Calls the OIDC service at `dex.example.com:32000` to authenticate the user whenever a user requests an app hosted on the cluster

The Kubernetes profile in this topic deploys `dex` as an OIDC provider, but you can use any OIDC service.

For more information and other uses of Kubernetes profiles, see [Using Kubernetes Profiles](#).

## Prerequisites

To use UAA as your OIDC provider, the TKGI API **Certificate to secure the TKGI API** field on the TKGI tile must be a proper certificate chain and have a SAN field. For more information, see configuring [TKGI API](#) in the *Installing TKGI* topic for your IaaS.

## Configure a Custom OIDC Provider

To configure a custom OIDC provider for TKGI clusters, complete the following:

1. Set Up Dex Workload
2. Set Up Communication Path
3. Deploy and Expose Dex
4. Create Kubernetes Profile
5. Create Cluster
6. Test Cluster Access

## Set Up Dex Workload

To configure `dex` as an OIDC provider for an LDAP directory:

1. Create a cluster in TKGI for installing dex as a pod:

```
tkgi create-cluster dex -p small -e dex-host.example.com
```

2. Run `tkgi cluster` for the cluster and record its [Kubernetes Master IP](#) address.

For example:

```
$ tkgi cluster dex

TKGI Version: 1.9.0-build.1

Name: dex

K8s Version: 1.24.3

Plan Name: small

UUID: dbe1d880-478f-4d0d-bb2e-0da3d9641f0d
```

```
Last Action: CREATE
Last Action State: succeeded
Last Action Description: Instance provisioning completed
Kubernetes Master Host: dex-host.example.com
Kubernetes Master Port: 8443
Worker Nodes: 1
Kubernetes Master IP(s): 10.0.11.11
Network Profile Name:
Kubernetes Profile Name:
Tags:
```

3. Add the `Kubernetes Master IP` address to your local `/etc/hosts` file.
4. Populate your `~/.kube/config` with context for dex:

```
tkgi get-credentials dex
```

5. Switch to the `admin` context of the dex cluster:

```
kubectl config use-context dex
```

6. To deploy a dex workload on a Kubernetes cluster, follow the steps in [Deploying dex on Kubernetes](#) in the dex GitHub repo.
  - ◊ To create a dex deployment that connects to an LDAP server, see [dex/examples/ldap/config-ldap.yaml](#) in the dex OIDC GitHub repository.

## Set Up Communication Path

To set up `\etcd\hosts` and TLS so that clusters can access dex securely:

1. Add the `/etc/hosts` entry for the public IP and the hostname `dex.example.com` on your local workstation. This lets you retrieve a token to access your OIDC-profile cluster later.

```
10.0.11.11 dex.example.com
```

2. To generate TLS assets for the dex deployment, complete the steps in [Generate TLS assets](#) in the dex documentation.
3. To add the generated TLS assets to the cluster as a secret, complete the steps in [Create cluster secrets](#) in the dex documentation.

## Deploy and Expose Dex

To run dex as a local service within a pod and exposes its endpoint via an IP address:

1. On a Kubernetes cluster, deploy dex using the instructions above.
2. Wait for the deployment to succeed.
3. Expose the dex deployment as a service named `dex-service`:

```
kubectl expose deployment dex --type=LoadBalancer --name=dex-service
```

For example:

```
$ kubectl expose deployment dex -type=LoadBalancer -name=dex-service
> service/dex-service exposed
```

4. This should create a dex service with a public IP address that clusters can use as an OIDC issuer URL. Retrieve the IP address by running:

```
kubectl get services dex-service
```

5. Add the IP of the dex service to your `/etc/hosts`:

```
35.222.29.10 dex.example.com
```

- Ensure that you map the dex service to `dex.example.com`, which the dex binary expects as `issuer_url` and for TLS handshakes.
- For this example, we set up the issuer URL as `https://dex.example.com:32000`.

## Create Kubernetes Profile

To create a Kubernetes profile that lets a cluster's `kube-api-server` connect to the dex service:

1. Create a Kubernetes profile `/tmp/profile.json` containing your custom OIDC settings under the `kube-apiserver` component.

For example:

```
$ cat /tmp/profile.json
{
 "name": "oidc-config",
 "description": "Kubernetes profile with OIDC configuration",
 "customizations": [
 {
 "component": "kube-apiserver",
 "arguments": {
 "oidc-client-id": "example-app",
 "oidc-issuer-url": "https://dex.example.com:32000",
 "oidc-username-claim": "email"
 },
 "file-arguments": {
 "oidc-ca-file": "/tmp/oidc-ca.pem"
 }
 }
]
}
```

}

Of all the supported `kube-apiserver` flags, the following are specific to OIDC:

- ◊ In the `arguments` block:
  - `oidc-issuer-url`: Set this to "`https://dex.example.com:32000`".
  - `oidc-client-id`
  - `oidc-username-claim`: Set this to "`email`" for testing with the example app below.
  - `oidc-groups-claim`
- ◊ In the `file-arguments` block:
  - `oidc-ca-file`: Set this to a path in the local file system that contains a CA certificate file. The certificate must be a proper certificate chain and have a SAN field.

For more information on `kube-apiserver` flags, see `kube-apiserver` in the Kubernetes documentation.

## 2. Create the profile:

```
tkgi create-kubernetes-profile /tmp/profile.json
```

In the example above, the file-path `/tmp/oidc-ca.pem` points to a CA certificate on the local file system, and the `tkgi create-kubernetes-profile` command sends this certificate to the API server when it creates the profile.

## Create Cluster

To create a cluster using the Kubernetes profile created above:

### 1. Run the following:

```
tkgi create-cluster cluster-with-custom-oidc -e cluster.example.com -p small --kubernetes-profile oidc-config
```



**Note:** Use only lowercase characters when naming your cluster if you manage your clusters with Tanzu Mission Control (TMC). Clusters with names that include an uppercase character cannot be attached to TMC.

### 2. Confirm the cluster has custom OIDC settings from the profile.

## Test Cluster Access

To test that the cluster uses the OIDC provider to control access, install an app, generate an ID token, and test the cluster.

You can use an example app from the dex repo or test with a full-fledged application, such as [Gangway](#), instead of the example app.

To test that the cluster uses the OIDC provider:

1. To install an example app, complete the steps to install the `example-app` in [Logging into the cluster](#) in the dex documentation.
2. Run the dex example app:

```
./bin/example-app --issuer https://dex.example.com:32000
--issuer-root-ca /tmp/ca.pem
```

- The example app only provides the `email` scope.

3. To fetch the token, complete the steps to generate an ID token in [Logging into the cluster](#) in the dex documentation.
4. Log in using the **Log in with Email** option and enter the email and password of an account in your OIDC IDP.
5. A page appears listing the ID Token, Access Token, Refresh Token, ID Token Issuer (`iss` claim), and other information.

#### ID Token:

```
eyJhbGciOiJSUzI1NiIsImtpZC16IjA3MDQyOWI0YzBhYWQ2MDC4YTEwMjVkJzYzllMDU2OGJlZjQ2ZWMiFo.eyJpc3MiOiJodHRwczovL2RleC5leGFtcGxJLmNvbTozMjAwMCIsInN1YiI6IkNpUXlNRlkwTlRsbE1dMWxnekJpTRsa01XWKRPGVs0WXkve1p1VmtOMlkzTmsallgRNCR3hrWvhBlwiYXVkijoizXhhXBsZSlhcHailC1leHaijo181ODM1MzIwNDcsImlhdcI6MTU4MzQ0NTY0NywiYXRFaGZzaC16IkJfWnVFUzliaDZ6eQdmZk10MVF6d1E1LCJlbWFpbC16ImFseYWS5QjURlc3Q1Y29tLiwiZWlhaWxfdmvyaNwzPzWQioOnRyWUsIm5hbwUiOijhbcGfuYSJ9.mz34p2AS8V2YJCUCU_YPkwjHnO8jWxfjFFElgbZJR7VoeAX3z1lnuxnUpcAt0OrvvkkPs7dg3JaX7_yP1_ekOWJInQacQowVB72vfI3hMexLG5810gMdoVrgbJpnwKX7phpP4ThYuolVALirR6QoQnjjbkBRMUjCeMj_ChcXdfg2dp7yVontsdcmzHMTMJvjzTRGMkfwjnSCF7obXPAofgnENKuuhTcmjk7ALAkjy8Ltb8B4od01MN_ZpUmMN3uRQYSRHnP9WfBL6hpsB0QqiiAuCDWAZycLyTR_UjofAeQd2by80lf3wVUTESECoMtI8KHsjpUEMVY8dXqUEQ
```

#### Access Token:

```
sbo56dbkhr0l5tzreyjxf55e
```

#### Claims:

```
{
 "iss": "https://dex.example.com:32000",
 "sub": "CiqyGvONT11MC11MzBiLTRkNWytoT0K4Yy0zZGVkN2Y3Nj1kYjESBGxkyXA",
 "aud": "example-app",
 "exp": 1583532047,
 "iat": 1583445647,
 "at_hash": "D_ZuE9H6szxgffMNIQzwQ",
 "email": "alana@test.com",
 "email_verified": true,
 "name": "alana"
}
```

#### Refresh Token:

```
Ch1leGh4bW91dnRqeXJjcGFxbG4zenVuNHV5EhljeXizYjYyeWdwemp5NGJ0Ymdyb2JtNGR2
```

[Redeem refresh token](#)

6. Wait for the token to be generated.
7. Edit your `.kube/config` file to add a new context for the test user:

```
apiVersion: v1
clusters:
- cluster:
 certificate-authority-data: CA-CERT
 server: CLUSTER-URL
 name: TEST-CLUSTER
contexts:
- [EXISTING-CONTEXTS]
- context:
 cluster: TEST-CLUSTER
 user: TEST-USER
 name: TEST-CONTEXT
current-context: TEST-CONTEXT
kind: Config
preferences: {}
users:
- [EXISTING-USERS]
- name: TEST-USER
 user:
```

```
token: ID-TOKEN
```

Where:

- ◊ `CA` is your CA certificate.
- ◊ `CLUSTER-URL` is the address of the test service, such as `https://cluster.example.com:8443`.
- ◊ `TEST-CLUSTER` is the name of the test cluster, such as `cluster-with-custom-oidc`.
- ◊ `TEST-USER` is the test account username, such as `alana`.
- ◊ `TEST-CONTEXT` is a name you create for the new context, such as `cluster-with-custom-oidc-ldap-alana`.
- ◊ `ID-TOKEN` is the ID Token retrieved by the `example-app` app above.

Include the `cluster.server` and `user.token` values retrieved using the example app.

8. Create a `ClusterRole` YAML file that grants permissions to access services and pods in the `default` namespace:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
 namespace: default
 name: pod-reader-clusterRolebinding
rules:
- apiGroups: [""]
 resources: ["pods", "services"]
 verbs: ["get", "watch", "list"]
```

9. Run `kubectl apply` or `kubectl create` to pass the `ClusterRole` spec file to the kube controller:

```
kubectl apply -f ClusterRole.yml
```

10. Create a `ClusterRoleBinding` YAML file that applies the `ClusterRole` role to the test user.

For example:

```
apiVersion: rbac.authorization.k8s.io/v1
This role binding allows "alana@test.com" to read pods in the "default" namespace.
kind: ClusterRoleBinding
metadata:
 name: read-pods-clusterRolebinding
 namespace: default
subjects:
- kind: User
 name: alana@test.com # Name is case sensitive
 apiGroup: rbac.authorization.k8s.io
roleRef:
 kind: ClusterRole #this must be Role or ClusterRole
 name: pod-reader-clusterRolebinding # this must match the name of the Role or ClusterRole you wish to bind to
```

```
apiGroup: rbac.authorization.k8s.io
```

11. Run `kubectl apply` or `kubectl create` to pass the `ClusterRoleBinding` spec file to the kube controller:

```
kubectl apply -f ClusterRoleBinding.yml
```

12. Confirm the test user can run the following:

```
kubectl get pods
```

The cluster is successfully authenticating the user by connecting to the dex OIDC provider via OIDC if the test user can run `kubectl get pods`.

## Deleting Clusters

This topic describes how to delete a Kubernetes cluster deployed by VMware Tanzu Kubernetes Grid Integrated Edition. Running the `tkgi delete-cluster` command automatically deletes all cluster objects.

If you are using Tanzu Kubernetes Grid Integrated Edition with NSX-T, see [vSphere with NSX-T Cluster Objects](#) for a list of vSphere and NSX-T objects that will be deleted as part of the cluster deletion process.

## Delete Cluster

You can delete a cluster using the TKGI CLI.

To avoid an incomplete deletion, prepare the cluster for deletion before deleting it:

1. If the cluster is configured with a PodDisruptionBudget (PDB), remove the PDB from the cluster.
2. Remove all static and dynamic PVCs from the cluster.
3. Remove all PVs with a reclaimPolicy of `Retain` from the cluster.
4. If you are deleting a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.



**Note:** Before deleting a cluster, remove the PVs and PVCs from the cluster to avoid making orphan disks of the cluster's attached disks.

To delete a cluster:

1. On the command line, run the following command to log in:

```
tkgi login -a TKGI-API -u USERNAME -k
```

Where:

- `TKGI-API` is the domain name for the TKGI API that you entered in [Ops Manager >](#)

**Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN).**

For example, `api.tkgi.example.com`.

- ◊ `USERNAME` is your user name.

See [Logging in to Tanzu Kubernetes Grid Integrated Edition](#) for more information about the `tkgi login` command.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

2. Run `tkgi delete-cluster CLUSTER-NAME` to delete a cluster. Replace `CLUSTER-NAME` with the unique name for your cluster. For example:

```
$ tkgi delete-cluster my-cluster
```

3. Confirm cluster deletion by entering `y`, or cancel cluster deletion by entering `n`.

For example:

```
Are you sure you want to delete cluster my-cluster? (y/n)
```

## Verify Cluster Deletion

To verify cluster deletion using the TKGI CLI:

1. To verify cluster deletion, run `tkgi cluster CLUSTER-NAME`. Replace `CLUSTER-NAME` with the unique name for your cluster.

For example:

```
$ tkgi cluster my-cluster

Name: my-cluster
Plan Name: small
UUID: 106aabc7-5ecb-4c54-a800-a32eef57a593
Last Action: DELETE
Last Action State: in progress
Last Action Description: Instance deletion in progress
Kubernetes Master Host: my-cluster.tkgi.local
```

```
Kubernetes Master Port: 8443
Worker Nodes: 3
Kubernetes Master IP(s): 10.196.219.88
Network Profile Name:
```

While Tanzu Kubernetes Grid Integrated Edition is deleting the cluster, the value for [Last Action Description](#) is [Instance deletion in progress](#).

2. Continue running the `tkgi cluster CLUSTER-NAME` command to track cluster deletion. The cluster is deleted when the CLI returns [Error: Cluster CLUSTER-NAME not found](#).
3. Run `tkgi clusters`. The cluster you deleted should not appear in the list of Tanzu Kubernetes Grid Integrated Edition clusters.



**Note:** If the cluster is not deleted, see [Cluster Deletion Fails in Troubleshooting](#).

## Delete Cluster without Prompt

If you do not want the TKGI CLI to prompt you to confirm cluster deletion, use the `--non-interactive` flag.

For example:

```
$ tkgi delete-cluster my-cluster --non-interactive
```



**Note:** If you use the `--non-interactive` flag to delete multiple clusters, delete each cluster one by one. Do not create a script that deletes multiple clusters using the `--non-interactive` flag. If you do, the BOSH Director might hang and become unusable until you log in to BOSH and cancel each deletion task.

## Supporting Clusters

This section describes how to support Kubernetes clusters provisioned by VMware Tanzu Kubernetes Grid Integrated Edition.

See the following topics:

- [Retrieving Cluster Credentials and Configuration](#)
- [Managing Cluster Access and Permissions](#)
- [Using Admission Control Plugins for Tanzu Kubernetes Grid Integrated Edition Clusters](#)
- [Getting Started with VMware Harbor Registry](#)
- [Configuring Tanzu Kubernetes Grid Integrated Edition Clusters with Private Docker Registry](#)

### CA Certificates (Beta)

- PersistentVolume Storage Options on vSphere
- Configuring and Using PersistentVolumes



**Note:** Tanzu Kubernetes Grid Integrated Edition does not currently support the Kubernetes Service Catalog and the GCP Service Broker.

## Load Balancing and Ingress

This section describes how to create and configure load balancers for VMware Tanzu Kubernetes Grid Integrated Edition clusters. See the following topics:

- Creating and Configuring a GCP Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters
- Creating and Configuring an AWS Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters
- Creating and Configuring an Azure Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters
- Configuring Ingress Routing

## Load Balancing and Ingress with VMware NSX

This section provides topics for configuring the NSX-T load balancer used for ingress resources.

Layer 7 load balancing is implemented via a Kubernetes ingress resource. The ingress is allocated an IP from the Floating IP Pool specified in the NSX-T configuration. NCP exposes the ingress load balancer service on this IP address for both the HTTP and HTTPS ports (port 80 and 443).

## Configuring Ingress Using the NSX-T Load Balancer

- Monitoring Ingress Resources
- Viewing and Troubleshooting the Health Status of Cluster Network Objects
- Configuring Ingress Resources and Load Balancer Services
- Defining a Network Profile for Load Balancer Sizing
- Scaling the HTTP/S Layer 7 Ingress Load Balancers Using the LoadBalancer CRD
- Defining Network Profiles for the HTTP/S Layer 7 Ingress Controller
- Defining Network Profiles for the TCP Layer 4 Load Balancer
- Using Ingress URL Rewrite

## Monitoring Ingress Resources

This topic describes how to monitor the health status of the NSX-T ingress load balancer resources.



**Note:** This feature requires NCP v2.5.1 or later.

## Overview

The NSX-T Load Balancer is a logical load balancer that handles a number of functions using virtual servers and pools.

The NSX-T load balancer creates a load balancer service for each Kubernetes cluster provisioned by Tanzu Kubernetes Grid Integrated Edition with NSX-T. For each load balancer service, NCP, by way of the CRD, creates corresponding NSXLoadBalancerMonitor objects.

By default Tanzu Kubernetes Grid Integrated Edition deploys the following NSX-T virtual servers for each Kubernetes cluster:

- One TCP layer 4 load balancer virtual server for the Kubernetes API server.
- One TCP layer 4 auto-scaled load balancer virtual server for **each** Kubernetes service resource of `type: LoadBalancer`.
- Two HTTP/HTTPS layer 7 ingress routing virtual servers. These virtual servers are attached to the Kubernetes Ingress Controller cluster load balancer service and can be manually scaled. Tanzu Kubernetes Grid Integrated Edition uses Kubernetes custom resources to monitor the state of the NSX-T load balancer service and scale the virtual servers created for ingress.

For information about scaling TCP layer 4 ingress controller see [Defining Network Profiles for the TCP Layer 4 Load Balancer](#).

For information about configuring layer 7 ingress routing load balancers see [Scaling the HTTP/S Layer 7 Ingress Load Balancers Using the LoadBalancer CRD](#). For information about configuring the layer 7 ingress controller see [Defining Network Profiles for the HTTP/S Layer 7 Ingress Controller](#).

For more information about the NSX-T Load Balancer, see [Create an IP Pool in Manager Mode](#) or [Add an IP Address Pool](#) in the VMware documentation.

For more information about Kubernetes custom resources, see [Custom resources](#) in the Kubernetes documentation.

## Monitor the NSX-T Load Balancer Service

You can use the NSXLoadBalancerMonitor CRD to monitor the NSX-T load balancer service, including traffic, usage and health score information.

The NSXLoadBalancerMonitor returns statistics showing the number of connections and throughput of the virtual servers for each type of load balancer.

In addition to connections and throughput statistics the NSXLoadBalancerMonitor CRD returns two health scores for the current performance of load balancers:

- `servicePressureIndex` which represents an overall health score for the NSX-T load balancer service.
- `infraPressureIndex` which represents the health score of the NSX-T Edge Node that is running the load balancer and associated virtual servers.

Based on the health score the user can decide what action to take:

- If the health score is poor for one of the layer 4 load balancers, you can use a network profile to increase the size of the NSX-T load balancer service. For more information see [Defining Network Profiles for the TCP Layer 4 Load Balancer](#).
- If the health score is poor for the layer 7 ingress load balancers, you can use the [Scaling the HTTP/S Layer 7 Ingress Load Balancers Using the LoadBalancer CRD](#) to manually scale ingress.

The table below summarizes the actions that you can take based on the health scores.

| servicePressureIndex | infraPressureIndex | Cluster Manager                                                                                                                                                                                                                                                                          | Infrastructure Admin                                           |
|----------------------|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------|
| LOW or WARM          | LOW or WARM        | NONE                                                                                                                                                                                                                                                                                     | NONE                                                           |
| LOW or WARM          | HIGH               | Alert infra admin                                                                                                                                                                                                                                                                        | Move the LBS from the CRITICAL Edge Node to another Edge Node. |
| HIGH                 | LOW or WARM        | Resolve the LBS health score by <a href="#">Scaling the HTTP/S Layer 7 Ingress Load Balancers Using the LoadBalancer CRD</a> and, if necessary, by increasing the size of the LBS using <a href="#">Defining Network Profiles for the TCP Layer 4 Load Balancer</a> .                    | NONE                                                           |
| HIGH                 | HIGH               | Alert infra admin; Resolve the LBS health score by <a href="#">Scaling the HTTP/S Layer 7 Ingress Load Balancers Using the LoadBalancer CRD</a> and, if necessary, by increasing the size of the LBS using <a href="#">Defining Network Profiles for the TCP Layer 4 Load Balancer</a> . | Move the LBS from the CRITICAL Edge Node to another Edge Node. |

## Monitor Your NSX-T Load Balancer Service Using the NSXLoadBalancerMonitor CRD

To monitor your NSX-T Load Balancer Service using the NSXLoadBalancerMonitor CRD, complete the following procedure.

1. To view the NSXLoadBalancerMonitor CRD, run the following command:

```
kubectl get crd
```

2. To determine the UUID of the NSX-T load balancer deployed for the cluster, run the following command:

```
kubectl get nsxlbmonitors
```

3. To view statistics, throughput, and health score for all virtual servers deployed by a specific load balancer service, run the following command:

```
kubectl describe nsxlbmonitors UUID-OF-LOAD-BALANCER
```

Where `UUID-OF-LOAD-BALANCER` is your load balancer's UUID.

For example:

```
$ kubectl describe nsxlbmonitor f61a8cec-28eb-4b0c-bf4a-906f3ce2d8e6
Name: f61a8cec-28eb-4b0c-bf4a-906f3ce2d8e6 Namespace: Labels:
 <none> Annotations: <none> API Version: vmware.com/v1alpha1 He
alrh: Metrics: Cpu Usage Percentage: 0 Poolmember Usage Percen
tage: 1 Service Pressure Index: 0,LOW Infra Pressure Index:
 0,LOW Metrics: Cpu Usage Percentage: 0 Lb Service Usa
ge Percentage: 0 Memory Usage Percentage: 0 Poolmember Usage Per
centage: 0 Kind: NSXLoadBalancerMonitor M
etadada: Creation Timestamp: 2019-11-13T19:37:10Z Generation:
 914 Resource Version: 17139 Self Link: /apis/vmware.com
/v1alpha1/nsxlbmonitors/f61a8cec-28eb-4b0c-bf4a-906f3ce2d8e6 UID:
 f56d3cf5-748d-44c3-8026-c6c569fde954 Traffic: Bytes In Rat
e: 0 Bytes Out Rate: 0 Current Session Rate: 0 Ip Addr
ess: 192.168.160.102 Max Sessions: 0 Packets In Ra
te: 0 Packets Out Rate: 0 Protocol: TCP Total
Sessions: 0 Virtual Server Name: pks-042bccde-2197-4e06-863e-
55129bf2e195-http Bytes In Rate: 0 Bytes Out Rate: 0 Cu
rrent Session Rate: 0 Ip Address: 192.168.160.102 Max Sess
ions: 0 Packets In Rate: 0 Packets Out Rate: 0 Pro
tocol: TCP Total Sessions: 0 Virtual Server Name:
 pks-042bccde-2197-4e06-863e-55129bf2e195-https_terminated Usage: Cur
rent Server Pool Count: 1 Current Virtual Server Count: 3 Events:
 <none>
```

## Viewing and Troubleshooting the Health Status of Cluster Network Objects

This topic describes how cluster managers and users can troubleshoot NSX-T networking errors using the `kubectl nsxerrors` command.

### About the NSX Errors CRD

The NSX Errors CRD gives you the ability to view errors related to NSX-T that might occur when applications are deployed to a TKGI-provisioned Kubernetes cluster. Previously, NSX-T errors were logged in NCP logs on the control plane nodes, which cluster users do not have access to. The NSX Errors CRD improves visibility and troubleshooting for cluster managers and users.

The NSX Errors CRD creates a `nsxerror` object for each Kubernetes resource that encounters an NSX error during attempted creation. In addition, the Kubernetes resource is annotated with the `nsxerror` object name. The NSX Error CRD provides the command `kubectl nsxerrors` that lets you view the NSX errors encountered during resource creation. The `nsxerror` object is deleted once the NSX error is resolved and the Kubernetes resource is successfully created.

### Errors Reported by the NSX Errors CRD

The following errors are reported by the NSX Errors CRD:

- Auto-scaler failed to allocate additional load balancer service due to Edge Node limit

- Number of pools exceed the load balancer service limit
- Number of pool members exceed the load balancer service and Edge Node limit
- Floating IP pool is exhausted when exposing the load balancer type service
- Pod IP Block is exhausted
- The number of available IP allocations is low
- The NSX manager is unavailable
- The NSX manager rate limit is exceeded

## NSX Errors CRD Example

To illustrate how the NSX Errors CRD works and can be used, consider the following example: the NSX auto-scaler fails to allocate additional load balancer services due to Edge Node limits reached. In this case, the number of virtual switches exceed load balancer service limits with auto-scaling enabled.

The resource is fetched by name to check its status.

```
kubectl get svc test-svc-3
test-svc-3 LoadBalancer 10.104.236.243 <pending> 80:32095/TCP,8080:32664/TCP 4
```

The status is pending so we look at the annotations. The `ncp/error` and `nsxerror` annotations are visible.

```
kubectl get svc test-svc-3 -o yaml
annotations:
 ncp/error.loadbalancer: SERVICE_LOADBALANCER_UNREALIZED
 Nsxerror: services-1f48fa28c17d983bc73c33f005611e0c
```

We use the command `kubectl get nsxerror` to view the details of the error, revealing that the number of load balancer virtual server instances requested exceeds the limits of the Edge Node.

```
kubectl get nsxerror services-1f48fa28c17d983bc73c33f005611e0c
- apiVersion: vmware.eng.com/v1
 kind: NSXError
 metadata:
 clusterName: ""
 creationTimestamp: 2019-01-22T03:17:16Z
 labels:
 error-object-type: services
 name: services-1f48fa28c17d983bc73c33f005611e0c
 namespace: ""
 resourceVersion: "1291084"
 selfLink: /apis/vmware.eng.com/v1/services-1f48fa28c17d983bc73c33f005611e0c
 uid: 386e60e5-1df4-11e9-abd8-000c29c02b4c
 spec:
 error-object-id: default.test-svc-1
 error-object-name: test-svc-1
 error-object-ns: default
 error-object-type: services
 message: [2019-01-21 19:17:16]10087: Number of loadbalancer requested exceed Edge node limit'
```

# Configuring Ingress Resources and Load Balancer Services

This topic describes example configurations for ingress routing (Layer 7) and load balancing (Layer 4) for Kubernetes clusters deployed by VMware Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T integration.



**Note:** The examples in this topic are based on NCP v2.3.2.

## Kubernetes Ingress Rules

A Kubernetes ingress resource exposes HTTP and HTTPS routes from outside the cluster to services within the cluster. Traffic routing is controlled by rules defined on the ingress resource.

You define ingress resource configuration in the manifest of your Kubernetes deployment. When you define an ingress rule, the hostname and path values are both optional. It is common to define an ingress rule that specifies a hostname and no path, but defining an ingress rule without a hostname is uncommon. You can use wildcard DNS entries to route traffic to the exposed ingress resource.

When you define two ingress rules with the same hostname, include both the hostname and path in the ingress rules to avoid ambiguity.

Rules:

- If multiple ingress rules use the same hostname and the same path, the first rule you create takes priority.
- If an ingress rule that includes only a hostname precedes a rule that includes both the same hostname and a path, the first rule takes priority.

For example:

- The following NSX ingress rule includes both a host and a path specification. The rule matches `host: test.com` and `path: /testpath` in the incoming request:

### Ingress Rule Example 1

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
 name: svc-ingress1
 annotations:
 kubernetes.io/ingress.class: "nsx"
spec:
 rules:
 - host: test.com
 http:
 paths:
 - path: /testpath
 backend:
 serviceName: svc1
 servicePort: 80
```

- The following NSX ingress rule includes only a host specification. The rule matches all `host:`

`test.com` in the incoming request:

### Ingress Rule Example 2

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
 name: svc-ingress2
 annotations:
 kubernetes.io/ingress.class: "nsx"
spec:
 rules:
 - host: test.com
 http:
 paths:
 - path:
 backend:
 serviceName: svc1
 servicePort: 80
```

- If you create **Ingress Rule Example 1** before **Ingress Rule Example 2**, then `svc-ingress1` serves the `test.com/testpath` URI because inbound requests hit the `host: test.com` and `path: /testpath` NSX ingress rule first.
- If you create **Ingress Rule Example 2** before **Ingress Rule Example 1**, then `svc2-ingress2` serves the `test.com/testpath` URI because inbound requests hit the `host: test.com` NSX ingress rule first.

For more information about Kubernetes ingress resources, see [Ingress](#) in the Kubernetes documentation.

## The NSX-T Load Balancer Service

NSX-T supports autoscaling, which spins up a new Kubernetes `type: LoadBalancer` service if the previous one has reached its scale limit. The NSX-T load balancer that is automatically provisioned by Tanzu Kubernetes Grid Integrated Edition provides two Layer 7 virtual servers for Kubernetes ingress resources, one for HTTP and the other for HTTPS.

For more information, see [Supported Load Balancer Features](#) in the NSX-T documentation.

The following is the format for the Kubernetes `LoadBalancer` service definition:

```
kind: Service
apiVersion: v1
metadata:
 name: SERVICE-NAME
spec:
 type: LoadBalancer
 selector:
 app: APP-NAME
 ports:
 - protocol: PROTOCOL
 port: PORT
 targetPort: TARGET-PORT
 name: PORT-NAME
```

Where:

- **SERVICE-NAME** is the name for your load balancer service.
- **APP-NAME** is the name of your app serviced by the load balancer service.
- **PROTOCOL** (Optional) is the network protocol to service. If the protocol is not specified it defaults to **TCP**. For more information about supported protocols, see [Supported protocols](#) in the Kubernetes documentation.
- **PORT** is the listening port. An integer value is supported. For example, **80**.
- **TARGET-PORT** is the target port. Either an integer or a string value is supported. For example, **8080** or **http**.
- **PORT-NAME** (Optional) is the port name. Kubernetes requires the port name be specified for multi-port services.

For example, the following is a **LoadBalancer** service definition for an Tanzu Kubernetes Grid Integrated Edition-provisioned cluster with NSX-T:

```
kind: Service
apiVersion: v1
metadata:
 name: test-service
spec:
 type: LoadBalancer
 selector:
 app: testApp
 ports:
 - protocol: TCP
 port: 80
 targetPort: 8080
 name: web
```



**Note:** With NCP v2.3.2 and earlier, the named **targetPort** must be an integer, not a string. If you define a service **type: LoadBalancer** with NSX-T, the value of **targetPort** must be a port number, not a port name.

For more information about the Kubernetes **LoadBalancer** service definition see [Type LoadBalancer](#) in the Kubernetes documentation.

When deploying a Kubernetes **LoadBalancer** service, NSX-T automatically creates a new virtual IP address (VIP) on the existing load balancer.

## Size a Load Balancer

This topic describes how to size a load balancer using a network profile.

## Load Balancer Sizing

When you deploy a Kubernetes cluster using Tanzu Kubernetes Grid Integrated Edition on NSX-T, an NSX-T Load Balancer is automatically provisioned. By default the size of this load balancer is

small. Using a network profile, you can customize the size of this load balancer and use a medium or large load balancer for Kubernetes clusters.

The NSX-T Load Balancer is a logical load balancer that handles a number of functions using virtual servers and pools. For more information, see [Supported Load Balancer Features](#) in the NSX-T documentation.

The following virtual servers are required for Tanzu Kubernetes Grid Integrated Edition:

- 1 global virtual server for the Kubernetes API which runs on the control plane nodes
- 1 TCP layer 4 virtual server for **each** Kubernetes service of `type:LoadBalancer`
- 2 HTTP and HTTPS layer 7 global virtual servers for Kubernetes ingress controller resources

The number of virtual servers that you can run depends on the size of the load balancer, which in turn depends on the size of the NSX-T Edge Node hosting the load balancer service. See [Scaling Load Balancer Resources](#) in the NSX-T documentation. Because of the number of virtual servers required by Tanzu Kubernetes Grid Integrated Edition, you can only use the large NSX Edge Node VM or the bare metal NSX Edge Node with Tanzu Kubernetes Grid Integrated Edition.

You cannot modify the Load Balancer Size configuration on an existing cluster.

The following network profile, `np-lb-med`, defines a medium load balancer:

```
{
 "name": "np-lb-med",
 "description": "Network profile for medium NSX-T load balancer",
 "parameters": {
 "lb_size": "medium"
 }
}
```

The following network profile, `np-lb-large`, defines a large load balancer:

```
{
 "name": "np-lb-large",
 "description": "Network profile for large NSX-T load balancer",
 "parameters": {
 "lb_size": "large"
 }
}
```

## Scaling the HTTP/S Layer 7 Ingress Load Balancers Using the LoadBalancer CRD

This topic describes how to scale ingress resources.



**Note:** This feature requires NCP v2.5.1 or later.

## Overview

The NSX-T Load Balancer is a logical load balancer that handles a number of functions using virtual servers and pools.

The NSX-T load balancer creates a load balancer service for each Kubernetes cluster provisioned by Tanzu Kubernetes Grid Integrated Edition with NSX-T. For each load balancer service, NCP, by way of the CRD, creates corresponding NSXLoadBalancerMonitor objects.

By default Tanzu Kubernetes Grid Integrated Edition deploys the following NSX-T virtual servers for each Kubernetes cluster:

- One TCP layer 4 load balancer virtual server for the Kubernetes API server.
- One TCP layer 4 auto-scaled load balancer virtual server for **each** Kubernetes service resource of `type: LoadBalancer`.
- Two HTTP/HTTPS layer 7 ingress routing virtual servers. These virtual servers are attached to the Kubernetes Ingress Controller cluster load balancer service and can be manually scaled. Tanzu Kubernetes Grid Integrated Edition uses Kubernetes custom resources to monitor the state of the NSX-T load balancer service and scale the virtual servers created for ingress.

For information about configuring layer 7 ingress routing load balancers see [Determine Your Load Balancer's Status](#), below. For information about configuring the layer 7 ingress controller see [Defining Network Profiles for the HTTP/S Layer 7 Ingress Controller](#).

For information about configuring TCP layer 4 ingress controller see [Defining Network Profiles for the TCP Layer 4 Load Balancer](#).

For more information about the NSX-T Load Balancer, see [Create an IP Pool in Manager Mode](#) or [Add an IP Address Pool](#) in the VMware documentation.

For more information about Kubernetes custom resources, see [Custom resources](#) in the Kubernetes documentation.

## Prerequisites

Before scaling your ingress load balancers you should understand your load balancer's status. Use the NSXLoadBalancerMonitor CRD to monitor your NSX-T load balancer service, including traffic, usage and health score information. The NSXLoadBalancerMonitor CRD provides information for the health of the NSX-T load balancer service, and the NSX-T Edge Node running the load balancer.

For more information about monitoring using the NSXLoadBalancerMonitor CRD see [Monitoring Ingress Resources](#).

## Scale Ingress Load Balancer Resources

The LoadBalancer CRD provides you with an interactive method to scale the load balancer for ingress routing.

### Create a New Ingress Load Balancer

Use the LoadBalancer CRD to create a new ingress load balancer.

1. To configure a new ingress load balancer, configure a new YAML file as follows:

```
apiVersion: vmware.com/v1alpha1
kind: LoadBalancer
metadata:
```

```

name: LB-NAME
spec:
 httpConfig: HTTP-CONFIG
 virtualIP: IP-ADDRESS
 port: PORT
 tls:
 port: TLS-PORT
 secretName: SECRET-NAME
 secretNamespace: SECRET-NAMESPACE
 xForwardedFor: FORWARD-TYPE
 affinity:
 type: IP-SOURCE
 timeout: TIMEOUT
 size: SIZE
 virtualNetwork: NETWORK-NAME
status:
 httpVirtualIP: V-IP-ADDRESS

```

Where:

- ◊ `LB-NAME` is the display name of the loadBalancer.
- ◊ `HTTP-CONFIG` (Optional) is the config to support http/https route on the loadBalancer. Set as `httpConfig: {}` to apply default settings.
- ◊ `IP-ADDRESS` (Optional) is the virtual IP address. Defaults to `auto_allocate`.
- ◊ `PORT` (Optional) is the port. Defaults to `80`.
- ◊ `TLS-PORT` (Optional) is the TLS port. Defaults to `443`.
- ◊ `SECRET-NAME` (Optional) is the TLS secret name. Defaults to `nil`.
- ◊ `SECRET-NAMESPACE` (Optional) is the TLS secret namespace. Defaults to `nil`. You must deploy the new ingress load balancer in the same namespace where you deploy the ingress resource.
- ◊ `FORWARD-TYPE` (Optional) is the forward type. Supported values are: `INSERT` and `REPLACE`. Defaults to `nil`.
- ◊ `IP-SOURCE` (Optional) is the source IP. Supported values are: `sourceIP` and `cookie`.
- ◊ `TIMEOUT` (Optional) is the connection timeout. Defaults to `10800`.
- ◊ `SIZE` (Optional) is the ingress load balancer size. Supported values are: `SMALL` and `MEDIUM`. Defaults to `SMALL`.
- ◊ `NETWORK-NAME` (Optional) is the virtual network name. Defaults to `nil`.
- ◊ `V-IP-ADDRESS` is the external IP address for http/https virtual server. The external IP address can be auto-allocated or user specified.

2. To create a new ingress load balancer run the following command:

```
kubectl apply -f YAML-FILE
```

Where `YAML-FILE` is the filename of a the load balancer configuration YAML file.

For example:

```
kubectl apply -f lb.yaml

apiVersion: vmware.com/v1alpha1

kind: LoadBalancer

metadata:

name: cluster1_lbs0

spec:

httpConfig:

virtualIP:

port: 233

tls:

port: 2333

secretName: default_secret

secretNamespace: default

xForwardedFor: INSERT

affinity:

type: source_ip

timeout: 100

size: MEDIUM

virtualNetwork: virtualnetwork1

status:

httpVirtualIP: <realized external ip>
```

## Configure Your Kubernetes Ingress Resource to Use the New Ingress Load Balancer

Annotate the Kubernetes ingress resource with the newly created ingress load balancer. NCP will attach the ingress rules to the scaled out load balancer.

- To configure a Kubernetes ingress resource with the new ingress load balancer, configure a new YAML file as follows:

```

apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
 name: ING-NAME
 annotations:
 kubernetes.io/ingress.class: "nsx"
 nsx/loadbalancer: LB-NAME
spec:
 rules:
 - host: HOST-NAME
 http:
 paths:
 - path: HTTP-PATH
 backend:
 serviceName: SERVICE-NAME
 servicePort: SERVICE-PORT

```

Where:

- ◊ `ING-NAME` is the name of the ingress resource.
- ◊ `LB-NAME` is the display name of the loadBalancer.
- ◊ `HOST-NAME` is the host name.
- ◊ `HTTP-PATH` is the HTTP path.
- ◊ `SERVICE-NAME` is the http backend service name.
- ◊ `SERVICE-PORT` is the http backend service port.

- To annotate the Kubernetes ingress resource with the newly created ingress load balancer, run the following command:

```
kubectl apply -f YAML-FILE
```

Where `YAML-FILE` is the filename of a the Kubernetes ingress resource configuration YAML file.

For example:

```

kubectl apply -f ingress.yaml

apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
 name: svc-ingress1

```

```

annotations:

kubernetes.io/ingress.class: "nsx" nsx/loadbalancer: cluster1_lbs0

spec:

rules:

- host: test.com

http:

paths:

- path: /testpath

backend:

serviceName: svc1

servicePort: 80

```

## Configure the HTTP/S Layer 7 Ingress Controller

This topic describes how to define network profiles for Kubernetes clusters provisioned with VMware Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T.

### Overview

The NSX-T Load Balancer is a logical load balancer that handles a number of functions using virtual servers and pools.

The NSX-T load balancer creates a load balancer service for each Kubernetes cluster provisioned by Tanzu Kubernetes Grid Integrated Edition with NSX-T. For each load balancer service, NCP, by way of the CRD, creates corresponding NSXLoadBalancerMonitor objects.

By default Tanzu Kubernetes Grid Integrated Edition deploys the following NSX-T virtual servers for each Kubernetes cluster:

- One TCP layer 4 load balancer virtual server for the Kubernetes API server.
- One TCP layer 4 auto-scaled load balancer virtual server for **each** Kubernetes service resource of `type: LoadBalancer`.
- Two HTTP/HTTPS layer 7 ingress routing virtual servers. These virtual servers are attached to the Kubernetes Ingress Controller cluster load balancer service and can be manually scaled. Tanzu Kubernetes Grid Integrated Edition uses Kubernetes custom resources to monitor the state of the NSX-T load balancer service and scale the virtual servers created for ingress.

For information about configuring the layer 7 ingress controller see [Defining Network Profiles for the HTTP/S Layer 7 Ingress Controller](#), below. For information about configuring layer 7 ingress routing load balancers see [Scaling the HTTP/S Layer 7 Ingress Load Balancers Using the LoadBalancer](#)

CRD.

For information about configuring TCP layer 4 ingress routing load balancers see [Defining Network Profiles for the TCP Layer 4 Load Balancer](#).

For more information about the NSX-T Load Balancer, see [Create an IP Pool in Manager Mode](#) or [Add an IP Address Pool](#) in the VMware documentation.

For more information about Kubernetes custom resources, see [Custom resources](#) in the Kubernetes documentation.

## Configure the HTTP/HTTPS Ingress Controller Network Profile

The HTTP/HTTPS layer 7 virtual servers provisioned for each Kubernetes service are controlled by the parameters exposed in a network profile.

### NSX-T HTTP/HTTPS Ingress Controller Network Profile Configuration

The NSX Ingress Controller is configured using the `ncp.ini` network profile configuration file.

The HTTP/HTTPS Ingress Controller network profile has the following format:

```
{
 "name": "ncp_network_profile",
 "description": "DESCRIP",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "nsx_lb": NSX-LB,
 "ingress_ip": "IP-ADDRESS",
 "ingress_persistence_settings": {
 "persistence_type": "PERS-TYPE",
 "persistence_timeout": TIMEOUT
 }
 }
 }
 }
}
```

Where:

- `DESCRIP` is your description for this network profile configuration.
- `NSX-LB` is your preference for whether the NSX-T Load Balancer is used for your Kubernetes clusters. For more information, see [Configure the NSX Ingress Controller](#) below.
- `IP-ADDRESS` is IP address to use for ingress controller load balancer. For more information, see [Configure the Ingress IP](#) below.
- `PERS-TYPE` is the persistence type to use for ingress controller load balancer. For more information, see [Configure the Ingress Persistence Settings](#) below.
- `TIMEOUT` is the persistence timeout to use for ingress controller load balancer. For more information, see [Configure the Ingress Persistence Settings](#) below.

For example:

```
{
 "name": "ncp_network_profile",
 "description": "Example network profile for ingress controller",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "nsx_lb": true,
 "ingress_ip": "192.168.160.212",
 "ingress_persistence_settings": {
 "persistence_type": "cookie",
 "persistence_timeout": 1
 }
 }
 }
 }
}
```

The following table describes the Ingress Controller configuration parameters:

| Parameter                                 | Type                         | Description                                                                                                                                                     |
|-------------------------------------------|------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>name</code>                         | String                       | User-defined name of the network profile.                                                                                                                       |
| <code>description</code>                  | String                       | User-defined description for the network profile.                                                                                                               |
| <code>parameters</code>                   | Map                          | Map containing one or more key-value pairs.                                                                                                                     |
| <code>cni_configurations</code>           | Map                          | Map containing <code>type</code> and <code>parameters</code> key-value pairs for configuring NCP.                                                               |
| <code>type</code>                         | Constant String              | Values: <code>"nsxt"</code> .                                                                                                                                   |
| <code>parameters</code>                   | Map                          | Map containing one or more key-value pairs for NCP settings.                                                                                                    |
| <code>nsx_lb</code>                       | Boolean<br><b>Updateable</b> | Use NSX-T layer 4 virtual server for each Kubernetes service of type LoadBalancer.<br>Values: <code>true, false</code> .<br>Default: <code>true</code> .        |
| <code>nsx_ingress_controller</code>       | Boolean                      | Use NSX-T layer 7 virtual server as the ingress controller for the Kubernetes cluster.<br>Values: <code>true, false</code> .<br>Default: <code>true</code> .    |
| <code>ingress_ip</code>                   | String                       | IP address to use for ingress controller load balancer.                                                                                                         |
| <code>ingress_persistence_settings</code> | String<br><b>Updateable</b>  | Map containing one or more key-value pairs for customizing Layer 7 persistence.<br>See also: <code>persistence_timeout</code> and <code>persistence_type</code> |
| <code>persistence_type</code>             | String<br><b>Updateable</b>  | An <code>ingress_persistence_settings</code> parameter. Specify the ingress persistence type.<br>Values: <code>"none", "cookie", "source_ip"</code> .           |

---

|                                  |                   |                                                                                                  |
|----------------------------------|-------------------|--------------------------------------------------------------------------------------------------|
| <code>persistence_timeout</code> | Integer           | An <code>ingress_persistence_settings</code> parameter. Persistence timeout interval in seconds. |
|                                  | <b>Updateable</b> | See also: <code>connect_retry_timeout</code> and <code>lb_http_response_timeout</code> .         |

---

The `nsx_lb` parameter is used to control the TCP layer 4 virtual server that is provisioned for each Kubernetes service of type: [LoadBalancer](#).

When you configure an NSX Load Balancer as your Kubernetes cluster [ingress resource](#), NCP instructs the NSX-T Load Balancer to provision two layer 7 virtual services (HTTP and HTTPS) as the cluster [Ingress Controller](#):

| <code>nsx_lb</code> setting | Description                                                                                           |
|-----------------------------|-------------------------------------------------------------------------------------------------------|
| <code>nsx_lb: true</code>   | Use an NSX-T Layer 4 LoadBalancer and NCP-provisioned Layer 7 Ingress Controller.                     |
| <code>nsx_lb: false</code>  | Use a third-party load balancer and a third-party ingress controller, such as <a href="#">NGINX</a> . |

---

## Configure the NSX Ingress Controller

NCP depends on the NSX-T Load Balancer to fulfill its role as an Ingress Controller. To use a third-party ingress controller, such as the [NGINX Ingress Controller](#), set `nsx_lb` to `false`.

For example:

- The following network profile uses the NSX-T Load Balancer and NSX Ingress Controller:

```
{
 "name": "example_network_profile",
 "description": "Use the NSX-T Load Balancer and NSX Ingress Controller",
 "parameters": {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "nsx_lb": true
 }
 }
 }
}
```

- The following network profile uses a third party load balancer and a third-party ingress controller:

```
{
 "name": "example_network_profile",
 "description": "Use a 3rd party load balancer and ingress controller",
 "parameters": {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "nsx_lb": false
 }
 }
 }
}
```

## Configure the Ingress IP

The `ingress_ip` parameter instructs NCP to create an ingress virtual server with the given IP address.

The `ingress_ip` parameter type is a string that accepts any valid IP address. Missing entry is accepted.

Example network profile for `ingress_ip`:

```
{
 "name": "example-network-profile",
 "description": "ingress_ip",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "ingress_ip": "192.168.160.212"
 }
 }
 }
}
```

An invalid IP address is rejected with an invalid parameter value error.

For example:

- The following network profile parameters cannot be parsed because the `"ingress_ip"` configuration specifies an invalid IP address:

```
{
 "name": "example-network-profile",
 "description": "ingress_ip-ERROR",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "ingress_ip": "192.168.460.212"
 }
 }
 }
}
```

- The following network profile cannot be parsed because the `"ingress_ip"` configuration is not a string and the JSON input is invalid:

```
{
 "name": "example-network-profile",
 "description": "ingress_ip-ERROR",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "ingress_ip": 192.168.160.212
 }
 }
 }
}
```

```
}
```

## Configure the Ingress Persistence Settings

The `ingress_persistence` parameter lets you customize layer 7 persistence for Kubernetes services.

The `ingress_persistence_settings` parameter is a map that supports two keys:

- `persistence_type`
- `persistence_timeout`

These two keys are correlated and must be set/unset at the same time. If `persistence_type` and `persistence_timeout` are not both specified, the network profile fails validation.

| Parameter                        | Data Type | Description                                                                                      |
|----------------------------------|-----------|--------------------------------------------------------------------------------------------------|
| <code>persistence_type</code>    | String    | Valid values are <code>cookie</code> or <code>source_ip</code> . An empty value is not accepted. |
| <code>persistence_timeout</code> | Integer   | Value that is equal to <code>1</code> or larger. Empty value is not accepted.                    |

For example:

- Network profile for `ingress_persistence_settings`:

```
{
 "name": "example_network_profile",
 "description": "ingress_persistence_settings",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "ingress_ip": "192.168.160.212"
 "ingress_persistence_settings": {
 "persistence_type": "cookie",
 "persistence_timeout": 1
 }
 }
 }
 }
}
```

- Network profile for `ingress_persistence_settings`:

```
{
 "name": "example_network_profile",
 "description": "ingress_persistence_settings",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "ingress_ip": "192.168.160.212"
 "ingress_persistence_settings": {
 "persistence_type": "source_ip",
 "persistence_timeout": 100
 }
 }
 }
 }
}
```

}

# Configure the TCP Layer 4 Load Balancer

This topic describes how to define network profile to configure the NSX-T Load Balancer for VMware Tanzu Kubernetes Grid Integrated Edition.

# Overview

The NSX-T Load Balancer is a logical load balancer that handles a number of functions using virtual servers and pools.

The NSX-T load balancer creates a load balancer service for each Kubernetes cluster provisioned by Tanzu Kubernetes Grid Integrated Edition with NSX-T. For each load balancer service, NCP, by way of the CRD, creates corresponding NSXLoadBalancerMonitor objects.

By default Tanzu Kubernetes Grid Integrated Edition deploys the following NSX-T virtual servers for each Kubernetes cluster:

- One TCP layer 4 load balancer virtual server for the Kubernetes API server.
  - One TCP layer 4 auto-scaled load balancer virtual server for **each** Kubernetes service resource of `type: LoadBalancer`.
  - Two HTTP/HTTPS layer 7 ingress routing virtual servers. These virtual servers are attached to the Kubernetes Ingress Controller cluster load balancer service and can be manually scaled. Tanzu Kubernetes Grid Integrated Edition uses Kubernetes custom resources to monitor the state of the NSX-T load balancer service and scale the virtual servers created for ingress.

For information about configuring TCP layer 4 ingress controller see [Configure the TCP Ingress Controller Network Profile](#), below.

For information about configuring layer 7 ingress routing load balancers see [Scaling the HTTP/S Layer 7 Ingress Load Balancers Using the LoadBalancer CRD](#). For information about configuring the layer 7 ingress controller see [Defining Network Profiles for the HTTP/S Layer 7 Ingress Controller](#).

For more information about the NSX-T Load Balancer, see [Create an IP Pool in Manager Mode](#) or [Add an IP Address Pool](#) in the VMware documentation.

For more information about Kubernetes custom resources, see [Custom resources](#) in the Kubernetes documentation.

## Configure the TCP Ingress Controller Network Profile

The TCP layer 4 virtual server provisioned for each Kubernetes service is controlled by the parameters exposed in a network profile.



**Note:** The TCP layer 4 virtual server that fronts the Kubernetes API server is always created, and it is not controlled by the parameters exposed in the network profile.

## NSX-T TCP Ingress Controller Network Profile Configuration

The NSX Ingress Controller is configured using the `ncp.ini` network profile configuration file.

The TCP Ingress Controller network profile has the following format:

```
{
 "name": "network_profile",
 "description": "DESCRIP",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "nsx_lb": NSX-LB,
 "x_forwarded_for": "FORWARD-TYPE",
 "max_14_lb_service": MAX-SERVERS,
 "l4_persistence_type": "source_ip",
 "l4_lb_algorithm": "LB-BEHAVIOR"
 }
 }
 }
}
```

Where:

- `DESCRIP` is your description for this network profile configuration.
- `NSX-LB` is your preference for whether the NSX-T Load Balancer is used for your Kubernetes clusters. For more information see [Configure Which NSX Load Balancer to Use](#), below.
- `FORWARD-TYPE` (Optional) is your request header original client source IP. For more information see the [Configure the Client Source IP Address](#), below.
- `MAX-SERVERS` is your maximum number of layer 4 virtual servers per cluster. For more information see the [Configure the Maximum Number of Layer 4 Load Balancer Virtual Servers](#), below.
- `LB-BEHAVIOR` (Optional) is your load balancer behavior. For more information see the [Configure the Layer 4 Load Balancer Algorithm](#), below.

For example:

```
{
 "name": "network_profile",
 "description": "DESCRIP",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "nsx_lb": true,
 "x_forwarded_for": "replace",
 "max_14_lb_service": 10,
 "l4_persistence_type": "source_ip",
 "l4_lb_algorithm": "weighted_round_robin"
 }
 }
 }
}
```

The following table describes the Ingress Controller configuration parameters:

| Parameter                       | Type     | Description                                                                                                                                                                                                                                                                                          |
|---------------------------------|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>name</code>               | String   | User-defined name of the network profile.                                                                                                                                                                                                                                                            |
| <code>description</code>        | String   | User-defined description for the network profile.                                                                                                                                                                                                                                                    |
| <code>parameters</code>         | Map      | Map containing one or more key-value pairs.                                                                                                                                                                                                                                                          |
| <code>cni_configurations</code> | Map      | Map containing <code>type</code> and <code>parameters</code> key-value pairs for configuring NCP.                                                                                                                                                                                                    |
| <code>type</code>               | Constant | Values: "nsxt".                                                                                                                                                                                                                                                                                      |
| <code>parameters</code>         | Map      | Map containing one or more key-value pairs for NCP settings.                                                                                                                                                                                                                                         |
| <code>nsx_lb</code>             | Boolean  | Use NSX-T layer 4 virtual server for each Kubernetes service of type LoadBalancer.<br>Values: <code>true</code> , <code>false</code> .<br>Default: <code>true</code> .                                                                                                                               |
| <code>x_forwarded_for</code>    | String   | Sets the original client source IP in the request header. Enabling the network profile <code>x_forwarded_for</code> parameter automatically enables the <code>x_forwarded_port</code> and <code>x_forward_protocol</code> parameters.<br>Values: "none", "insert" and "replace".<br>Default: "none". |
| <code>max_l4_lb_service</code>  | Integer  | Limit the maximum number of layer 4 virtual servers per cluster.<br>Minimum Value:1.<br>See also: <code>l4_lb_algorithm</code> and <code>l4_persistence_type</code> .                                                                                                                                |

---

|                                  |                                                                                                                                                                                                                                                                   |
|----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>14_persistence_type</code> | St Connection stickiness based on <code>source_ip</code> .<br>rin Values: "source_ip".<br>g See also: <code>14_lb_algorithm</code> and <code>max_14_lb_service</code> .<br>U<br>pd<br>at<br>ab<br>le                                                              |
| <code>14_lb_algorithm</code>     | St Layer 4 load balancer behavior.<br>rin Values: "round_robin", "least_connection", "ip_hash", "weighted_round_robin".<br>g Default: "round_robin".<br>U<br>pd See also: <code>14_persistence_type</code> and <code>max_14_lb_service</code> .<br>at<br>ab<br>le |

---

The `nsx_lb` parameter is used to control the TCP layer 4 virtual server that is provisioned for each Kubernetes service of type: [LoadBalancer](#).

When you configure an NSX Load Balancer as your Kubernetes cluster [ingress resource](#), NCP instructs the NSX-T Load Balancer to provision two layer 7 virtual services (HTTP and HTTPS) as the cluster [Ingress Controller](#):

| <code>nsx_lb</code> setting | Description                                                                                           |
|-----------------------------|-------------------------------------------------------------------------------------------------------|
| <code>nsx_lb: true</code>   | Use an NSX-T Layer 4 LoadBalancer and NCP-provisioned Layer 7 Ingress Controller.                     |
| <code>nsx_lb: false</code>  | Use a third-party load balancer and a third-party ingress controller, such as <a href="#">NGINX</a> . |

---

## Configure Which NSX Load Balancer to Use

The `nsx_lb` flag controls whether to deploy either the NSX-T Load Balancer or a third-party load balancer, such as Nginx.

The `nsx_lb` parameter accepts `true` or `false`. The default setting is `true` and the NSX-T Load Balancer is deployed. To use a third party load balancer, set this parameter to `false`.

For example:

```
{
 "name": "example_network_profile",
 "description": "nsx_lb is enabled",
 "parameters": {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "nsx_lb": false
 }
 }
 }
}
```



**Note:** The `nsx_lb` parameter maps to the `use_native_loadbalancer` parameter in NCP.ini.

For more information about configuring the NSX Ingress Controller component of the NSX-T Load Balancer, see [Defining Network Profiles for the HTTP/S Layer 7 Ingress Controller](#).

## Configure the Client Source IP Address

The `X-Forwarded-For` HTTP header field is used to identify the originating IP address of a client connecting to a web server through an HTTP proxy or load balancer.

In network profile, the `x_forwarded_for` parameter can be enabled to ensure that the client IP address will be set in the HTTP request header. The `x_forwarded_for` parameter is useful in situations where it is important to know the source IP address of the client request, such as for auditing purposes.



**Note:** Enabling the network profile `x_forwarded_for` parameter automatically enables the `x_forwarded_port` and `x_forwarded_protocol` parameters. This is supported with NSX-T v3.0.x and later.

The `x_forwarded_for` parameter type is String that accepts `"insert"` and `"replace"`. Any other type will be rejected. Missing entry is accepted.

If `x_forwarded_for` is set to `"insert"`, the client source IP will be appended (comma separated) to the existing set of client source IP addresses. If `x_forwarded_for` is set to `"replace"`, any existing client source IP address will be replaced with the current client source IP address.

For example, with the following network profile the source IP address of the client will be appended to the existing set of client source IP addresses:

```
{
 "name": "example-network-profile",
 "description": "x_forwarded_for insert",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "x_forwarded_for": "insert"
 }
 }
 }
}
```

For example, with the following network profile the existing source IP address of the client will replace all other client source IP entries:

```
{
 "name": "example-network-profile",
 "description": "x_forwarded_for replace",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "x_forwarded_for": "replace"
 }
 }
 }
}
```

```

 }
}
}
```

## Configure the Maximum Number of Layer 4 Load Balancer Virtual Servers

The default NSX-T Load Balancer behavior is that auto-scaling is unlimited. This means that the number of layer 4 virtual servers that can be deployed is governed only by the capacity of the Edge Cluster where the load balancer service is deployed. As a result, in theory it is possible for a single Kubernetes cluster to use up all of the layer 4 virtual servers that the Edge Cluster can support.

The `max_14_service` parameter sets the upper limit for the number of virtual servers that can be used by a Kubernetes cluster. You can use this parameter to limit the number of virtual servers that can be created per Kubernetes cluster.

The `max_14_lb_service` data type is an integer. The value must be larger or equal to 1. Missing entry is accepted.

For example, the following network profile uses the `max_14_lb_service` parameter to limit the number of layer 4 virtual servers to 100 per cluster:

```
{
 "name": "example_network_profile",
 "description": "max_14_lb_service",
 "parameters" : {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "max_14_lb_service": 100
 }
 }
 }
}
```

## Configure the Layer 4 Persistence Type

The `l4_persistence_type` is used to set connection stickiness based on `source_ip`.

The `l4_persistence_type` data type is string. The only accepted value is `source_ip`.

```
{
 "name": "example_network_profile",
 "description": "l4_persistence_type",
 "parameters": {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "l4_persistence_type": "source_ip"
 }
 }
 }
}
```

## Configure the Layer 4 Load Balancer Algorithm

The `l4_lb_algorithm` is used to set the algorithm type for the layer 4 NSX-T Load Balancer service.

The `l4_lb_algorithm` data type is string enumeration that accepts one of the following values:

- `"round_robin"` (default)
- `"least_connection"`
- `"ip_hash"`
- `"weighted_round_robin"`

For example, the following network profile specifies the `weighted_round_robin` as the load balancer algorithm:

```
{
 "name": "example_network_profile",
 "description": "l4_lb_algorithm",
 "parameters": {
 "cni_configurations": {
 "type": "nsxt",
 "parameters": {
 "l4_lb_algorithm": "weighted_round_robin"
 }
 }
 }
}
```

## Using Ingress URL Rewrite

This topic describes how to perform URL rewrite for Kubernetes ingress resources.

## About Support for URL Rewrite for Ingress Resources

Tanzu Kubernetes Grid Integrated Edition with NSX-T supports ingress URL path rewrite using NSX-T v2.5.1+ and NCP v2.5.1+.

All the ingress paths will be rewritten to the provided value. If an ingress has annotation `ingress.kubernetes.io/rewrite-target: /` and has path `/tea`, for example, the URI `/tea` will be rewritten to `/` before the request is sent to the backend service. Numbered capture groups are supported.

## URL Rewrite Example

The following example shows how to implement URL rewrite.

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
 name: cafe-ingress
 annotations:
 kubernetes.io/ingress.class: "nsx"
 ncp/use-regex: "True"
 #/tea/cup will be rewritten to /cup before sending request to endpoint
 ingress.kubernetes.io/rewrite-target: /$1
```

```

spec:
 rules:
 - host: cafe.example.com
 http:
 paths:
 - path: /tea/(.*)
 backend:
 serviceName: tea-svc
 servicePort: 80
 - path: /coffee/(.*)
 backend:
 serviceName: coffee-svc
 servicePort: 80

```

## Creating and Configuring an AWS Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters

This topic describes how to configure an Amazon Web Services (AWS) load balancer for your VMware Tanzu Kubernetes Grid Integrated Edition cluster.

A load balancer is a third-party device that distributes network and application traffic across resources. Using a load balancer can also prevent individual network components from being overloaded by high traffic. For more information about the different types of load balancers used in a Tanzu Kubernetes Grid Integrated Edition deployment see [Load Balancers in TKGI](#).

You can use an AWS Tanzu Kubernetes Grid Integrated Edition cluster load balancer to secure and facilitate access to a Tanzu Kubernetes Grid Integrated Edition cluster from outside the network. You can also [reconfigure](#) your AWS Tanzu Kubernetes Grid Integrated Edition cluster load balancers.

Using an AWS Tanzu Kubernetes Grid Integrated Edition cluster load balancer is optional, but adding one to your Kubernetes cluster can make it easier to manage the cluster using the TKGI API and [kubectl](#).



**Note:** If Kubernetes control plane node VMs are recreated for any reason, you must reconfigure your AWS TKGI cluster load balancers to point to the new control plane VMs.

## Prerequisite

The version of the TKGI CLI you are using must match the version of the Tanzu Kubernetes Grid Integrated Edition tile that you are installing.



**Note:** This procedure uses example commands which you should modify to represent the details of your Tanzu Kubernetes Grid Integrated Edition installation.

## Configure AWS Load Balancer

### Step 1: Define Load Balancer

To define your load balancer using AWS, you must provide a name, select a VPC, specify listeners,

and select subnets where you want to create the load balancer.

Perform the following steps:

1. In a browser, navigate to the [AWS Management Console](#).
2. Under **Compute**, click **EC2**.
3. In the **EC2 Dashboard**, under **Load Balancing**, click **Load Balancers**.
4. Click **Create Load Balancer**.
5. Under **Classic Load Balancer**, click **Create**.
6. On the **Define Load Balancer** page, complete the **Basic Configuration** section as follows:
7. **Load Balancer name**: Name the load balancer. VMware recommends that you name your load balancer `k8s-master-CLUSTERNAME` where `CLUSTERNAME` is a unique name that you provide when creating the cluster. For example, `k8s-master-mycluster`.
  1. **Create LB inside**: Select the VPC where you installed Ops Manager.
  2. **Create an internal load balancer**: Do not enable this checkbox. The cluster load balancer must be internet-facing.
8. Complete the **Listeners Configuration** section as follows:
  1. Configure the first listener as follows.
    - Under **Load Balancer Protocol**, select **TCP**.
    - Under **Load Balancer Port**, enter `8443`.
    - Under **Instance Protocol**, select **TCP**.
    - Under **Instance Port**, enter `8443`.
9. Under **Select Subnets**, select the public subnets for your load balancer in the availability zones where you want to create the load balancer.
10. Click **Next: Assign Security Groups**.

## Step 2: Assign Security Groups

Perform the following steps to assign security groups:

1. On the **Assign Security Groups** page, select one of the following:
  - ◊ **Create a new security group**: Complete the security group configuration as follows:
    1. **Security group name**: Name your security group.
    2. Confirm that your security group includes **Protocol TCP** with **Ports 8443**.
  - ◊ **Select an existing security group**: Select the default security group. The default security group includes **Protocol TCP** with **Ports 8443**.
2. Click **Next: Configure Security Settings**.

## Step 3: Configure Security Settings

On the **Configure Security Settings** page, ignore the warning. SSL termination is done on the Kubernetes API.

## Step 4: Configure Health Check

Perform the following steps to configure the health check:

1. On the **Configure Health Check** page, set the **Ping Protocol** to **TCP**.
2. For **Ping Port**, enter **8443**.
3. Click **Next: Add EC2 Instances**.

## Step 5: Add EC2 Instances

Perform the following steps:

1. Verify the settings under **Availability Zone Distribution**.
2. Click **Add Tags**.

## (Optional) Step 6: Add Tags

Perform the following steps to add tags:

1. Add tags to your resources to help organize and identify them. Each tag consists of a case-sensitive key-value pair.
2. Click **Review and Create**.

## Step 7: Review and Create the Load Balancer

Perform the following steps to review your load balancer details and create your load balancer:

1. On the **Review** page, review your load balancer details and edit any as necessary.
2. Click **Create**.

## Step 8: Create a Cluster

Create a Kubernetes cluster using the AWS-assigned address of your load balancer as the external hostname when you run the `tkgi create-cluster` command. For example:

```
$ tkgi create-cluster my-cluster
-external-hostname example111a6511e9a099028c856be95-155233362.eu-west-1.elb.amazonaws.com
-plan small -num-nodes 10
```

For more information, see [Create a Kubernetes Cluster](#) section of *Creating Clusters*.

## Step 9: Point the Load Balancer to All Control Plane VMs

1. Locate the VM IDs of all control plane node VMs for your cluster. For information about locating the VM IDs, see [Identify Kubernetes Cluster Control Plane VMs](#) in *Creating Clusters*.
2. Navigate to the [AWS console](#).
3. Under EC2, select **Load balancers**.

4. Select the load balancer.
5. On the **Instances** tab, click **Edit instances**.
6. Select all control plane nodes in the list of VMs.
7. Click **Save**.

## Reconfigure AWS Load Balancer

If Kubernetes control plane node VMs are recreated for any reason, you must reconfigure your cluster load balancers to point to the new control plane VMs. For example, after a stemcell upgrade, BOSH recreates the VMs in your deployment.

To reconfigure your AWS cluster load balancer to use the new control plane VMs, do the following:

1. Locate the VM IDs of the new control plane node VMs for the cluster. For information about locating the VM IDs, see [Identify Kubernetes Cluster Control Plane VMs](#) in *Creating Clusters*.
2. Navigate to the [AWS console](#).
3. Under EC2, select **Load balancers**.
4. Select the load balancer.
5. On the **Instances** tab, click **Edit instances**.
6. Select the new control plane nodes in the list of VMs.
7. Click **Save**.

## Creating and Configuring an Azure Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters

This topic describes how to create and configure an Azure load balancer for your VMware Tanzu Kubernetes Grid Integrated Edition cluster. Using an Azure load balancer is optional, but you might want to add one to your Kubernetes cluster to manage the cluster using the TKGI API and Kubernetes CLI (`kubectl`).

A load balancer is a third-party device that distributes network and application traffic across resources. You can use a load balancer to secure and facilitate access to a Tanzu Kubernetes Grid Integrated Edition cluster from outside the network. Using a load balancer can also prevent individual network components from being overloaded by high traffic.



**Note:** If your Kubernetes control plane node VMs are recreated for any reason, you must reconfigure your cluster load balancers to point to the new control plane VMs. For instructions, see [Reconfigure Load Balancer](#).

## Prerequisites

To complete the steps below, you must identify the TKGI API VM. You can find the name in the following ways:

- In the Azure Dashboard, locate the VM tagged with `instance_group:pivotal-container-`

service.

- On the command line, run `bosh vms`.

## Create and Configure a Load Balancer

Follow the steps below to create and configure an Azure load balancer for your Tanzu Kubernetes Grid Integrated Edition cluster.

### Create Load Balancer

1. In a browser, navigate to the [Azure Dashboard](#).
2. Open the **Load Balancers** service.
3. Click **Add**.
4. On the **Create load balancer** page, complete the form as follows:
  1. **Name:** Name the load balancer.
  2. **Type:** Select **Public**.
  3. **SKU:** Select **Standard**.
  4. **Public IP address:** Select **Create new** and name the new IP address.
  5. **Availability zone:** Select an availability zone or **Zone-redundant**.
  6. **Subscription:** Select the subscription which has Tanzu Kubernetes Grid Integrated Edition deployed.
  7. **Resource group:** Select the resource group which has Tanzu Kubernetes Grid Integrated Edition deployed.
  8. **Location:** Select the location group which has Tanzu Kubernetes Grid Integrated Edition deployed.
5. Click **Create**.

### Create Backend Pool

1. From the Azure Dashboard, open the **Load Balancers** service.
2. Click the name of the load balancer that you created in [Create Load Balancer](#).
3. On your load balancer page, locate and record the IP address of your load balancer.
4. In the **Settings** menu, select **Backend pools**.
5. On the **Backend pools** page, click **Add**.
6. On the **Add backend pool** page, complete the form as follows:
  1. **Name:** Name the backend pool.
  2. **Virtual network:** Select the virtual network where the TKGI API VM is deployed.
  3. **Virtual machine:** Select all of the control plane VMs for your cluster. For information about identifying the control plane VM IDs, see [Identify Kubernetes Cluster Control Plane VMs](#) in [Creating Clusters](#).

7. Click **Add**.

## Create Health Probe

1. From the Azure Dashboard, open the **Load Balancers** service.
2. In the **Settings** menu, select **Health probes**.
3. On the **Health probes** page, click **Add**.
4. On the **Add health probe** page, complete the form as follows:
  1. **Name**: Name the health probe.
  2. **Protocol**: Select **TCP**.
  3. **Port**: Enter **8443**.
  4. **Interval**: Enter the interval of time to wait between probe attempts.
  5. **Unhealthy Threshold**: Enter a number of consecutive probe failures that must occur before a VM is considered unhealthy.
5. Click **OK**.

## Create Load Balancing Rule

1. From the Azure Dashboard, open the **Load Balancers** service.
2. In the **Settings** menu, select **Load Balancing Rules**.
3. On the **Load balancing rules** page, click **Add**.
4. On the **Add load balancing rules** page, complete the form as follows:
  1. **Name**: Name the load balancing rule.
  2. **IP Version**: Select **IPv4**.
  3. **Frontend IP address**: Select the appropriate IP address. Clients communicate with your load balancer on the selected IP address and service traffic is routed to the target VM by this NAT rule.
  4. **Protocol**: Select **TCP**.
  5. **Port**: Enter **8443**.
  6. **Backend port**: Enter **8443**.
  7. **Backend Pool**: Select the backend pool that you created in [Create Backend Pool](#).
  8. **Health Probe**: Select the health probe that you created in [Create Health Probe](#).
  9. **Session persistence**: Select **None**.
5. Click **OK**.

## Create Inbound Security Rule

1. From the Azure Dashboard, open the **Security Groups** service.
2. Click the name of the Security Group attached to the subnet where the TKGI API is deployed.

3. In the **Settings** menu for your security group, select **Inbound security rules**.
4. Click **Add**.
5. On the **Add inbound security rule** page, click **Advanced** and complete the form as follows:
  1. **Name**: Name the inbound security rule.
  2. **Source**: Select **Any**.
  3. **Source port range**: Enter **\***.
  4. **Destination**: Select **Any**.
  5. **Destination port range**: Enter **8443**.
6. Click **OK**.

## Verify Hostname Resolution

Verify that the **External hostname** used when creating a Kubernetes cluster resolves to the IP address of the load balancer.

For more information, see [Create a Kubernetes Cluster](#) in *Creating Clusters*.

## Reconfigure Load Balancer

If your Kubernetes control plane node VMs are recreated for any reason, you must reconfigure your cluster load balancers to point to the new control plane VMs. For example, after a stemcell upgrade, BOSH recreates the VMs in your deployment.

To reconfigure your Azure cluster load balancer to use the new control plane VMs, do the following:

1. Identify the VM IDs of the new control plane node VMs for the cluster. For information about identifying the control plane VM IDs, see [Identify Kubernetes Cluster Control Plane VMs](#) in *Creating Clusters*.
2. In a browser, navigate to the [Azure Dashboard](#).
3. Open the **Load Balancers** service.
4. Select the load balancer for your cluster.
5. In the **Settings** menu, select **Backend pools**.
6. Update the VMs list with the new control plane VM IDs.
7. Click **Save**.

## Creating and Configuring a GCP Load Balancer for Tanzu Kubernetes Grid Integrated Edition Clusters

This topic describes how to configure a Google Cloud Platform (GCP) load balancer for a Kubernetes cluster deployed by VMware Tanzu Kubernetes Grid Integrated Edition.



**Note:** Support for GCP is deprecated and will be entirely removed in a future TKGI version.

## Overview

A load balancer is a third-party device that distributes network and application traffic across resources. You can use a load balancer to access a TKGI-deployed cluster from outside the network using the TKGI API and `kubectl`. Using a load balancer can also prevent individual network components from being overloaded by high traffic.

You can configure GCP load balancers only for TKGI clusters that are deployed on GCP.

To configure a GCP load balancer, follow the procedures below:

1. [Create a GCP Load Balancer](#)
2. [Create a DNS Entry](#)
3. [Create the Cluster](#)
4. [Configure Load Balancer Back End](#)
5. [Create a Network Tag](#)
6. [Create Firewall Rules](#)
7. [Access the Cluster](#)

To reconfigure a cluster load balancer, follow the procedures in [Reconfigure Load Balancer](#) below.

## Prerequisites

The procedures in this topic have the following prerequisites:

- To complete these procedures, you must have already configured a load balancer to access the TKGI API. For more information, see [Creating a GCP Load Balancer for the TKGI API](#).
- The version of the TKGI CLI you are using must match the version of the Tanzu Kubernetes Grid Integrated Edition tile that you are installing.

## Configure GCP Load Balancer

Follow the procedures in this section to create and configure a load balancer for TKGI-deployed Kubernetes clusters using GCP. Modify the example commands in these procedures to match your Tanzu Kubernetes Grid Integrated Edition installation.

### Create a GCP Load Balancer

To create a GCP load balancer for your TKGI clusters, do the following:

1. Navigate to the [Google Cloud Platform console](#).
2. In the sidebar menu, select **Network Services > Load balancing**.
3. Click **Create a Load Balancer**.
4. In the **TCP Load Balancing** pane, click **Start configuration**.
5. Click **Continue**. The **New TCP load balancer** menu opens.
6. Give the load balancer a name. For example, `my-cluster`.

7. Click **Frontend configuration** and configure the following settings:
  1. Click **IP**.
  2. Select **Create IP address**.
  3. Give the IP address a name. For example, `my-cluster-ip`.
  4. Click **Reserve**. GCP assigns an IP address.
  5. In the **Port** field, enter `8443`.
  6. Click **Done** to complete front end configuration.
8. Review your load balancer configuration and click **Create**.

## Create a DNS Entry

To create a DNS entry in GCP for your TKGI cluster, do the following:

1. From the GCP console, navigate to **Network Services > Cloud DNS**.
2. Select the DNS zone for your domain. To retrieve your zone name, select the zone you used when you created the TKGI API DNS entry. See the [Create a DNS Entry](#) section in *Creating a GCP Load Balancer for the TKGI API*.
3. Click **Add record set**.
4. Under **DNS Name**, enter a subdomain for the load balancer. For example, if your domain is `example.com`, enter `my-cluster` in this field to use `my-cluster.example.com` as your TKGI cluster load balancer hostname.
5. Under **Resource Record Type**, select **A** to create a DNS address record.
6. Enter a value for **TTL** and select a **TTL Unit**.
7. Enter the GCP-assigned IP address you created in [Create a Load Balancer](#) above.
8. Click **Create**.

## Create the Cluster

To create a cluster, follow the steps input [Create a Kubernetes Cluster](#) section of *Creating Clusters*. Use the TKGI cluster hostname from the above step as the external hostname when you run the `tkgi create-cluster` command.

## Configure Load Balancer Back End

To configure the back end of the load balancer, do the following:

1. Record the ID for your control plane node VMs by doing one of the following:
  - ❖ Complete [Identify Kubernetes Cluster Control Plane VMs](#) in *Creating Clusters*.
  - ❖ Complete the following procedure:
    1. Log in to TKGI by running the following command:

```
tkgi login -a TKGI-API -u USERNAME -k
```

Where:

- **TKGI-API** is the domain name for the TKGI API that you entered in **Ops Manager > Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN)**. For example, `api.tkgi.example.com`.
  - **USERNAME** is your user name.
2. Locate the control plane node IP addresses by running the following command:
- ```
tkgi cluster CLUSTER-NAME
```
- Where `CLUSTER-NAME` is the unique name for your cluster.
- From the output of this command, record the value of **Kubernetes Master IP(s)**. This value lists the IP addresses of all control plane node VMs in the cluster.
3. Navigate to the [Google Cloud Platform console](#).
 4. From the sidebar menu, navigate to **Compute Engine > VM instances**.
 5. Filter the VMs using the network name you provided when you deployed Ops Manager on GCP.
 6. Record the IDs of the control plane node VMs associated with the IP addresses you recorded in the above step. The above IP addresses appear under the **Internal IP** column.



Note: If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

2. In the [Google Cloud Platform console](#), from the sidebar menu, navigate to **Network Services > Load balancing**.
3. Select the load balancer that you created for the cluster and click **Edit**.
4. Click **Backend configuration** and configure the following settings:
 1. Select all the control plane node VMs for your cluster from the dropdown.



Warning: If control plane VMs are recreated for any reason, such as a stemcell upgrade, you must reconfigure the load balancer to target the new control plane VMs. For more information, see the [Reconfigure Load Balancer](#) section below.

2. Specify any other configuration options you require and click **Update** to complete

back end configuration.



Note: For clusters with multiple control plane node VMs, health checks on port 8443 are recommended.

Create a Network Tag

To create a network tag, do the following:

1. In the Google Cloud Platform sidebar menu, select **Compute Engine > VM instances**.
2. Filter to find the control plane instances of your cluster. Type `master` in the **Filter VM Instances** search box and press **Enter**.
3. Click the name of the control plane instances. The **VM instance details** menu opens.
4. Click **Edit**.
5. Click in the **Network tags** field and type a human-readable name in lowercase letters. Press **Enter** to create the network tag.
6. Scroll to the bottom of the screen and click **Save**.

Create Firewall Rules

To create firewall rules, do the following:

1. In the Google Cloud Platform sidebar menu, select **VPC Network > Firewall Rules**.
2. Click **Create Firewall Rule**. The **Create a firewall rule** menu opens.
3. Give your firewall rule a human-readable name in lower case letters. For ease of use, you might want to align this name with the name of the load balancer you created in [Create a GCP Load Balancer](#).
4. In the **Network** menu, select the VPC network on which you have deployed the Tanzu Kubernetes Grid Integrated Edition tile.
5. In the **Direction of traffic** field, select **Ingress**.
6. In the **Action on match** field, select **Allow**.
7. Confirm that the **Targets** menu is set to `Specified target tags` and enter the tag you made in [Create a Network Tag](#) in the **Target tags** field.
8. In the **Source filter** field, choose an option to filter source traffic.
9. Based on your choice in the **Source filter** field, specify IP addresses, Subnets, or Source tags to allow access to your cluster.
10. In the **Protocols and ports** field, choose **Specified protocols and ports** and enter the port number you specified in [Create a GCP Load Balancer](#), prepended by `tcp:..`. For example: `tcp:8443`.
11. Specify any other configuration options you require and click **Done** to complete front end configuration.
12. Click **Create**.

Access the Cluster

To complete cluster configuration, do the following:

- From your local workstation, run:

```
tkgi get-credentials CLUSTER-NAME`
```

Where `CLUSTER-NAME` is the unique name for your cluster.

For example:

```
$ tkgi get-credentials tkgi-example-cluster
Fetching credentials for cluster tkgi-example-cluster. Context set for
cluster tkgi-example-cluster.
```

The `tkgi get-credentials` command creates a local `kubeconfig` that enables you to manage the cluster. For more information about the `tkgi get-credentials` command, see [Retrieving Cluster Credentials and Configuration](#).



Note: If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

- Run `kubectl cluster-info` to confirm you can access your cluster using the Kubernetes CLI.

See [Managing Tanzu Kubernetes Grid Integrated Edition](#) for information about checking cluster health and viewing cluster logs.

Reconfigure Load Balancer

If Kubernetes control plane node VMs are recreated for any reason, you must reconfigure your cluster load balancers to point to the new control plane VMs. For example, after a stemcell upgrade, BOSH recreates the VMs in your deployment.

To reconfigure your GCP cluster load balancer to use the new control plane VMs, do the following:

- Locate the VM IDs of the new control plane node VMs for the cluster. For information about locating the VM IDs, see [Identify Kubernetes Cluster Control Plane VMs](#) in [Creating Clusters](#).
- Navigate to the [GCP console](#).
- In the sidebar menu, select **Network Services > Load balancing**.
- Select your cluster load balancer and click **Edit**.
- Click **Backend configuration**.
- Click **Select existing instances**.
- Select the new control plane VM IDs from the dropdown. Use the VM IDs you located in the

first step of this procedure.

8. Click **Update**.

Configuring Ingress Routing

This topic provides resources for configuring an ingress controller on VMware Tanzu Kubernetes Grid Integrated Edition.

For information about configuring an ingress controller using NSX-T, see [Configuring Ingress Resources and Load Balancer Services](#).

Overview

In Kubernetes, an ingress is an API object that manages external access to the services in a cluster. You can use ingress rules to provide HTTP or HTTPS routes to services within the cluster instead of creating a load balancer. For more information, see [Ingress](#) in the Kubernetes documentation.

The cluster must have an ingress controller running. You define ingress resource configuration in the manifest of your Kubernetes deployment, and then use wildcard DNS entries to route traffic to the exposed ingress resource.

To configure an ingress controller, you must do the following:

1. [Deploy a Kubernetes Ingress Controller](#)
2. [Configure DNS](#)
3. (Optional) [Configure TLS](#)
4. [Deploy an App to the Cluster](#)

Prerequisites

Before you configure an ingress controller, you must have the following:

- A TKGI-deployed cluster with its own load balancer. See [Creating Clusters](#).
- A wildcard DNS record that points to the cluster load balancer.

Deploy a Kubernetes Ingress Controller

You can deploy an ingress controller of your choice to your Kubernetes cluster. For a list of ingress controllers that Kubernetes supports, see [Ingress Controllers](#) in the Kubernetes documentation.



Note: For information about configuring an ingress controller using NGINX on Amazon Web Services (AWS), Azure, or Google Cloud Platform (GCP), see [How to set up an Ingress Controller for a TKGI cluster](#) in the Knowledge Base.

To deploy an open source ingress controller to a TKGI cluster, do the following:

1. To set the kubectl context for the cluster where you want to deploy the ingress controller, run the following command:

```
[REDACTED]
```

```
tkgi get-credentials CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of your TKGI-deployed Kubernetes cluster.

For example:

```
$ tkgi get-credentials tkgi-example-cluster
```

Fetching credentials for cluster tkgi-example-cluster.

Context set for cluster tkgi-example-cluster.

You can now switch between clusters by using:

```
$ kubectl config use-context &lt;cluster-name&gt;
```



Note: If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

2. To verify a DNS service is enabled for your Kubernetes cluster, run the following command:

```
kubectl cluster-info
```

If a DNS service is enabled, the DNS service's URL is included in the `kubectl cluster-info` output.

For example:

```
$ kubectl cluster-info
Kubernetes master is running at https://104.197.5.247
elasticsearch-logging is running at https://104.197.5.247/api/v1/namespaces/kube-system/services/elasticsearch-logging/proxy
kibana-logging is running at https://104.197.5.247/api/v1/namespaces/kube-system/services/kibana-logging/proxy
CoreDNS is running at https://104.197.5.247/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
grafana is running at https://104.197.5.247/api/v1/namespaces/kube-system/services/monitoring-grafana/proxy
```

The current default Kubernetes cluster DNS service is `CoreDNS`. The example output above includes the URL for this DNS service, indicating it is running.

If a DNS service is not running for your cluster, enable the `CoreDNS` service:

1. Navigate to Ops Manager and click the **BOSH Director** tile.
1. Click the **Director Config** pane.

1. Select the **Enable Post Deploy Scripts** checkbox.
1. Click **Review Pending Changes**, and then **Apply Changes**.
1. Delete the cluster, and then re-create the cluster.
3. Follow the installation instructions for the Kubernetes ingress controller you choose to deploy. For example, see the installation guide in the [Istio documentation](#).

Configure DNS

After you deploy an ingress controller to your cluster, locate the HTTP port number that the ingress rules expose. Configure DNS to point to the exposed port on your Kubernetes worker node VMs.

To configure DNS for your cluster, do the following:

1. Run `kubectl get services` in the namespace where you deployed the ingress controller. For example, if you deployed Istio, run the following command:

```
kubectl --namespace=istio-system get services
```

In the output of this command, locate the exposed HTTP port.

For example:

```
$ kubectl -namespace=istio-system get services NAME          TYPE
      CLUSTER-IP     EXTERNAL-IP    PORT(S)  istio-ingress  LoadB
alancer   10.100.200.200  <pending>        80:30822/TCP,443:31441/TCP
```

In the example above, the exposed HTTP port is 30822.

2. List the IP addresses for the Kubernetes worker node VMs by running the following command:
- ```
kubectl -o jsonpath='{.items[*].status.addresses[0].address}' get nodes
```
3. Configure your load balancer to point to the Kubernetes worker node VMs, using the IP addresses you located in the previous step and the exposed port number you located in the first step.

## (Optional) Configure TLS

Enable Transport Layer Security (TLS) for the domain you configured for the cluster.

To configure TLS, do the following:

1. (Optional) Run the following command to generate a self-signed certificate:

```
openssl req -x509 \
-nodes -newkey rsa:4096 \
-keyout KEY-PATH.pem \
-out CERT-PATH.pem \
-days 365 \
```

```
-subj "/CN=*.TKGI.EXAMPLE.COM"
```

Where:

- ◊ `KEY-PATH.pem` is the file path for the key you are generating.
- ◊ `CERT-PATH.pem` is the file path for the certificate you are generating.
- ◊ `*.TKGI.EXAMPLE.COM` is the wildcard domain you configured in [Configure DNS](#).

2. Upload your TLS certificate and key to your ingress controller namespace by running the following command:

```
kubectl -n INGRESS-NAMESPACE create secret tls INGRESS-CERT \
--key KEY-PATH.pem --cert CERT-PATH.pem
```

Where:

- ◊ `INGRESS-CERT` is a name you provide for the Kubernetes secret that contains your TLS certificate and key pair.
- ◊ `KEY-PATH.pem` is the file path for your TLS key.
- ◊ `CERT-PATH.pem` is the file path for your TLS certificate.

For example:

```
$ kubectl -n istio-system create secret tls istio-ingress-certs
-key /tmp/tls.key -cert /tmp/tls.crt
```

## Deploy an App to the Cluster

When your cluster has an ingress controller running and DNS configured, you can deploy an app to the cluster that uses the ingress rules.

To deploy an app that uses ingress rules, do the following:

1. Deploy your app manifest by running the following command:

```
kubectl create -f YOUR-APP.yml
```

Where `YOUR-APP.yml` is the file path for your app manifest.

2. In the app manifest for your ingress controller, change the value of the `host:` property to match the wildcard domain you configured in [Configure DNS](#) above.
3. Deploy your ingress controller app manifest by running the following command:

```
kubectl create -f INGRESS-CONTROLLER.yml
```

Where `INGRESS-CONTROLLER.yml` is the file path for your ingress controller app manifest.

4. Navigate to the fully qualified domain name (FQDN) you defined in your app manifest and confirm that you can access your app workload.
5. (Optional) If you configured TLS, do the following:

1. Add the following to your ingress controller manifest to enable TLS:

```
spec:
 tls:
 - secretName: INGRESS-CERT
 rules:
 - host: INGRESS.TKGI.EXAMPLE.COM
```

Where:

- `INGRESS-CERT` is the name of the Kubernetes secret that contains your TLS certificate and key pair.
- `INGRESS.TKGI.EXAMPLE.COM` is the domain you defined for your app in the app manifest.

2. Redeploy the ingress controller manifest to update the ingress service by running the following command:

```
kubectl replace -f INGRESS-CONTROLLER.yml
```

Where `INGRESS-CONTROLLER.yml` is the file path for your ingress controller app manifest.

3. Navigate to the FQDN you defined in your app manifest and confirm that you can access your app workload.

## Using Admission Control Plugins for Tanzu Kubernetes Grid Integrated Edition Clusters

### Topic provided by VMware

This section describes how to activate, use, and deactivate admission control plugins for VMware Tanzu Kubernetes Grid Integrated Edition clusters.

For more information about Admission Controllers, see [Using Admission Controllers](#) in the Kubernetes documentation.

Tanzu Kubernetes Grid Integrated Edition supports three admission control plugins. See the following topics for details on each:

- [Enabling the PodSecurityPolicy Admission Plugin for Tanzu Kubernetes Grid Integrated Edition Clusters and Using Pod Security Policies](#)
- [Enabling the SecurityContextDeny Admission Plugin for Tanzu Kubernetes Grid Integrated Edition Clusters](#)

To deactivate one or more Admission Plugins, refer to the following topic:

- [Deactivating Admission Control Plugins for Tanzu Kubernetes Grid Integrated Edition Clusters](#)

## Pod Security Admission in Tanzu Kubernetes Grid Integrated

## Edition

This topic describes how to use Kubernetes Pod Security Admission (PSA) with VMware Tanzu Kubernetes Grid Integrated Edition (TKGI).

**Note:** Support for Kubernetes Pod Security Policy (PSP) has been removed in Kubernetes v1.25.

## About Pod Security Admission

PSA is the Kubernetes-recommended way to implement security standards. TKGI supports the built-in PSA in Kubernetes. PSA is enabled in TKGI, by default.

For more information on PSA, see [Pod Security Admission](#) in the Kubernetes documentation.

## Pod Security Admission and TKGI

**Note:** To control the PSA security permissions in a TKGI namespace, you must have the privileges to create, update, or patch the namespace. To ensure security of the system, restrict the namespace permissions to the trusted user accounts.

The following table describes the required PSA level for TKGI System namespaces:

| TKGI System Namespace      | PSA Level  |
|----------------------------|------------|
| kube-system                | Privileged |
| nsx-system                 | Restricted |
| pks-system                 | Privileged |
| pks-system-host-monitoring | Restricted |
| vmware-system-csi          | Baseline   |

For more information on implementing Pod Security Standards with namespace labels, see [Enforce Pod Security Standards with Namespace Labels](#) in the Kubernetes documentation.

## Migrate from PSP to PSA Controller

To migrate from PSP to PSA Controller, see [Migrate from PodSecurityPolicy to the Built-In PodSecurity Admission Controller](#) in the Kubernetes documentation.

## Enabling the SecurityContextDeny Admission Plugin

### Topic provided by VMware

This section describes how to enable the SecurityContextDeny admission controller for VMware Tanzu Kubernetes Grid Integrated Edition clusters.

## About the SecurityContextDeny Admission Plugin

The SecurityContextDeny admission controller plugin will deny any pod that attempts to set certain escalating Security Context fields.

In Kubernetes, a [security context](#) defines privilege and access control settings for a pod or container. The securityContext field is a PodSecurityContext object. For more information, see [Set the security context for a Pod](#) in the Kubernetes documentation.

## When to Enable the SecurityContextDeny Admission Plugin

The SecurityContextDeny admission plugin must be enabled if a cluster does not use pod security admission (PSA) to restrict the set of values a security context can take. See [Enabling and Using Pod Security Policies](#) for more information.

PSA is the preferred method for providing a more secure Kubernetes environment. However, PSA has administrative overhead. Enabling the SecurityContextDeny is a stopgap method of providing a more secure Kubernetes environment when it is not feasible to use PSA. If you plan to use PSA in the future, consider enabling the SecurityContextDeny admission plugin as an interim security measure.

## Impact of Enabling the SecurityContextDeny Admission Plugin

This section describes the impact of enabling the SecurityContextDeny admission control plugin for new and existing cluster plans.

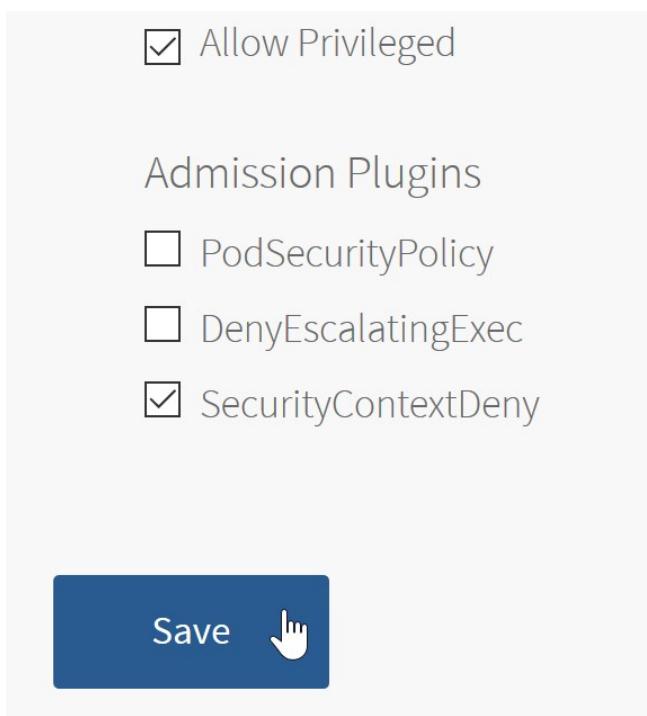
**New Cluster.** If you enable the SecurityContextDeny admission plugin in a plan and deploy a new Kubernetes cluster based on that plan, cluster users will not be able to create securityContext capabilities on that cluster.

**Existing Cluster.** If you enable the SecurityContextDeny admission plugin in a plan and update a Kubernetes cluster, cluster users will no longer be able to create securityContext capabilities on that cluster. This assumes you enable **Upgrade all clusters errand** or update your cluster individually through the TKGI Command Line Interface (TKGI CLI).

## Enabling the SecurityContextDeny Admission Plugin

To enable the SecurityContextDeny admission plugin:

1. In the TKGI tile, select the desired Plan, such as Plan 1.
2. At the bottom of the configuration panel, select the **SecurityContextDeny** option.



3. Click **Save**.
4. On the Installation Dashboard, click **Review Pending Changes**.
5. For Tanzu Kubernetes Grid Integrated Edition, verify that **Upgrade all clusters errand** is enabled.
6. Click **Apply Changes** to deploy the cluster with the admission plugin enabled.

Alternatively, instead of enabling **Upgrade all clusters errand**, you can upgrade individual Kubernetes clusters through the TKGI Command Line Interface (TKGI CLI). For instructions on upgrading individual Kubernetes clusters, see [Upgrading Clusters](#).

## Deactivating Admission Control Plugins for Tanzu Kubernetes Grid Integrated Edition Clusters

### Topic provided by VMware

This section describes how to deactivate one or more admission control plugins for VMware Tanzu Kubernetes Grid Integrated Edition clusters. For more information, see [Using Admission Control Plugins for Tanzu Kubernetes Grid Integrated Edition Clusters](#).

## Deactivating a Single Admission Control Plugin

To deactivate a single admission control plugin, do the following:

1. Log in to Ops Manager.
2. Click the Tanzu Kubernetes Grid Integrated Edition tile.
3. Select the plan where you configured the admission control plugin, such as **Plan 1**.
4. Deselect the admission control plugin.
5. Click **Save**.

6. In the **Errands** pane, verify that **Upgrade all clusters errand** is activated.
7. Return to **Installation Dashboard** and select **Review Pending Changes**.
8. Click **Apply Changes**.

Alternatively, instead of enabling **Upgrade all clusters errand**, you can upgrade individual Kubernetes clusters through the TKGI Command Line Interface (TKGI CLI). For instructions on upgrading individual Kubernetes clusters, see [Upgrading Clusters](#).

## Deactivating an Orphaned Admission Control Plugin

The Ops Manager UI does not let you deselect (deactivate) all admission control plugins.

In other words, after an admission control plugin is activated, the Ops Manager UI requires that at least one admission control plugin checkbox is selected (activated).

To deactivate an orphaned Admission control Plugin, complete the following workflow:

1. Obtain the FQDN, user name and password of your Ops Manager.
2. Authenticate into the Ops Manager API and retrieve a UAA access token to access Ops Manager. For more information, see [Using the Ops Manager API](#).
3. Obtain the BOSH deployment name for the Tanzu Kubernetes Grid Integrated Edition tile by doing one of the following options:
  1. Option 1: Use the Ops Manager API:
    1. In a terminal, run the following command:

```
curl -i "https://OPS-MAN-FQDN/api/v0/staged/products" -X GET -H "Authorization: Bearer UAA-ACCESS-TOKEN" -k
```
    2. In the output, locate the `installation_name` that begins with `pivotal-container-service`.
    3. Copy the entire BOSH deployment name, including the unique GUID. For example, `pivotal-container-service-4b48fc5b704d54c6c7de`.
  2. Option 2: Use the Ops Manager UI:
    1. In Ops Manager, click on the Tanzu Kubernetes Grid Integrated Edition tile.
    2. Copy the BOSH deployment name including the GUID from the URL:

The deployment name contains “pivotal-container-service” and a unique GUID string. For example, `pivotal-container-service-4b48fc5b704d54c6c7de.`

- To deactivate the orphaned admission control plugin, run the following Ops Manager API command:

```
curl -i "https://OPS-MAN-FQDN/api/v0/staged/pivotal-container-service-GUID/properties" \
-H "Authorization: Bearer UAA-ACCESS-TOKEN" \
-X PUT -d '{"properties": {"properties.PLAN-NUMBER_selector.active.admission_plugins": {"value": []}}}' \
-H "Content-Type: application/json"
```

Where:

- `OPS-MAN-FQDN` is the URL of your Ops Manager.
- `pivotal-container-service-GUID` is the BOSH deployment name of your Tanzu Kubernetes Grid Integrated Edition that you retrieved earlier in this procedure.
- `UAA-ACCESS-TOKEN` is the UAA token you retrieved earlier in this procedure.
- `PLAN-NUMBER` is the plan configuration you want to update. For example, `plan1` or `plan2`.

For example:

```
$ curl -i "https://pcf.example.com/api/v0/staged/products/pivotal-container-service-4b48fc5b704d54c6c7de/properties"

-H "Authorization: Bearer aBcdEfg0hIJKlm123.e"

-X PUT -d '{"properties": {"properties.plan1_selector.active.admission_plugins": {"value": []}}}'
```

```
-H "Content-Type: application/json"
```

5. From the output, verify that the command returns a [HTTP 200](#) status code.
6. Validate your manifest change in the Ops Manager UI. Do the following:
  1. Log in to Ops Manager.
  2. Select **Review Pending Changes**.
  3. On the Review Pending Changes pane, navigate to the Tanzu Kubernetes Grid Integrated Edition section and select **SEE CHANGES**.
  4. Verify that the admission control plugins are displayed as removed in the **Manifest** section. For example:

**Manifest**

```

instance_groups:
- name: pivotal-container-service
 jobs:
 - name: pks-nsx-t-osb-proxy
 properties:
 plans:
 - name: small
 properties:
 - enabled_admission_plugins:
 - pod_security_policy
 - denyEscalatingExec
 properties:
 service_catalog:
 plans:
 - name: small
 properties:
 - enabled_admission_plugins:
 - podSecurityPolicy
 - denyEscalatingExec

```

7. Click **Apply Changes**.

## Retrieving Cluster Credentials and Configuration

This topic describes how to use the `tkgi get-credentials` command in VMware Tanzu Kubernetes Grid Integrated Edition using the TKGI Command Line Interface (TKGI CLI).

The `tkgi get-credentials` command performs the following actions:

- Fetch the cluster's kubeconfig

- Add the cluster's kubeconfig to the existing kubeconfig
- Create a new kubeconfig, if none exists
- Switch the context to the `CLUSTER-NAME` provided

When you run `tkgi get-credentials CLUSTER-NAME`, TKGI sets the context to the cluster you provide as the `CLUSTER-NAME`. TKGI binds your username to the cluster and populates the kubeconfig file on your local workstation with cluster credentials and configuration.

The default path for your kubeconfig is `$HOME/.kube/config`.

If you access multiple clusters, you can choose to use a custom kubeconfig file for each cluster. To save cluster credentials to a custom kubeconfig, use the `KUBECONFIG` environment variable when you run `tkgi get-credentials`. For example:

```
$ KUBECONFIG=/path/to/my-cluster.config tkgi get-credentials my-cluster
```

## Retrieve Cluster Credentials

Perform the following steps to populate your local kubeconfig with cluster credentials and configuration:

1. On the command line, run the following command to log in:

```
tkgi login -a TKGI-API -u USERNAME -k
```

Where:

- ◊ `TKGI-API` is the domain name for the TKGI API that you entered in **Ops Manager > Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN)**. For example, `api.tkgi.example.com`.
- ◊ `USERNAME` is your user name.

See [Logging in to Tanzu Kubernetes Grid Integrated Edition](#) for more information about the `tkgi login` command.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

2. Run the following command:

```
tkgi get-credentials CLUSTER-NAME
```

Where `CLUSTER-NAME` is the unique name for your cluster.

For example:

```
$ tkgi get-credentials tkgi-example-cluster
```

Fetching credentials for cluster tkgi-example-cluster. Context set for cluster tkgi-example-cluster.

You can now switch between clusters by using: \$kubectl config use-context <cluster-name>



**Note:** If you enable OpenID Connect (OIDC) in the Tanzu Kubernetes Grid Integrated Edition tile, TKGI requires your password to run the `tkgi get-credentials CLUSTER-NAME` command. This allows TKGI to retrieve valid tokens for the kubeconfig file. You can provide your password at the prompt or as the `TKGI_USER_PASSWORD` environment variable. For more information, see the *Configure OpenID Connect* section of [Installing Tanzu Kubernetes Grid Integrated Edition](#) for your IaaS.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

## Run kubectl Commands

After TKGI populates your kubeconfig, you can use the Kubernetes Command Line Interface (kubectl) to run commands against your Kubernetes clusters.

See [Installing the Kubernetes CLI](#) for information about installing kubectl.

For information about using kubectl, see [Command line tool \(kubectl\)](#) in the Kubernetes documentation.

## Retrieving Cluster Credentials and Configuration

This topic describes how to use the `tkgi get-credentials` command in VMware Tanzu Kubernetes Grid Integrated Edition using the TKGI Command Line Interface (TKGI CLI).

The `tkgi get-credentials` command performs the following actions:

- Fetch the cluster's kubeconfig
- Add the cluster's kubeconfig to the existing kubeconfig
- Create a new kubeconfig, if none exists
- Switch the context to the `CLUSTER-NAME` provided

When you run `tkgi get-credentials CLUSTER-NAME`, TKGI sets the context to the cluster you provide as the `CLUSTER-NAME`. TKGI binds your username to the cluster and populates the kubeconfig file on your local workstation with cluster credentials and configuration.

The default path for your kubeconfig is `$HOME/.kube/config`.

If you access multiple clusters, you can choose to use a custom kubeconfig file for each cluster. To save cluster credentials to a custom kubeconfig, use the `KUBECONFIG` environment variable when you run `tkgi get-credentials`. For example:

```
$ KUBECONFIG=/path/to/my-cluster.config tkgi get-credentials my-cluster
```

## Retrieve Cluster Credentials

Perform the following steps to populate your local kubeconfig with cluster credentials and configuration:

1. On the command line, run the following command to log in:

```
tkgi login -a TKGI-API -u USERNAME -k
```

Where:

- TKGI-API is the domain name for the TKGI API that you entered in **Ops Manager > Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN)**. For example, `api.tkgi.example.com`.
- USERNAME is your user name.

See [Logging in to Tanzu Kubernetes Grid Integrated Edition](#) for more information about the `tkgi login` command.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

2. Run the following command:

```
tkgi get-credentials CLUSTER-NAME
```

Where `CLUSTER-NAME` is the unique name for your cluster.

For example:

```
$ tkgi get-credentials tkgi-example-cluster
```

```
Fetching credentials for cluster tkgi-example-cluster. Context set for
```

```
cluster tkgi-example-cluster.
```

You can now switch between clusters by using: `$kubectl config use-context <cluster-name>`



**Note:** If you enable OpenID Connect (OIDC) in the Tanzu Kubernetes Grid Integrated Edition tile, TKGI requires your password to run the `tkgi get-credentials <CLUSTER-NAME>` command. This allows TKGI to retrieve valid tokens for the kubeconfig file. You can provide your password at the prompt or as the `TKGI_USER_PASSWORD` environment variable. For more information, see the *Configure OpenID Connect* section of [Installing Tanzu Kubernetes Grid Integrated Edition](#) for your IaaS.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

## Run kubectl Commands

After TKGI populates your kubeconfig, you can use the Kubernetes Command Line Interface (kubectl) to run commands against your Kubernetes clusters.

See [Installing the Kubernetes CLI](#) for information about installing kubectl.

For information about using kubectl, see [Command line tool \(kubectl\)](#) in the Kubernetes documentation.

## Managing Cluster Access and Permissions

This topic describes how to grant Kubernetes cluster access and namespace permissions to Kubernetes users in VMware Tanzu Kubernetes Grid Integrated Edition.

### Overview

Tanzu Kubernetes Grid Integrated Edition admin users can grant Kubernetes users, such as developers, permissions to specific clusters.

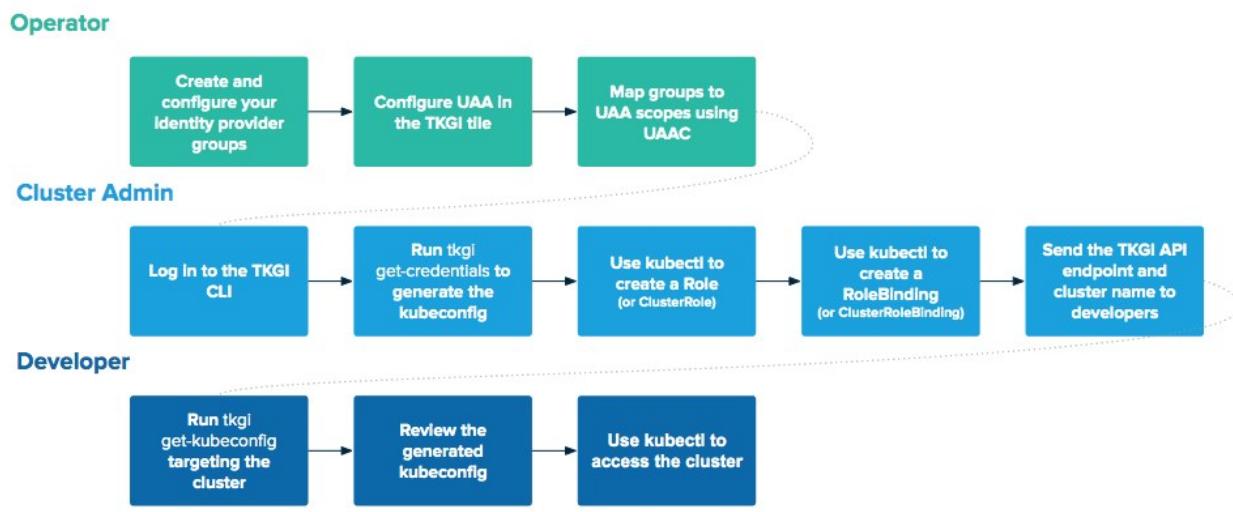
If you are an Tanzu Kubernetes Grid Integrated Edition admin user, you can do the following:

- Grant user access to a cluster with a `ClusterRole` or a namespace within a cluster with a `Role`. See [Grant Cluster Access to a User](#) below.
- Grant group access to a cluster with a `ClusterRole` or a namespace within a cluster with a `Role`. See [Grant Cluster Access to a Group](#) below.

After you grant user or group access to an Tanzu Kubernetes Grid Integrated Edition-provisioned cluster, Kubernetes users can connect to the cluster through the Kubernetes CLI (`kubectl`). Kubernetes users cannot create, resize, or delete clusters.

## Example Workflow

The following diagram outlines the workflow to grant cluster access to users who belong to an identity provider group:



[View a larger version of this image.](#)

For more information, see [RoleBinding and ClusterRoleBinding](#) and [Default Roles and Role Bindings](#) in the Kubernetes documentation.

## Prerequisites

Before setting up cluster access for users in Tanzu Kubernetes Grid Integrated Edition, you must have the following:

- Access to an Tanzu Kubernetes Grid Integrated Edition admin user account. For information about how to create Tanzu Kubernetes Grid Integrated Edition admin users, see [Managing Tanzu Kubernetes Grid Integrated Edition Users with UAA](#).
- Fully qualified domain name (FQDN) of your TKG I deployment.
- OpenID Connect (OIDC) provider for your Kubernetes clusters, configured using one or both of the following:
  - Global OIDC provider configuration for all clusters in [Ops Manager Installation Dashboard > Tanzu Kubernetes Grid Integrated Edition > Settings > UAA > Configure created clusters to use UAA as the OIDC provider](#). For instructions, see [UAA](#) in the *Installing* topic for your IaaS.
  - Custom OIDC provider configuration for individual clusters through a Kubernetes profile. For instructions, see [Adding an OIDC Provider](#).

## Grant Cluster Access to a User

To grant cluster access to a user, do the following:

1. Log in to Tanzu Kubernetes Grid Integrated Edition by running following command:

```
tkgi login -u USERNAME -p PASSWORD -a TKGI-API --ca-cert CERT-PATH
```

Where:

- ◊ **USERNAME** is your cluster admin username.
- ◊ **PASSWORD** is your cluster admin password.
- ◊ **TKGI-API** is the FQDN you use to access the TKGI API.
- ◊ **CERT-PATH** is the path to your root CA certificate.

Provide the certificate to validate the TKGI API certificate with SSL.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

2. Confirm that you can successfully connect to a cluster and use kubectl as a cluster admin by running the following command:

```
tkgi get-credentials CLUSTER-NAME
```

This step creates a `ClusterRoleBinding` for the cluster admin.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

3. When prompted, re-enter your password.
4. Create a YAML file for either `Role` or `ClusterRole`. Use the following example as a template:

```
kind: ROLE-TYPE
apiVersion: rbac.authorization.k8s.io/v1
metadata:
 namespace: NAMESPACE
 name: ROLE-OR-CLUSTER-ROLE-NAME
rules:
- apiGroups:
 resources: RESOURCE
 verbs: API-REQUEST-VERB
```

Where:

- ◊ `ROLE-TYPE` is the type of role you are creating. This must be either `Role` or `ClusterRole`.
  - ◊ `NAMESPACE` is the namespace within the cluster. This is omitted when creating a `ClusterRole`.
  - ◊ `ROLE-OR-CLUSTER-ROLE-NAME` is the name of the `Role` or `ClusterRole` you are creating. This name is created by the cluster admin.
  - ◊ `RESOURCE` is the resource you are granting access to. It must be specified in a comma-separated array. For example: `["pod-reader"]`.
  - ◊ `API-REQUEST-VERB` is the request verb used to specify resource requests. For more information, see [Determine the Request Verb](#) in the Kubernetes documentation.
5. Create the `Role` or `ClusterRole` resource defined in your YAML file by running the following command:

```
kubectl create -f ROLE-CONFIGURATION.yml
```

Where `ROLE-CONFIGURATION.yml` is the YAML file you created in the above step.

6. Create a YAML file containing either a `ClusterRoleBinding` or a `RoleBinding` for the Kubernetes end user. Use the following example as a template:

```
kind: ROLE-BINDING-TYPE
apiVersion: rbac.authorization.k8s.io/v1
metadata:
 name: ROLE-OR-CLUSTER-ROLE-BINDING-NAME
 namespace: NAMESPACE
subjects:
- kind: User
 name: USERNAME
 apiGroup: rbac.authorization.k8s.io
roleRef:
 kind: ROLE-TYPE
 name: ROLE-OR-CLUSTER-ROLE-BINDING-NAME
 apiGroup: rbac.authorization.k8s.io
```

Where:

- ◊ `ROLE-BINDING-TYPE` is the type of role binding you are creating. This must be either `RoleBinding` or `ClusterRoleBinding`.
- ◊ `ROLE-OR-CLUSTER-ROLE-BINDING-NAME` is the name of the role binding. This is given by the cluster admin.
- ◊ `NAMESPACE` is the namespace within the cluster. This is omitted when creating a `ClusterRole`.
- ◊ `USERNAME` is the Kubernetes end user username. This is the username created for your organization's LDAP or SAML identity provider.



**Note:** If you configured an OIDC username prefix in [Ops Manager Installation Dashboard > Tanzu Kubernetes Grid Integrated Edition >](#)

**Settings > UAA** or in a Kubernetes profile, you must prepend `USERNAME` with the prefix you configured. For more information, see [UAA](#) in the *Installing* topic for your IaaS and [Adding an OIDC Provider](#).

\* `ROLE-TYPE` is the type of role you created in the previous step. This must be either `Role` or `ClusterRole`.

\* `ROLE-OR-CLUSTER-ROLE-NAME` is the name of the `Role` or `ClusterRole` you created in the previous step.

7. Create the `RoleBinding` or `ClusterRoleBinding` resource defined in your YAML file by running following command:

```
kubectl apply -f ROLE-BINDING-CONFIGURATION.yml
```

Where `ROLE-BINDING-CONFIGURATION.yml` is the YAML file you created in the above step.

8. Share the following information with your Kubernetes end users:

- TKGI API FQDN
- Cluster name

## Obtain Cluster Access as a User

To obtain access to a Tanzu Kubernetes Grid Integrated Edition-provisioned cluster, the end user must do the following:

1. Fetch the kubeconfig file by running one of the following command:
  - If you want to validate the TKGI API certificate with SSL, run the following command:

```
tkgi get-kubeconfig CLUSTER-NAME -u USERNAME -a TKGI-API --ca-cert CERT-PATH
```

Where:

- `CLUSTER-NAME` is the cluster name provided by the cluster admin.
- `USERNAME` is the Kubernetes end user username. This is the username created for your organization's LDAP or SAML identity provider.
- `TKGI-API` is the FQDN you use to access the TKGI API.
- `CERT-PATH` is the path to your root CA certificate. Provide the certificate to validate the TKGI API certificate with SSL.

For example:

```
$ tkgi get-kubeconfig my-cluster -u naomi -a api.tkgi.example.com
 -ca-cert /var/tempest/workspaces/default/root_ca_certificate
```

- If your CA is trusted and you want to skip SSL validation, run the following command:

```
tkgi get-kubeconfig CLUSTER-NAME -u USERNAME -a TKGI-API -k
```

Where `-k` is the shortcut flag to skip SSL validation.

For example:

```
$ tkgi get-kubeconfig my-cluster -u naomi -a api.tkgi.example.com -k
```



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

2. When prompted, enter your password.
3. The TKGI CLI generates a kubeconfig for the cluster you have access to. Review the following example kubeconfig file:

```
apiVersion: v1
clusters:
- cluster:
 certificate-authority-data: PROVIDED-BY-ADMIN
 server: PROVIDED-BY-ADMIN
 name: PROVIDED-BY-ADMIN
contexts:
- context:
 cluster: PROVIDED-BY-ADMIN
 user: PROVIDED-BY-USER
 name: PROVIDED-BY-ADMIN
current-context: PROVIDED-BY-ADMIN
kind: Config
preferences: {}
users:
- name: PROVIDED-BY-USER
 user:
 auth-provider:
 config:
 client-id: pks_cluster_client
 cluster_client_secret: ""
 id-token: PROVIDED-BY-USER
 idp-issuer-url: https://PROVIDED-BY-ADMIN:8443/oauth/token
 refresh-token: PROVIDED-BY-USER
 name: oidc
```

4. Access the cluster using kubectl. For more information about kubectl commands, see [Overview of kubectl](#) in the Kubernetes documentation.

## Grant Cluster Access to a Group

Cluster admins can grant access to an identity provider group by creating a `ClusterRoleBinding` or

`RoleBinding` for that group. You can grant access to an identity provider group only if you use a LDAP or SAML identity provider for UAA. You can configure a LDAP or SAML identity provider in **Ops Manager Installation Dashboard > Tanzu Kubernetes Grid Integrated Edition > Settings > UAA**.



**Note:** If you are using a LDAP group, you must confirm that the LDAP group you are giving access is in the allowlist in the Tanzu Kubernetes Grid Integrated Edition tile. To do this, review **External Groups Whitelist** in **Ops Manager Installation Dashboard > Tanzu Kubernetes Grid Integrated Edition > Settings > UAA**.

To grant cluster access to an identity provider group, do the procedure in [Grant Cluster Access to a User](#) above and replace step 6 with the following:

1. In the YAML file for a `ClusterRoleBinding` or a `RoleBinding`, replace the `subjects` section with the following:

```
subjects:
- kind: Group
 name: NAME-OF-GROUP
 apiGroup: rbac.authorization.k8s.io
```

Use the following example as a template:

```
kind: ROLE-BINDING-TYPE
apiVersion: rbac.authorization.k8s.io/v1
metadata:
 name: ROLE-OR-CLUSTER-ROLE-BINDING-NAME
 namespace: NAMESPACE
subjects:
- kind: Group
 name: NAME-OF-GROUP
 apiGroup: rbac.authorization.k8s.io
roleRef:
 kind: ROLE-TYPE
 name: ROLE-OR-CLUSTER-ROLE-NAME
 apiGroup: rbac.authorization.k8s.io
```

Where:

- `ROLE-BINDING-TYPE` is the type of role binding you are creating. This must be either `RoleBinding` or `ClusterRoleBinding`.
- `ROLE-OR-CLUSTER-ROLE-BINDING-NAME` is the name of the role binding. This is given by the cluster admin.
- `NAMESPACE` is the namespace within the cluster. This is omitted when creating a `ClusterRole`.
- `NAME-OF-GROUP` is the identity provider group name. This name is case sensitive.



**Note:** If you configured an OIDC groups prefix in **Ops Manager Installation Dashboard > Tanzu Kubernetes Grid Integrated Edition > Settings > UAA** or in a Kubernetes profile, you must prepend

**NAME-OF-GROUP** with the prefix you configured. For more information, see [UAA](#) in the *Installing* topic for your IaaS and [Adding an OIDC Provider](#).

- **ROLE-TYPE** is the type of role you created in the previous step. This must be either [Role](#) or [ClusterRole](#).
- **ROLE-OR-CLUSTER-ROLE-NAME** is the name of the [Role](#) or [ClusterRole](#) you are binding the Group to.

## Authenticate Windows Clusters with Active Directory

This topic describes how to integrate Microsoft Active Directory (AD) with your Tanzu Kubernetes Grid Integrated Edition (TKGI) Windows worker-based Kubernetes clusters.

### Overview

Windows Server with Active Directory can control access to Windows worker-based Kubernetes clusters in TKGI. To enable this, you integrate a group Managed Service Account (gMSA) in AD with the cluster's Windows pods and containers.



**Note:** Once a cluster has been created or updated to use AD authentication, you cannot update it to stop using AD authentication.

For information about gMSAs see [Group Managed Service Accounts Overview](#) in the Microsoft Windows Server documentation.

### Prerequisites

To use AD to control access to Windows worker-based Kubernetes clusters, you need an AD server configured with:

- A gMSA account
- A security group that includes the gMSA account

### GMSA Configuration Settings

The [tkgi](#) CLI uses a JSON-formatted cluster configuration file to add the cluster's Windows nodes into the AD domain server and security groups, giving them access to the gMSA. You pass the file's location to the CLI's [--config-file](#) flag.



**Note:** Cluster configuration files can also include settings for non-gMSA features, such as proxies. You combine all such settings into a single, general-purpose configuration file to pass to the [-config-file](#) flag.

gMSA settings in the cluster configuration file are:

- **enable\_gmsa:** Boolean. [true](#) to configure gMSA for a cluster. [false](#) to ignore all settings

below.

- `domain_controller_ip_address`: Address of the AD server with the gMSA account.
- `domain_fqdn`: FQDN to add the cluster nodes to, created in AD.
- `domain_user_username`: Username for an AD account with permission to add nodes to a new domain.
- `domain_user_password`: Password the AD account.
- `domain_security_group`: AD security group that the gMSA account is included under.
- `domain_service_account`: AD gMSA account.

For example:

```
{
 "enable_gmsa": true,
 "domain_controller_ip_address": "10.199.17.52",
 "domain_fqdn": "tkgi-ad.local",
 "domain_user_username": "tkgi-admin",
 "domain_user_password": "Passw0rd",
 "domain_security_group": "WebApp01Hosts",
 "domain_service_account": "WebApp01"
}
```

## Create and Integrate a Cluster with AD Authentication

To create a TKGI cluster configured with AD gMSA authentication:

1. Define the gMSA settings in a configuration file on your local filesystem, as described in [GMSA Configuration Settings](#), above.
2. Pass the file location to the `--config-file` flag of `tkgi create-cluster`. See [Creating Clusters](#) for more information.
3. Integrate the cluster with the AD gMSA as described in [Integrate the Cluster with Active Directory](#), below.

## Change a Cluster's Active Directory Authentication

To configure an existing TKGI cluster for Active Directory gMSA authentication, or to change its gMSA configuration:



**Note:** Once a cluster has been created or updated to use AD authentication, you cannot update it to stop using AD authentication.

1. Define the gMSA settings in a configuration file on your local filesystem, as described in [GMSA Configuration Settings](#), above.
  - To retain a previous setting, do not include it in the configuration file.
  - To unset a previous setting, set it to `{}` (for an object) or `""` (for a string) in the configuration file.

2. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
3. Run the following command to update the cluster with the configuration file:

```
tkgi update-cluster CLUSTER-NAME --config-file CONFIG-FILE-NAME
```

Where:

- ◊ `CLUSTER-NAME` is the name of the existing Kubernetes cluster.
- ◊ `CONFIG-FILE-NAME` is the path and filename of the configuration file you want to apply to the cluster.

4. Integrate the cluster with the AD gMSA as described in [Integrate the Cluster with Active Directory](#), below.

## Integrate a Cluster with Active Directory

Once you have a cluster configured in Tanzu Kubernetes Grid and AD to access the gMSA, you need to run the following Kubernetes operations to integrate the cluster with the gMSA at the Kubernetes level.

This procedure is described in [Configure GMSA for Windows Pods and containers](#) in the Kubernetes documentation, with one exception. The exception is that you used the `--config-file` flag in the `tkgi` CLI to add the Windows nodes into the AD, instead of following the step [Configure GMSAs and Windows nodes in Active Directory](#) in the Kubernetes documentation.

1. [Install the GMSACredentialSpec CRD](#).
2. To prepare for the next step, download the example script `deploy-gmsa-webhook.sh` and update its `CA_BUNDLE` setting:
  1. `bosh ssh` into the cluster's control plane node. For more information, see [SSH into a Kubernetes Cluster VM](#).
  2. Fetch the `/var/vcap/jobs/kube-controller-manager/config/cluster-signing-ca.pem` file and name it locally as `ca-master.pem`.
  3. Edit `CA_BUNDLE` setting in the `deploy-gmsa-webhook.sh` example script linked above to read as follows:

```
CA_BUNDLE="$(cat ca-master.pem|base64 -w 0)"
```

3. Run the script to install two webhooks into the cluster that populate and validate gMSA credential spec references at the Pod or container level. For more information, see [Install webhooks to validate GMSA users](#).
4. Log in to Active Directory and create gMSA credential spec resources using the `domain_service_account` and the domain information in your configuration file. For more information, see [Create GMSA credential spec resources](#).

This step includes converting your configuration file from JSON to YAML.

5. Configure cluster role to enable RBAC on specific GMSA credential specs for the credential

specs created in the previous step.

6. Assign role to service accounts to use specific GMSA `credspecs` for the role created in the previous step.
7. Configure GMSA credential spec reference in Pod spec for the Pod spec created earlier.

## View a Cluster's gMSA Configuration

You can see a cluster's current gMSA configuration by viewing its BOSH manifest:

1. Identify the names of your cluster deployments:

```
bosh deployments
```



**Note:** Cluster deployment names start with `service-instance_`.

2. For any cluster you want to view, download its manifest:

```
bosh -d DEPLOYMENT-NAME manifest > /tmp/YOUR-DEPLOYMENT-MANIFEST.yml
```

Where:

- ◊ `DEPLOYMENT-NAME` is the name of your Kubernetes cluster deployment.
- ◊ `YOUR-DEPLOYMENT-MANIFEST` is the name of your Kubernetes cluster deployment manifest.

3. Search the manifest for the **GMSA configuration settings** above.

## Configure Cluster Proxies

This topic describes how customize HTTP/HTTPS proxies for individual Tanzu Kubernetes Grid Integrated Edition (TKGI) clusters.

### Overview

TKGI applies your HTTP/HTTPS cluster proxies to traffic from the cluster's Kubernetes and containerd processes, such as the Kubernetes API server, Kube Controller, Kubelet, and containerd daemon.

To create or change a cluster's proxy configuration, see:

- [Create a Cluster with a Custom Proxy Configuration](#)
- [Change a Cluster's Proxy Configuration](#)

To view a cluster's proxy configuration, see:

- [View a Cluster's Proxy Configuration](#)

To configure global HTTP/HTTPS proxies for TKGI on vSphere or AWS, see:

- **vSphere:** [Using Proxies with Tanzu Kubernetes Grid Integrated Edition on NSX-T](#), or
- **AWS:** [Using Proxies with Tanzu Kubernetes Grid Integrated Edition on AWS](#)

These two topics also cover how the proxies work, and how they can be useful.

## Create a Cluster with a Custom Proxy Configuration

To create a cluster with a custom proxy configuration:

1. Define the proxy settings in a configuration file, as described in [Proxy Configuration Settings](#), below.
2. Pass the file location to the `--config-file` flag of `tkgi create-cluster`. See [Creating Clusters](#) for more information.

## Change a Cluster's Proxy Configuration

To change a cluster's proxy configuration:

1. Define the proxy settings in a configuration file, as described in [Proxy Configuration Settings](#), below.
  - ◊ To retain a previous setting, do not include it in the configuration file.
  - ◊ To unset a previous setting, set it to `{}` (for an object) or `""` (for a string) in the configuration file.
2. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
3. Run the following command to update the cluster with the configuration file:

```
tkgi update-cluster CLUSTER-NAME --config-file CONFIG-FILE-NAME
```

Where:

- ◊ `CLUSTER-NAME` is the name of the existing Kubernetes cluster.
- ◊ `CONFIG-FILE-NAME` is the path and filename of the configuration file you want to apply to the cluster.



**Note:** When you use `tkgi update-cluster` to update an existing cluster, the attached network-profile must consist of only updatable settings.

## Proxy Configuration Settings

To configure HTTP/HTTPS settings for a TKGI cluster, you first define them in a cluster configuration JSON file on your local filesystem.

Proxy settings that you can configure are:

| Setting | Description |
|---------|-------------|
|---------|-------------|

|                                    |                                                                                                                                                                                                                                         |
|------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>http_proxy</code>            | HTTP proxy URL and credentials. This overrides the global <b>HTTP Proxy</b> settings in the TKGI tile > <b>Networking</b> pane.                                                                                                         |
| <code>https_proxy</code>           | HTTPS proxy URL and credentials. This overrides the global <b>HTTP Proxy</b> settings in the TKGI tile.                                                                                                                                 |
| <code>no_proxy</code>              | Comma-separated list of IP addresses that bypass the proxy for internal communication. This interacts with the tile's global <b>No Proxy</b> setting based on the <code>global_no_proxy_merge</code> setting, below.                    |
| <code>global_no_proxy_merge</code> | Boolean value. The default <code>false</code> setting merges the <code>no_proxy</code> setting above with the global <b>No Proxy</b> list set in the tile. Setting this to <code>true</code> overrides the global <b>No Proxy</b> list. |

For example, the following configuration file overrides the `http_proxy` and `https_proxy` settings in the tile, and merges the `no_proxy` list here with the `no_proxy` list set in the tile:

```
{
 "http_proxy": {
 "url": "http://example.com",
 "username": "admin",
 "password": "admin"
 },
 "https_proxy": {
 "url": "http://example.com",
 "username": "admin",
 "password": "admin"
 },
 "no_proxy": "127.0.0.1,localhost,*.example.com,198.51.100.0/24",
 "global_no_proxy_merge": true
}
```



**Note:** Cluster configuration files can also include settings for non-proxy features, such as enabling cluster access by group Managed Service Accounts (gMSAs). You combine all such settings into a single, general-purpose configuration file to pass to the `-config-file` flag.

## View a Cluster’s Proxy Configuration

You can see a cluster’s current proxy configuration by viewing its BOSH manifest:

- To identify the names of your cluster deployments:

```
bosh deployments
```



**Note:** Cluster deployment names start with `service-instance_`.

- To download the manifest for any cluster you want to view:

```
bosh -d DEPLOYMENT-NAME manifest > /tmp/YOUR-DEPLOYMENT-MANIFEST.yml
```

Where:

- ◊ `DEPLOYMENT-NAME` is the name of your Kubernetes cluster deployment.
- ◊ `YOUR-DEPLOYMENT-MANIFEST` is the name of your Kubernetes cluster deployment manifest.

3. Search the manifest for `proxy` to see its proxy settings under `jobs.properties.env`, for example:

```
jobs:
 - name: containerd
 release: kubo
 properties:
 bridge: cni0
 default_ulimits:
 - nofile=1048576
 env:
 http_proxy: ""
 https_proxy: ""
 no_proxy: .internal,.svc,.svc.cluster.local,.svc.cluster,api.pks.local,
10.100.200.0/24,10.200.0.0/16,88.0.0.0/24,192.168.111.0/24,192.168.139.1,192.16
8.160.0/24,nsxmanager.pks.vmware.local
 flannel: false
 ip_masq: false
 iptables: false
 live_restore: true
 log_level: error
 log_options:
 - max-size=128m
 - max-file=2
 storage_driver: overlay2
 store_dir: /var/vcap/store
```

## Getting Started with VMware Harbor Registry

This topic describes VMware Harbor Registry, an enterprise-class image registry server that stores and distributes container images for VMware Tanzu Kubernetes Grid Integrated Edition.

### Overview

Harbor allows you to store and manage container images for your Tanzu Kubernetes Grid Integrated Edition deployment. Deploying an image registry alongside Tanzu Kubernetes Grid Integrated Edition improves image transfer speed.

As an enterprise private registry, Harbor also offers enhanced performance and improved security. By configuring Harbor with Tanzu Kubernetes Grid Integrated Edition, you can apply enterprise features to your image registry, such as security, identity, and management.

You can install Harbor alongside Tanzu Kubernetes Grid Integrated Edition on vSphere, Amazon Web Services (AWS), Google Cloud Platform (GCP), and Microsoft Azure.

### Install Harbor

To install Harbor, do the following:

1. Install Tanzu Kubernetes Grid Integrated Edition. For more information, see the *Installing Tanzu Kubernetes Grid Integrated Edition* topic for your cloud provider.
2. Install Harbor. For more information, see [Installing and Configuring VMware Harbor Registry](#).

## Use Harbor

Before you can push images to Harbor, you must do the following:

1. Configure authentication and role-based access control (RBAC) for Harbor. For more information, see [Create Projects](#) in the Harbor documentation.
2. Create a Harbor project that contains all repositories for your app. For more information, see [Create Projects](#) in the Harbor documentation.

After you configure Harbor, you can do the following:

- Push or pull Docker images to your Harbor project using the Docker command-line interface (CLI). For more information, see [Pulling and Pushing Images in the Docker Client] (<https://goharbor.io/docs/1.10/working-with-projects/working-with-images/pulling-pushing-images/>) in the Harbor documentation.
- Manage Helm charts in your Harbor project using either the Harbor portal or the Helm CLI. For more information, see [Managing Helm Charts] (<https://goharbor.io/docs/1.10/working-with-projects/working-with-images/managing-helm-charts/>) in the Harbor documentation.
- Install Clair to activate vulnerability scanning for images stored in Harbor. For more information, see [Step 8: Configure Container Vulnerability Scanning Using Clair] ([https://docs.vmware.com/en/VMware-Harbor-Registry/services/vmware-harbor-registry/GUID-installing.html#configure\\_clair](https://docs.vmware.com/en/VMware-Harbor-Registry/services/vmware-harbor-registry/GUID-installing.html#configure_clair)) in *Installing and Configuring VMware Harbor Registry*.

For more information about managing images in Harbor, see the [Working with Images, Tags, and Helm Charts] (<https://goharbor.io/docs/1.10/working-with-projects/working-with-images/>) in the Harbor documentation.

## Manage Harbor

As a Harbor administrator, you can manage the following in the Harbor portal:

- **Authentication:** Select either local user authentication or configure LDAP/Active Directory integration. If you select local user authentication, you can activate or deactivate user self-registration.
- **Users and roles:** Manage privileges for Harbor users.
- **Email settings:** Configure a mail server for user password resets.
- **Project creation:** Specify which users can create projects.
- **Registry permissions:** Manage permissions for image registry access.
- **Endpoints:** Add and remove image registry endpoints.
- **Replication policies:** Add and remove rules for replication jobs.

For more information about managing Harbor as an administrator, see [Harbor Administration](#) in the

Harbor documentation.

## Configuring Cluster Access to Private Docker Registries (Beta)

This topic describes how to configure VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) Kubernetes clusters with private registry SSL Certificate Authority (CA) certificates.



**Warning:** Configuring TKGI clusters with private Docker registry CA certificates is currently in beta and is intended for evaluation and test purposes only. Do not use this feature in a TKGI production environment.

### Overview

You can store images in a private Docker registry and secure the registry with SSL CA certificates:

- You can enable your TKGI Kubernetes clusters to authenticate into a private Docker registry by configuring your clusters with SSL CA certificates.
- You can configure both new and existing TKGI clusters to have Docker registry CA certificates.



**Note:** Only Linux clusters can be configured to have Docker registry CA certificates.

To create a new cluster configured with Docker registry SSL CA certificates, complete the following procedures:

1. Set up Your API Access Token
2. Create a Cluster with SSL CA Certificates

To update an existing cluster with Docker registry SSL CA certificates, complete the following procedures:

1. Set up Your API Access Token
2. Update a Cluster with SSL CA Certificates



**Note:** The procedures documented in this topic configure an individual TKGI Kubernetes cluster with a Docker Registry SSL CA certificate. See [Import the CA Certificate Used to Sign the Harbor Certificate and Key to BOSH](#) in *Integrating VMware Harbor Registry with Tanzu Kubernetes Grid Integrated Edition* if you want to apply a single Harbor Registry certificate to all of your TKGI clusters.

### Prerequisites

Before configuring TKGI Kubernetes clusters to have Docker registry CA certificates, you must have the following:

- A private Docker registry configured to use SSL CA certificates. For more information about

securing a private Docker registry, see [Use self-signed certificates in the Docker Registry](#) manual.



**Warning:** The FQDN for the private Docker registry cannot contain a hyphen, dash, or semi-colon. If such a character is included in the registry name the TKGI API will reject it as not a valid character.

## Set up Your API Access Token

The curl commands in this topic use an access token environment variable to authenticate to the TKGI API endpoints.

1. To export your access token into an environment variable, run the following command:

```
tkgi login -a TKGI-API -u USER-ID -p 'PASSWORD' -k; \
export YOUR-ACCESS-TOKEN=$(bosh int ~/.pks/creds.yml --path /access_token)
```

Where:

- **TKGI-API** is the FQDN of your TKGI API endpoint. For example, `api.tkgi.example.com`.
- **USER-ID** is your Tanzu Kubernetes Grid Integrated Edition user ID.
- **PASSWORD** is your Tanzu Kubernetes Grid Integrated Edition password.
- **YOUR-ACCESS-TOKEN** is the name of your access token environment variable.

For example:

```
$ tkgi login -a tkgi.my.lab -u alana -p 'psswrdaabc123...!' -k;
export my_token=$(bosh int ~/.pks/creds.yml -path /access_token)
```



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

## Create a Cluster with SSL CA Certificates

You can create a new cluster configured with one or more SSL CA certificates by using the TKGI API `create-cluster` endpoint.

1. To create a cluster configured with one or more SSL CA certificates, run the following command:

```
curl -X POST \
https://TKGI-API:9021/v1/clusters \
```

```
-H 'Accept: application/json' \
-H "Authorization: Bearer $YOUR-ACCESS-TOKEN" \
-H 'Content-Type: application/json' \
-H 'Host: TKGI-API:9021' \
-d '{
 "name": "CLUSTER-NAME",
 "plan_name": "PLAN-NAME",
 "parameters": {
 "kubernetes_master_host": "KUBERNETES-CONTROLPLANE-HOST",
 "custom_ca_certs": [
 {
 "domain_name": "DOMAIN-NAME",
 "ca_cert": "CA-CERTIFICATE"
 }
]
 }
}'
```

Where:

- ◊ `TKGI-API` is the FQDN of your TKGI API endpoint. For example, `api.tkgi.example.com`.
- ◊ `YOUR-ACCESS-TOKEN` is the name of your access token environment variable.
- ◊ `CLUSTER-NAME` is the name of your cluster.



**Note:** Use only lowercase characters when naming your cluster if you manage your clusters with Tanzu Mission Control (TMC). Clusters with names that include an uppercase character cannot be attached to TMC.

- ◊ `PLAN-NAME` is the name of your plan.
- ◊ `KUBERNETES-CONTROLPLANE-HOST` is your Kubernetes control plane host.
- ◊ `DOMAIN-NAME` is a Docker Registry URL. You cannot remove an existing Docker Registry URL from a cluster. If you specify a URL that is already registered with your cluster, the cluster's existing CA certificate for that URL is overwritten.
- ◊ `CA-CERTIFICATE` is the CA certificate that corresponds to `DOMAIN-NAME`. For more information about using a CA certificate in a TKGI API command, see [Prepare a Certificate String for Command Line Use](#), below.

You can configure your cluster with additional certificates by including the certificates in the `custom_ca_certs` array as additional `domain_name`, `ca_cert` pairs.



**Note:** You can include wildcard characters in your `domain_name` URLs. For example, `*.docker.com`.

## Update a Cluster with SSL CA Certificates

You can update an existing cluster with one or more SSL CA certificates by using the TKGI API

`update-cluster` endpoint.

- To configure an existing cluster with one or more SSL CA certificates, run the following command:

```
curl -X PATCH \
 https://TKGI-API:9021/v1/clusters/CLUSTER-NAME \
 -H 'Accept: application/json' \
 -H "Authorization: Bearer $YOUR-ACCESS-TOKEN" \
 -H 'Content-Type: application/json' \
 -H 'Host: TKGI-API:9021' \
 -d '{
 "custom_ca_certs": [
 {
 "domain_name": "DOMAIN-NAME",
 "ca_cert": "CA-CERTIFICATE"
 }
]
}'
```

Where:

- `TKGI-API` is the FQDN of your TKGI API endpoint. For example, `api.tkgi.example.com`.
- `YOUR-ACCESS-TOKEN` is the name of your access token environment variable.
- `CLUSTER-NAME` is the name of your cluster.
- `DOMAIN-NAME` is a Docker Registry URL. You cannot remove an existing Docker Registry URL from a cluster. If you specify a URL that is already registered with your cluster, the cluster's existing CA certificate for that URL is overwritten.
- `CA-CERTIFICATE` is the CA certificate that corresponds to `DOMAIN-NAME`. For more information about using a CA certificate in a TKGI API command, see [Prepare a Certificate String for Command Line Use](#), below.

You can configure your cluster with additional certificates by including the certificates in the `custom_ca_certs` array as additional `domain_name`, `ca_cert` pairs.



**Note:** You can include wildcard characters in your `domain_name` URLs. For example, `*.docker.com`.

## SSL CA Certificate Formats

SSL CA certificates are unique CA-issued ASCII text strings.

The CAs issue most certificates as a PEM formatted ASCII text files. PEM certificate files typically have the extensions `.pem`, `.crt`, `.cer`, or `.key`.

PEM files start with the string `-----BEGIN CERTIFICATE-----`, terminate with `-----END CERTIFICATE-----`, and are Base64-encoded. Certificate strings are long and are frequently stored within a certificate file with newline wrapping every 64 characters.

## Prepare a Certificate String for Command Line Use

When you provide a certificate string on a command line or TKGI API command, as in the TKGI API commands above, your certificate string must be provided without newline wrapping.



**Note:** The TKGI API does not validate certificate strings for correctness. Ensure your certificate string is free of newline characters before using the certificate string in a TKGI API command.

To prepare your certificate string for command line use:

1. To remove newline wrapping from a certificate string, run the following command:

```
awk 'NF {sub(/\r/, ""); printf "%s\\n",$0;}' CA-PEM
```

Where `CA-PEM` is the filename of your PEM-formatted CA certificate file.

This command returns your certificate string without newline wrapping.

## Deploying and Managing Cloud Native Storage (CNS) on vSphere

This topic describes how to use the vSphere Container Storage Interface (CSI) Driver that is automatically installed to clusters by VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) on vSphere.

### Overview

vSphere Cloud Native Storage (CNS) provides comprehensive data management for stateful, containerized apps, enabling apps to survive restarts and outages. Stateful containers can use vSphere storage primitives such as standard volume, persistent volume, and dynamic provisioning, independent of VM and container lifecycle.

You can install vSphere CNS on TKGI-provisioned clusters by configuring TKGI to automatically install a vSphere CSI Driver. To enable automatic CSI driver installation on your clusters, see [Storage](#) in *Installing TKGI on vSphere*.

When automatic vSphere CSI Driver installation is enabled, your clusters use your tile **Kubernetes Cloud Provider** storage settings as the default vSphere CNS configuration.

The automatically deployed vSphere CSI Driver supports high availability (HA) configurations. HA support is automatically enabled on clusters with multiple control plane nodes and uses only one active CSI Controller.

Use the vSphere client to review your cluster storage volumes and their backing virtual disks, and to set a storage policy on your storage volumes or monitor policy compliance. vSphere storage backs up your cluster volumes.

For more information about VMware CNS, see [Getting Started with VMware Cloud Native Storage](#).

For more information about using the Kubernetes CSI Driver, see [Persistent Volumes](#) in the Kubernetes documentation.

In TKGI, you can configure the vSphere CSI Driver to:

- Customize, deploy and manage vSphere CNS volumes:
  - To customize file volumes, see [Customize vSphere File Volumes](#) below.
  - To use and customize CNS Volumes, see [Create or Use CNS Block Volumes](#) below.
  - To migrate in-tree storage volumes to the vSphere CSI Driver, see [Migrate an In-Tree vSphere Storage Volume to the vSphere CSI Driver](#) below.
- Customize clusters to support vSphere Topology-Aware Volume Provisioning:
  - To apply vSphere Topology-Aware Volume Provisioning to your cluster, see [Customize a Cluster with vSphere Topology-Aware Volume Provisioning](#) below.
- Customize and manage vSphere CNS:
  - To use CNS in a multi-data center environment, see [Configure CNS Data Centers](#) below.
- Customize the maximum number of snapshots for a volume
  - To customize the maximum number of snapshots for each persistent volume, see [Customize the Maximum Number of Volume Snapshots](#) below.

## Requirements and Limitations of the vSphere CSI Driver

For information about the supported features and the known limitations of the vSphere CSI Driver, see:

- [vSphere CSI Driver Supported Features and Requirements](#)
- [Unsupported Features and Limitations](#)

## vSphere CSI Driver Supported Features and Requirements

The vSphere CSI Driver supports different features depending on driver version, environment and storage type.

TKGI supports only the following vSphere CSI Driver features:

- Dynamic Block PV support\*
- Dynamic File PV support\*
- Dynamic Virtual Volume (vVOL) PV support
- Encryption support via VMcrypt\*
- Enhanced Object Health in UI for vSAN Datastores

- Kubernetes Multi-node Control Plane support
- Static PV Provisioning
- Topology-aware volume provisioning
- Volume snapshot and restore

\* For information on the usage limitations and environment and version requirements of these vSphere CSI Driver features, see [Supported Kubernetes Functionality in Compatibility Matrices for vSphere Container Storage Plug-in](#) in the VMware vSphere Container Storage Plug-in documentation.

For information on the vCenter, datastore, and cluster types supported by the vSphere CSI Driver, see [vSphere Functionality Supported by vSphere Container Storage Plug-in](#) in the VMware vSphere Container Storage Plug-in documentation.

For information on the scaling limitations of the vSphere CSI Driver, see [Configuration Maximums for vSphere Container Storage Plug-in](#) in the VMware vSphere Container Storage Plug-in documentation.

## Unsupported Features and Limitations

vSphere Storage DRS, Manual Storage vMotion, and other VMware vSphere features are not supported by the vSphere Container Storage Plug-in and cannot be used by the TKGI clusters that use or migrate to the vSphere CSI Driver.

For more information on the limitations of the VMware vSphere Container Storage Plug-in, see [vSphere Functionality Supported by vSphere Container Storage Plug-in](#) in the VMware vSphere Container Storage Plug-in documentation.

## Customize vSphere File Volumes

To create, modify or remove a customized vSphere file volume:

- [Create a Cluster with Customized File Volume Parameters](#)
- [Modify a Cluster with Customized File Volume Parameters](#)
- [Remove File Volume Parameters from a Cluster](#)

## Prerequisites

To use file volumes, you must enable vSAN File Services in the vSphere vCenter. For information about enabling vSAN File Services, see [Configure File Services](#) in the VMware vSphere documentation.

## Create a Cluster with Customized File Volume Parameters

To create a new cluster with a vSphere file volume:

1. Create a JSON or YAML formatted volume configuration file containing the following:

```
{
 "target_vsan_fileshare_datastore_urls": "DS-URLS",
 "net_permissions": [
 {
 "name": "PERMISSION-NAME",
 "ips": "IP-ADDRESS",
 "permissions": "PERMISSION",
 "rootsquash": "ACCESS-LEVEL"
 },
 {
 "name": "PERMISSION-NAME",
 "ips": "IP-ADDRESS",
 "permissions": "PERMISSION",
 "rootsquash": "ACCESS-LEVEL"
 }
]
}
```

Where:

- ◆ **DS-URLS** is a comma-separated list of datastores for deploying file share volumes. For example: "ds:///vmfs/volumes/vsan:52635b9067079319-95a7473222c4c9cd/".
- ◆ **PERMISSION-NAME** is your name for a NetPermission.
- ◆ **IP-ADDRESS** is the IP range or IP subnet affected by a NetPermission restriction.
- ◆ **PERMISSION** is the access permission to the file share volume for a NetPermission restriction.
- ◆ **ACCESS-LEVEL** is the security access level for the file share volume for a NetPermission restriction.

For information, see [File Volume Configuration](#) below.

2. To create a cluster with attached file volumes:

```
tkgi create-cluster CLUSTER-NAME --config-file CONFIG-FILE
```

Where:

- ◆ **CLUSTER-NAME** is the name of your cluster.
- ◆ **CONFIG-FILE** is the name of your config file.

For example:

```
$ tkgi create-cluster demo -e demo.cluster -plan Small -config-file ./conf1.json
```

## Modify a Cluster with Customized File Volume Parameters

To modify an existing cluster with a vSphere file volume:

1. Create a file volume configuration file. For information, see [File Volume Configuration](#) below.
2. To update your cluster with file volumes:

```
tkgi update-cluster CLUSTER-NAME --config-file CONFIG-FILE
```

Where:

- ◊ **CLUSTER-NAME** is the name of your cluster.
- ◊ **CONFIG-FILE** is the name of your config file.

## Remove File Volume Parameters from a Cluster

To remove a vSphere file volume configuration from a cluster:

1. Create a file volume configuration file containing either the `disable_target_vsan_fileshare_datastore_urls` or `disable_net_permissions` parameters set to `true` to deactivate an existing file volume parameter.

For more information, see [File Volume Configuration](#) below.

2. To remove the configured file volume parameter from your cluster:

```
tkgi update-cluster CLUSTER-NAME --config-file CONFIG-FILE
```

Where:

- ◊ **CLUSTER-NAME** is the name of your cluster.
- ◊ **CONFIG-FILE** is the name of your config file.

## File Volume Configuration

Create a JSON or YAML formatted File Volume configuration file to enable or deactivate vSphere file volume support.

For example:

- The following configuration enables all File Volume features:

```
{
 "target_vsan_fileshare_datastore_urls": "ds:///vmfs/volumes/vsan:52635b906707
9319-95a7473222c4c9cd/",
 "net_permissions": [
 {
 "name": "demo1",
 "ips": "192.168.0.0/16",
 "permissions": "READ_WRITE",
 "rootsquash": false
 },
 {
 "name": "demo2",
 "ips": "10.0.0.0/8",
 "permissions": "READ_WRITE",
 "rootsquash": false
 }
]
}
```

```

 "permissions": "READ_ONLY",
 "rootsquash": false
 }
]
}

```

- The following configuration deactivates File Volume features:

```

{
 "disable_target_vsan_fileshare_datastore_urls": true,
 "disable_net_permissions": true
}

```

## File Volume DataStores Configuration

The following are accepted Datastore URLs parameters:

| Name                                         | Type    | Description                                                                                                                                     |
|----------------------------------------------|---------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| target_vsan_fileshare_datastore_urls         | string  | A comma separated list of datastores for deploying file share volumes.                                                                          |
| disable_target_vsan_fileshare_datastore_urls | Boolean | Deactivate the target_vsan_fileshare_datastore_urls.<br>Values: <code>true</code> , <code>false</code> .<br>Default Value: <code>false</code> . |

## File Volume NetPermissions Object Configuration

The following are accepted NetPermissions objects:

| Name                    | Type    | Description                                                                                                                |
|-------------------------|---------|----------------------------------------------------------------------------------------------------------------------------|
| net_permissions         | Array   | Properties defining a NetPermissions object.                                                                               |
| disable_net_permissions | Boolean | Deactivate the net_permissions.<br>Values: <code>true</code> , <code>false</code> .<br>Default Value: <code>false</code> . |

The following are supported NetPermissions object parameters:

| Name        | Type    | Description                                                                                                                                                                            |
|-------------|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| name        | string  | Name of the NetPermission object.                                                                                                                                                      |
| ips         | string  | IP range or IP subnet affected by the NetPermission restrictions.<br>Default Value: <code>"*"</code> .                                                                                 |
| permissions | string  | Access permission to the file share volume.<br>Values: <code>"READ_WRITE"</code> , <code>"READ_ONLY"</code> , <code>"NO_ACCESS"</code> .<br>Default Value: <code>"READ_WRITE"</code> . |
| rootsquash  | Boolean | Security access level for the file share volume.<br>Values: <code>true</code> , <code>false</code> .<br>Default Value: <code>false</code> .                                            |

For more information on NetPermissions object parameters, see [Procedure in Create a Kubernetes Secret for vSphere Container Storage Plug-in](#).

## Create or Use CNS Block Volumes

To dynamically provision a block volume using the vSphere CSI Driver:

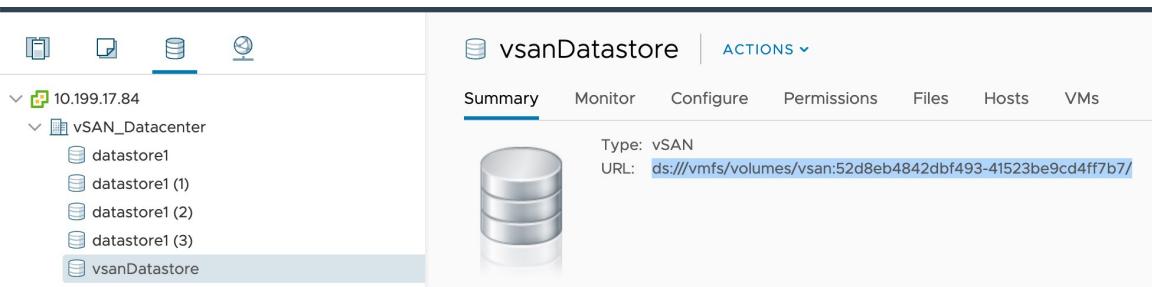
1. Create a vSphere Storage Class
2. Create a PersistentVolumeClaim
3. Create Workloads Using Persistent Volumes

For more information on vSphere CSI Driver configuration, see the [example/vanilla-k8s-block-driver](#) configuration for the CSI driver version you are using in [vsphere-csi-driver](#) in the VMware kubernetes-sigs GitHub repo.

## Create a vSphere Storage Class

To create a vSphere Storage Class:

1. Open vCenter.
2. Open the vSAN Datastore Summary pane.



3. Determine the `datastoreurl` value for your Datastore.
4. Create the following YAML:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
 name: demo-sts-storageclass
 annotations:
 storageclass.kubernetes.io/is-default-class: "true"
provisioner: csi.vsphere.vmware.com
allowVolumeExpansion: ALLOW-EXPANSION
parameters:
 datastoreurl: "DATASTORE-URL"
```

Where:

- `ALLOW-EXPANSION` defines whether the cluster's persistent volume size is either resizable or static. Set to `true` for resizable and `false` for static size.
- `DATASTORE-URL` is the URL to your Datastore. For a non-vSAN datastore, the `datastoreurl` value looks like `ds:///vmfs/volumes/5e66e525-8e46bd39-c184-`

```
005056ae28de/.
```

For example:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
 name: demo-sts-storageclass
 annotations:
 storageclass.kubernetes.io/is-default-class: "true"
provisioner: csi.vsphere.vmware.com
allowVolumeExpansion: true
parameters:
 datastoreurl: "ds:///vmfs/volumes/vsan:52d8eb4842dbf493-41523be9cd4ff7b7/"
```

For more information about StorageClass, see [Storage Classes](#) in the Kubernetes documentation.

## Create a PersistentVolumeClaim

To create a Persistent Volume using the vSphere CSI Driver:

1. Create a Storage Class. For more information, see [Create a vSphere Storage Class](#) below.
2. To apply the StorageClass configuration:

```
kubectl apply -f CONFIG-FILE
```

Where `CONFIG-FILE` is the name of your StorageClass configuration file.

3. Create the PersistentVolumeClaim configuration for the file volume. For information about configuring a PVC, see [Persistent Volumes](#) in the Kubernetes documentation.

For example:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
 name: example-vanilla-block-pvc
spec:
 accessModes:
 - ReadWriteOnce
 resources:
 requests:
 storage: 5Gi
 storageClassName: example-vanilla-block-sc
```

4. To apply the PVC configuration:

```
kubectl apply -f CONFIG-FILE
```

Where `CONFIG-FILE` is the name of your PVC configuration file.

## Create Workloads Using Persistent Volumes

1. Create a Pod configuration file containing `volumeMounts` and `volumes` parameters.

For example:

```
apiVersion: v1
kind: Pod
metadata:
 name: example-vanilla-block-pod
spec:
 containers:
 - name: test-container
 image: gcr.io/google_containers/busybox:1.24
 command: ["/bin/sh", "-c", "echo 'hello' > /mnt/volume1/index.html && chmod o+rX /mnt /mnt/volume1/index.html && while true ; do sleep 2 ; done"]
 volumeMounts:
 - name: test-volume
 mountPath: /mnt/volume1
 restartPolicy: Never
 volumes:
 - name: test-volume
 persistentVolumeClaim:
 claimName: example-vanilla-block-pvc
```

2. To apply the Pod configuration to your workload:

```
kubectl apply -f CONFIG-FILE
```

Where `CONFIG-FILE` is the name of your configuration file.

For more information and examples of Pod configurations, see the `example` configurations for the CSI driver version you are using in [vsphere-csi-driver](#) in the VMware kubernetes-sigs GitHub repo.

## Customize a Cluster with vSphere Topology-Aware Volume Provisioning

TKGI supports the vSphere Container Storage Plug-in's topology-aware volume provisioning features.

For more information on volume provisioning features, see [Allowed Topologies](#) in the Kubernetes documentation and [Topology-Aware Volume Provisioning](#) in the VMware vSphere Container Storage Plug-in documentation.

## Topology Overview

TKGI supports clusters with topology-aware volume provisioning.

To create a cluster with topology-aware volume provisioning:

1. Prepare for Topology

2. See [Guidelines and Best Practices for Deployment with Topology](#) in *Deploying vSphere Container Storage Plug-in with Topology* in the VMware vSphere Container Storage Plug-in documentation.
3. [Create a Cluster with Topology](#)

To manage a cluster configured with topology-aware volume provisioning:

1. [Prepare for Topology](#)
2. See [Guidelines and Best Practices for Deployment with Topology](#) in *Deploying vSphere Container Storage Plug-in with Topology* in the VMware vSphere Container Storage Plug-in documentation.
3. [Manage Clusters with Topology-Aware Volumes](#)



**Note:** You cannot add topology-aware volume provisioning to an existing cluster within TKGI.

## Prepare for Topology

Before creating a new cluster with Topology-aware volume provisioning:

1. Verify your environment meets the requirements listed in [Topology Limitations and Prerequisites](#) below.
2. Review the vSphere CSI Topology deployment recommendations. For more information, see [Guidelines and Best Practices for Deployment with Topology](#) in *Deploying vSphere Container Storage Plug-in with Topology* in the VMware vSphere Container Storage Plug-in documentation.
3. Create vSphere Center categories and tags as described in [Procedures](#) in *Deploying vSphere Container Storage Plug-in with Topology* in the VMware vSphere Container Storage Plug-in documentation.

For more information on creating vSphere Center tags and categories, see [Create, Edit, or Delete a Tag Category](#) in the VMware vSphere documentation.

## Topology Limitations and Prerequisites

In TKGI you can create a new cluster with topology-aware volume provisioning enabled. You cannot add topology-aware volume provisioning to an existing cluster.

TKGI support for Topology-aware volume provisioning requires:

- The **vSphere CSI Driver Integration** option must be enabled on the TKGI tile. For more information, see [Storage](#) in installing TKGI on vSphere.
- You have created vSphere CSI topology categories and tags in your vSphere environment. For more information, see [Prepare for Topology](#) below.
- You have prepared your environment as described in the vSphere CSI Topology deployment

recommendations. For more information, see [Guidelines and Best Practices for Deployment with Topology](#) in *Deploying vSphere Container Storage Plug-in with Topology* in the VMware vSphere Container Storage Plug-in documentation.

- The topology zone tags you create on your vSphere Client must be consistent with the existing AZs created in BOSH. Create topology zone tags on your vSphere Client using only AZ names existing for BOSH.
- The topology feature does not support clusters with a Compute Profile that includes AZ settings.

## Create a Cluster with Topology

To create a new cluster with a vSphere Topology configuration:

1. Create a JSON or YAML configuration file containing the following:

```
{
 "csi_topology_labels": {
 "topology_categories": "REGION-TAG, ZONE-TAG"
 }
}
```

Where:

- ◊ **REGION-TAG** is the vSphere Center region tag you created in [Prepare for Topology](#) above.
- ◊ **ZONE-TAG** is one of the vSphere Center zone tags you created in [Prepare for Topology](#) above.

For example:

```
{
 "csi_topology_labels": {
 "topology_categories": "k8s-region, k8s-zone"
 }
}
```

For more information, see [Guidelines and Best Practices for Deployment with Topology](#) in *Deploying vSphere Container Storage Plug-in with Topology* in the VMware vSphere Container Storage Plug-in documentation.

2. To create a cluster with Topology-aware volume provisioning:

```
tkgi create-cluster CLUSTER-NAME --config-file CONFIG-FILE
```

Where:

- ◊ **CLUSTER-NAME** is the name of your cluster.
- ◊ **CONFIG-FILE** is the name of your config file.

For example:

```
$ tkgi create-cluster demo -e demo.cluster -plan Small -config-file ./conf1.json
```

## Manage Clusters with Topology-Aware Volumes

As you manage your clusters with topology-aware volume provisioning enabled, note the following limitations on existing clusters.

When running `tkgi update-cluster` on a cluster created with a topology-aware volume:

- You must use the same `csi_topology_labels` configuration that was used during cluster creation.
- You cannot add or remove topology-aware volume provisioning from the cluster.

## Migrate In-Tree vSphere Storage to the vSphere CSI Driver

Kubernetes' support for in-tree vSphere storage volumes has been deprecated, and support will be removed in a future Kubernetes version.

The TKGI v1.17 upgrade process will automatically migrate your in-tree vSphere storage volumes to vSphere CSI. If you have existing clusters that use in-tree vSphere storage volumes, you can continue to use the volumes with your current version of TKGI, but VMware strongly recommends that you migrate your in-tree vSphere storage volumes to vSphere CSI volumes as soon as possible.

To manually migrate a cluster from an in-tree vSphere storage volume to a vSphere CSI Driver volume, see [Migrate an In-Tree vSphere Storage Volume to the vSphere CSI Driver](#) below.

To prepare TKGI to use vSphere CSI Driver volumes by default and to automatically migrate clusters from in-tree vSphere storage volumes to vSphere CSI Driver volumes during TKGI upgrades, see [Prepare for Automatic Migration of Volumes from In-Tree vSphere Storage to CSI](#) below.



**Note:** You cannot migrate the PV or the PVC on a cluster from the vSphere CSI Driver to the In-Tree vSphere Storage Driver.

## Prepare for Automatic Migration of Volumes from In-Tree vSphere Storage to CSI

If your existing clusters have in-tree vSphere storage volumes, you must prepare for them to be automatically migrated before upgrading to TKGI 1.17.

To prepare for automatic migration from in-tree vSphere storage volumes to vSphere CSI:

1. Enable **vSphere CSI Drive Integration** on the Tanzu Kubernetes Grid Integrated Edition tile. For more information, see [Storage](#) in *Installing TKGI on vSphere*.
2. (Optional) Test migrating your in-tree vSphere storage volumes to vSphere CSI volumes. See [Migrate an In-Tree vSphere Storage Volume to the vSphere CSI Driver](#) below.

3. (Optional) To avoid a slow upgrade to TKGI v1.17, migrate your in-tree vSphere storage volumes to the vSphere CSI Driver before upgrading. See [Migrate an In-Tree vSphere Storage Volume to the vSphere CSI Driver](#) below.



**Note:** VMware strongly recommends that you migrate your in-tree vSphere storage volumes to vSphere CSI volumes as soon as possible.

## Migrate an In-Tree vSphere Storage Volume to the vSphere CSI Driver

You can use `tkgi update-cluster` to migrate the PersistentVolume (PV) and PersistentVolumeClaim (PVC) on an existing TKGI cluster from the In-Tree vSphere Storage Driver to the automatically installed vSphere CSI Driver.

Migrating a TKGI cluster from the In-Tree vSphere Storage Driver to the vSphere CSI Driver requires the following:

- You must use TKGI CLI v1.12 or later.
- TKGI automatic vSphere CSI Driver integration must be enabled. For information on enabling the **vSphere CSI Driver Integration** option on the TKGI tile, see [Storage in Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#).
- TKGI must be installed on vSphere v7.0 U2 or later.
- The cluster must be a Linux TKGI cluster.

To migrate a cluster from an In-Tree vSphere Storage Driver to the vSphere CSI Driver:

1. Upgrade your Kubernetes cluster to the current TKGI version of the TKGI tile.
2. Review and complete all relevant steps documented in the vSphere CSI Migration documentation:
  - [Prerequisites for Installing the vSphere Container Storage Plug-in](#)
  - [Migrating In-Tree vSphere Volumes to vSphere Container Storage Plug-in](#)
  - [vSphere Container Storage Plug-in Upgrade Considerations and Guidelines](#)



**Warning:** Before migrating to the vSphere CSI driver, confirm your cluster's volume storage is configured as described in [Considerations for Migration of In-Tree vSphere Volumes](#).

3. Create a configuration file containing the following:

```
{
 "enable_csi_migration": "true"
}
```

4. To migrate your cluster to the vSphere CSI Driver:

```
tkgi update-cluster CLUSTER-NAME --config-file CONFIG-FILE
```

Where:

- ◊ `CLUSTER-NAME` is the name of your cluster.
- ◊ `CONFIG-FILE` is the name of the config file you created in the preceding steps.

## Customize and Manage vSphere CNS

To configure or manage your vSphere CSI Driver:

- [Customize the Maximum Number of Volume Snapshots](#)
- [Configure CNS Data Centers](#)
- [Manage Topology After Switching to the Automatically Deployed vSphere CSI Driver](#)

### Customize the Maximum Number of Volume Snapshots

The vSphere CSI driver lets you customize the maximum number of snapshots for a persistent volume. By default, the system sets a maximum of three volume snapshots as suggested by the VMware snapshots best practices in a vSphere environment.

In your cluster configuration file, use the following parameters to customize the maximum number of snapshots:

- `global-max-snapshots-per-block-volume` for the block volumes on VMFS datastores. If you do not use this parameter, the system sets the maximum snapshots for the block volumes to 3.
- `granular-max-snapshots-per-block-volume-vsan` for the volumes on VMware vSAN. If you do not use this parameter, the system sets the maximum snapshots for the vSAN volumes to the value specified for `global-max-snapshots-per-block-volume`.

To customize the maximum number of snapshots on a persistent volume, create a JSON or YAML formatted configuration file containing the following:

```
{
 "snapshot_parameters":{
 "global_max_snapshots_per_block_volume": NUMBER
 }
}
```

Where:

- `NUMBER` is the maximum number of snapshots on the volume.

For example:

```
{
 "snapshot_parameters":{
```

```

 "global_max_snapshots_per_block_volume": 4
 }
}

```

To customize the maximum number of snapshots on a vSAN volume, create a JSON or YAML formatted configuration file containing the following:

```

{
 "snapshot_parameters":{
 "granular_max_snapshots_per_block_volume_vsan": NUMBER
 }
}

```

Where:

- **NUMBER** is the maximum number of snapshots on the volume.

For example:

```

{
 "snapshot_parameters":{
 "granular_max_snapshots_per_block_volume_vsan": 4
 }
}

```

To create a new cluster or update an existing cluster with the new snapshot configuration:

- To create a cluster:

```
tkgi create-cluster CLUSTER-NAME --config-file CONFIG-FILE
```

Where:

- **CLUSTER-NAME** is the name of your cluster.
- **CONFIG-FILE** is the name of your config file.

For example:

```
$ tkgi create-cluster demo -e demo.cluster -plan Small -config-file ./snapshot.json
```

- To update an existing cluster:

```
tkgi update-cluster CLUSTER-NAME --config-file CONFIG-FILE
```

Where:

- **CLUSTER-NAME** is the name of your cluster.
- **CONFIG-FILE** is the name of your config file.

For example:

```
$ tkgi update-cluster demo -config-file ./snapshot.json
```

For more information on volume snapshots, see [Volume Snapshot and Restore](#) in VMware vSphere

Container Storage Plug-in Documentation.

## Configure CNS Data Centers

If your clusters are in a multi-data center environment, configure the data centers that must mount CNS storage for the clusters.



**Note:** You must configure CNS data centers when the Topology feature is enabled in a multi-data center environment.

To configure CNS data centers for a multi-data center environment:

1. Create a JSON or YAML formatted configuration file containing the following:

```
{
 "csi_datacenters": "DATA-CENTER-LIST"
}
```

Where:

- ◆ **DATA-CENTER-LIST** is a comma-separated list of vCenter data centers that must mount your CNS storage. The default data center for a cluster is the data center defined on the TKGI tile in **Kubernetes Cloud Provider > Datacenter Name**.

For example:

```
{
 "csi_datacenters": "kubo-dc1,kubo-dc2"
}
```

For more information on the `csi_datacenters` parameter, see the description of `datacenters` in [Procedure in Create a Kubernetes Secret for vSphere Container Storage Plug-in](#).

2. To create a new cluster or update an existing cluster with your vCenter data centers:

- ◆ To create a cluster:

```
tkgi create-cluster CLUSTER-NAME --config-file CONFIG-FILE
```

Where:

- **CLUSTER-NAME** is the name of your cluster.
- **CONFIG-FILE** is the name of your config file.

For example:

```
$ tkgi create-cluster demo -e demo.cluster -plan Small -config-file ./conf1.json
```

- ◆ To update an existing cluster:

```
tkgi update-cluster CLUSTER-NAME --config-file CONFIG-FILE
```

Where:

- `CLUSTER-NAME` is the name of your cluster.
- `CONFIG-FILE` is the name of your config file.

## Manage Topology After Switching to the Automatically Deployed vSphere CSI Driver

After switching from a manually installed vSphere CSI Driver to the TKGI automatically deployed CSI Driver, the topology configuration must not be changed.

Configure topology based on the manually installed vSphere CSI Driver configuration:

- **Region and Zone Topology Labels:**

You must continue to use `region`, and `zone` labels if your manually deployed vSphere CSI Driver was configured using the legacy `region`, and `zone` topology configuration labels.

Your revised cluster configuration file must include a `csi_topology_labels` parameter that assigns `region` and `zone` values.

For example, if your vSphere Secret configuration for the manually installed vSphere CSI driver included the following:

```
[Labels]
region = k8s-region
zone = k8s-zone
```

Your new cluster configuration must include the following instead:

```
{
 "csi_topology_labels": {
 "region": "k8s-region"
 "zone": "k8s-zone"
 }
}
```

- **topology\_categories Topology Label:**

You must continue to use the `topology_categories` label if your manually deployed vSphere CSI Driver was configured using the `topology_categories` topology configuration label.

Your revised cluster configuration file must include a `csi_topology_labels` parameter that assigns a `topology_categories` value.

For example, if your vSphere Secret configuration for the manually installed vSphere CSI driver included the following:

```
[Labels]
topology-categories = "k8s-region, k8s-zone"
```

Your new cluster configuration must include the following instead:

```
{
 "csi_topology_labels": {
 "topology_categories": "k8s-region,k8s-zone"
 }
}
```

- **Topology Deactivated:**

You must not enable topology if topology was not enabled while using the manually deployed vSphere CSI Driver.

## PersistentVolume Storage Options on vSphere

This topic describes options for configuring VMware Tanzu Kubernetes Grid Integrated Edition on vSphere to support stateful apps using PersistentVolumes (PVs).



**Note:** This topic assumes that you have strong familiarity with PVs and workloads in Kubernetes.

For procedural information about configuring PVs, see [Configuring and Using PersistentVolumes](#).

For information about which vSphere CSI Driver features are supported by TKGI, see [vSphere CSI Driver Supported Features and Requirements](#) in *Deploying and Managing Cloud Native Storage (CNS) on vSphere*.

## Considerations for Running Stateful Apps in Kubernetes

There are several factors to consider when running stateful apps in Kubernetes:

- **Pods are ephemeral by nature.** Data that needs to be persisted must be accessible on restart and rescheduling of a pod.
- **When a pod is rescheduled, it might be on a different host.** Storage must be available on the new host for the pod to start gracefully.
- **The app should not manage the volume and data.** The underlying infrastructure should handle the complexity of unmounting and mounting.
- **Certain apps have a strong sense of identity.** When a container with a certain ID uses a disk, the disk becomes tied to that container. If a pod with a certain ID gets rescheduled, the disk associated with that ID must be reattached to the new pod instance.

## Persistent Volume Provisioning Support in Kubernetes

Kubernetes provides two ways to provision persistent storage for stateful applications:

- **Static provisioning:** A Kubernetes administrator creates the Virtual Machine Disk (VMDK) and PVs. Developers issue PersistentVolumeClaims (PVCs) on the pre-defined PVs.
- **Dynamic provisioning:** Developers issue PVCs against a StorageClass object. The provisioning of the persistent storage depends on the infrastructure. With Tanzu Kubernetes Grid Integrated Edition on vSphere, the vSphere Cloud Provider (VCP) automatically provisions the VMDK and PVs.

For more information about PVs in Kubernetes, refer to the [Kubernetes documentation](#).

PVs can be used with two types of Kubernetes workloads:

- [Deployments](#)
- [StatefulSets](#)

## vSphere Support for Static and Dynamic PVs

With Tanzu Kubernetes Grid Integrated Edition on vSphere, you can choose one of two storage options to support stateful apps:

- vSAN datastores
- Network File Share (NFS) or VMFS over Internet Small Computer Systems Interface (iSCSI), or fiber channel (FC) datastores

Refer to the [vSAN documentation](#) and the [VMFS documentation](#) for more information about these storage options.



**Note:** This topic assumes that you have strong familiarity vSAN and VMFS storage technologies on the vSphere platform.

In Tanzu Kubernetes Grid Integrated Edition, an availability zone (AZ) corresponds to a vSphere cluster and a resource pool within that cluster. A resource pool is a vSphere construct that is not linked to a particular ESXi host. Resource pools can be used in testing environments to enable a single vSphere cluster to support multiple AZs. As a recommended practice, deploy multiple AZs across different vSphere clusters to afford best availability in production.

The vSAN datastore boundary is delimited by the vSphere cluster. All ESXi hosts in the same vSphere cluster belong to the same vSAN datastore. ESXi hosts in a different vSphere cluster belong to a different vSAN datastore. Each vSphere cluster has its own vSAN datastore.

The table below summarizes Tanzu Kubernetes Grid Integrated Edition support for PVs in Kubernetes when deployed on vSphere:

| Storage Mechanism                                                                                                                                                         | vSAN datastore                                            | File system datastore<br>(VMFS/NFS over iSCSI/FC)         |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------|-----------------------------------------------------------|
| <ul style="list-style-type: none"> <li>• Single vSphere compute cluster with single datastore</li> <li>• Single AZ and resource pool</li> </ul>                           | Both static and dynamic PV provisioning are supported.    | Both static and dynamic PV provisioning are supported.    |
| <ul style="list-style-type: none"> <li>• Multiple vSphere compute clusters each with local datastore</li> <li>• Multiple AZs each using separate resource pool</li> </ul> | Neither static nor dynamic PV provisioning are supported. | Neither static nor dynamic PV provisioning are supported. |

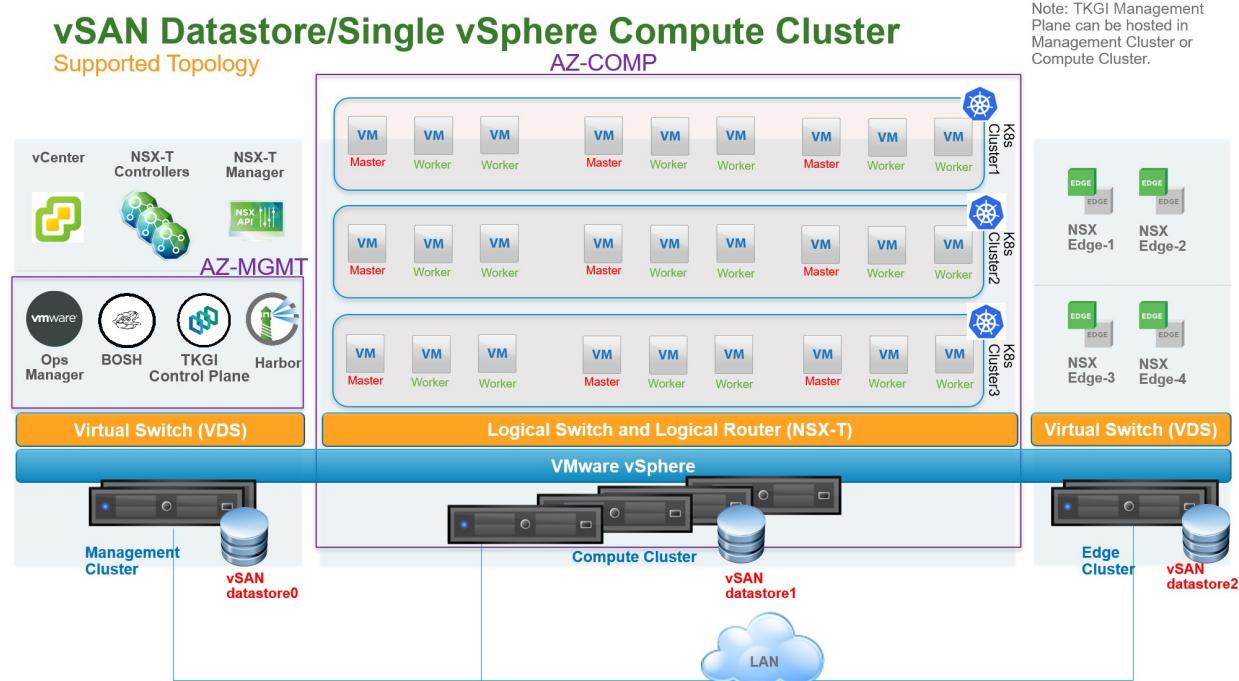
- Multiple vSphere compute clusters with shared datastore
  - Multiple AZs using a shared resource pool
- vSAN does not support sharing datastores across vSphere clusters. Can be accomplished by providing vSphere clusters with access to additional shared storage such as VMFS/NFS over iSCSI/FC.
- Both static and dynamic PV provisioning are supported.



**Note:** This information assumes that the underlying vSphere infrastructure is a single locality environment where all vSphere compute clusters are closed in terms of distance from one to the others. It does not apply to multi-site or vSAN stretched cluster configurations.

## Single vSphere Compute Cluster with vSAN Datastore

The following diagram illustrates a vSphere environment with a single compute cluster and a local vSAN datastore. This topology is also supported for environments with a single AZ or multiple AZs using multiple resource pools under the same vSphere cluster. For this topology, Tanzu Kubernetes Grid Integrated Edition supports both static and dynamic PV provisioning. Dynamic PV provisioning is recommended.



In this topology, a single vSphere compute cluster hosts all Kubernetes clusters. vSAN is enabled on the compute cluster which exposes a single unique vSAN datastore. In the above diagram, this datastore is labeled **vSAN datastore1**.

You can configure a single computer cluster in the following ways:

- If you use a single Tanzu Kubernetes Grid Integrated Edition foundation, create an AZ that is mapped directly to the single cluster.
- If you use multiple Tanzu Kubernetes Grid Integrated Edition foundations, create an AZ that is mapped to this single cluster and a Resource Pool.

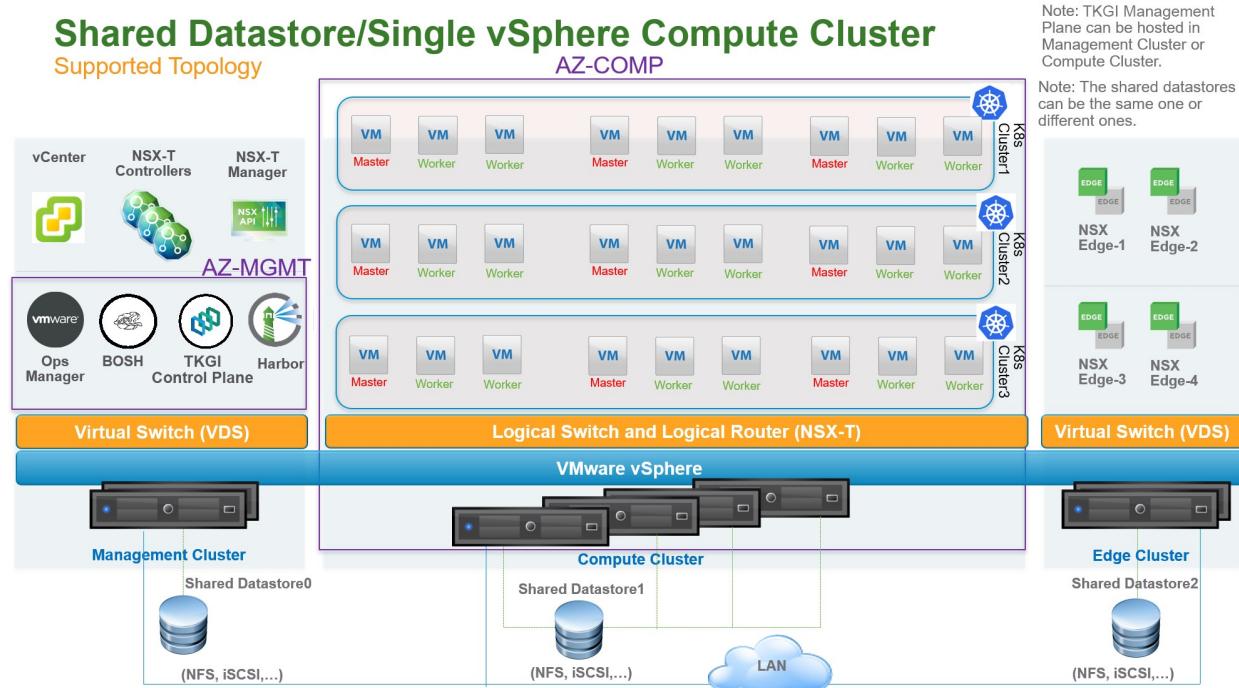
With this topology, you can create multiple vSAN datastores on the same compute cluster using different disk groups on each ESXi host. PVs, backed by respective VMDK files, can be dispatched across the datastores to mitigate the impact of datastore failure. For StatefulSets, all PVs used by different instances of the replica land in the same datastore.

This topology has the following failover scenarios:

- **Disks on ESXi hosts are down:** If the failure is within the limit of the vSAN `failure to tolerate` value, there is no impact on PVs.
- **ESXi hosts are down:** If the failure is within the limit of the vSAN `failure to tolerate` value, there is no impact on PVs.
- **Datastore is down:** PVs on the down datastore are unreachable.

## Single vSphere Compute Cluster with File System Datastore

The following diagram illustrates a vSphere environment with a single vSphere compute cluster and a shared datastore using NFS or VMFS over iSCSI, or FC. For this topology, Tanzu Kubernetes Grid Integrated Edition supports both static and dynamic PV provisioning. Dynamic PV provisioning is recommended.



In this topology, a single vSphere compute cluster hosts all Kubernetes clusters. The shared datastore is used with the compute cluster. In the above diagram, this datastore is labeled **Shared Datastore1**.

One or more AZs can be instantiated on top of the compute cluster. With this configuration, one or more AZs are mapped to vSphere resource pools. The AZ is not bound to a failure domain because its resource pool is not linked to a particular ESXi host.

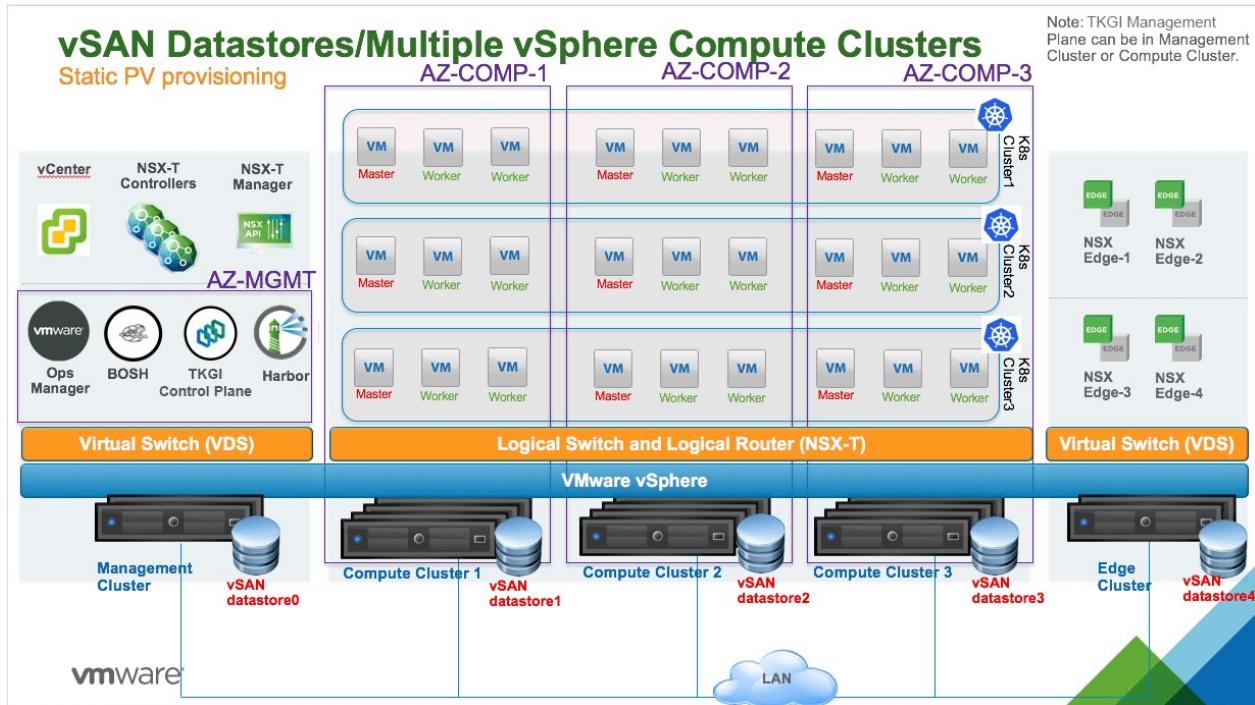
With this topology, you can create multiple shared datastores connected to the same compute cluster. PVs, backed by respective VMDK files, can be dispatched across the datastores to mitigate the impact of datastore failure. For StatefulSets, all PVs used by different instances of the replica land in the same datastore.

This topology has the following failover scenarios:

- **ESXi hosts are down:** No impact on PVs.
- **Datastore is down:** PVs on the down datastore are unreachable.

## Multiple vSphere Compute Clusters Each with vSAN Datastore

The following diagram illustrates a vSphere environment with multiple vSphere compute clusters with vSAN datastores that are local to each compute cluster.

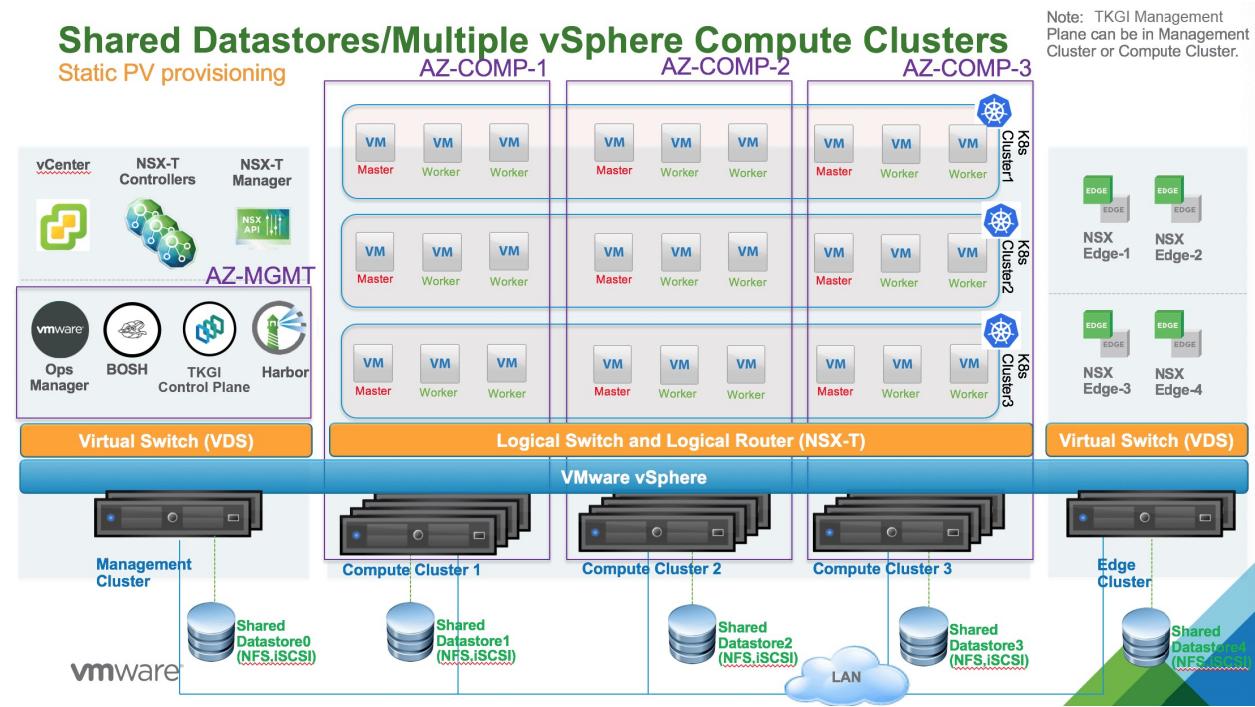


In this topology, vSAN is enabled on each compute cluster. There is one local vSAN datastore per compute cluster. For example, in the above diagram, vSAN datastore1 is provisioned for Compute Cluster 1 and vSAN datastore2 is provisioned for Compute Cluster 2.

One or more AZs can be instantiated. Each AZ is mapped to a vSphere compute cluster. The AZ is bound to a failure domain which is typically the physical rack where the compute cluster is hosted.

## Multiple vSphere Compute Clusters Each with File System Datastore

The following diagram illustrates a vSphere environment with multiple vSphere compute clusters with NFS or VMFS over iSCSI, or FC shared datastores.



In this topology, multiple vSphere compute clusters are used to host all Kubernetes clusters. A unique shared datastore is used per vSphere compute cluster. For example, in the above diagram, Shared Datastore1 is connected to Compute Cluster 1 and Shared Datastore2 is connected to Compute Cluster 2.

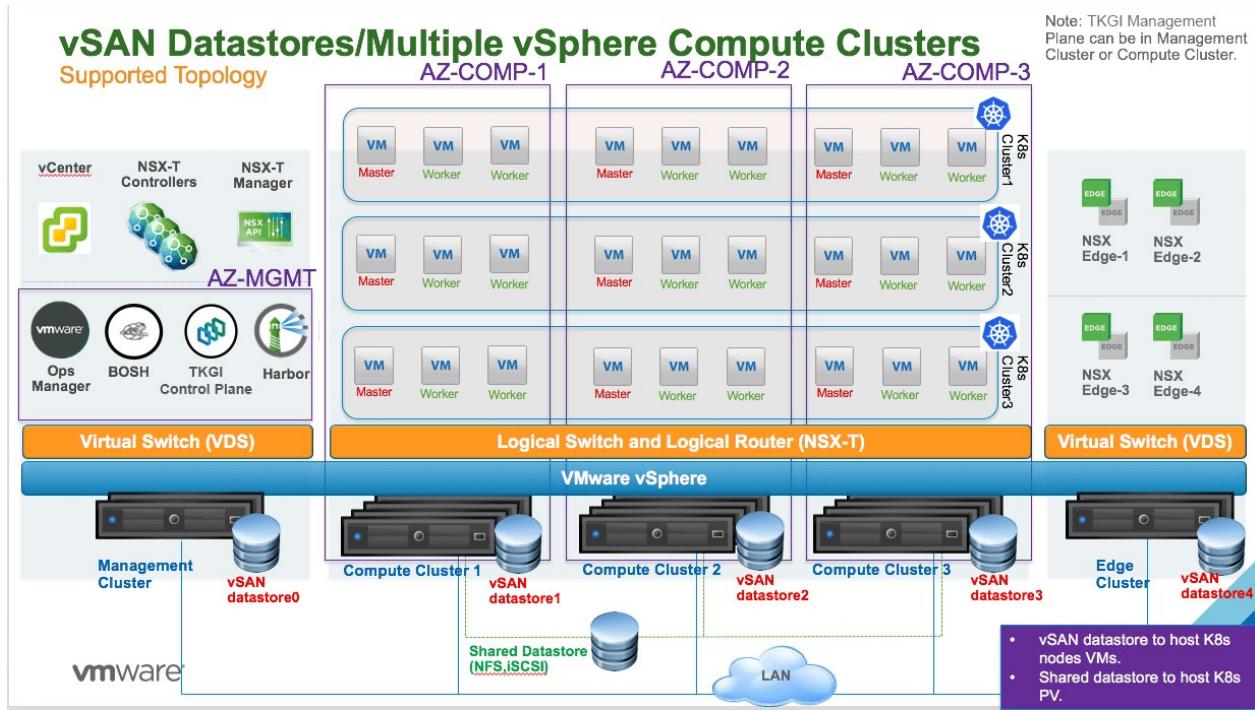
One or more AZs can be instantiated. Each AZ is mapped to a vSphere compute cluster. The AZ is bound to a failure domain which is typically the physical rack where the compute cluster is hosted.

## Multiple vSphere Compute Clusters with Local vSAN and Shared File System Datastore

With this topology, each vSAN datastore is only visible from each vSphere compute cluster. It is not possible to have a vSAN datastore shared across all vSphere compute clusters.

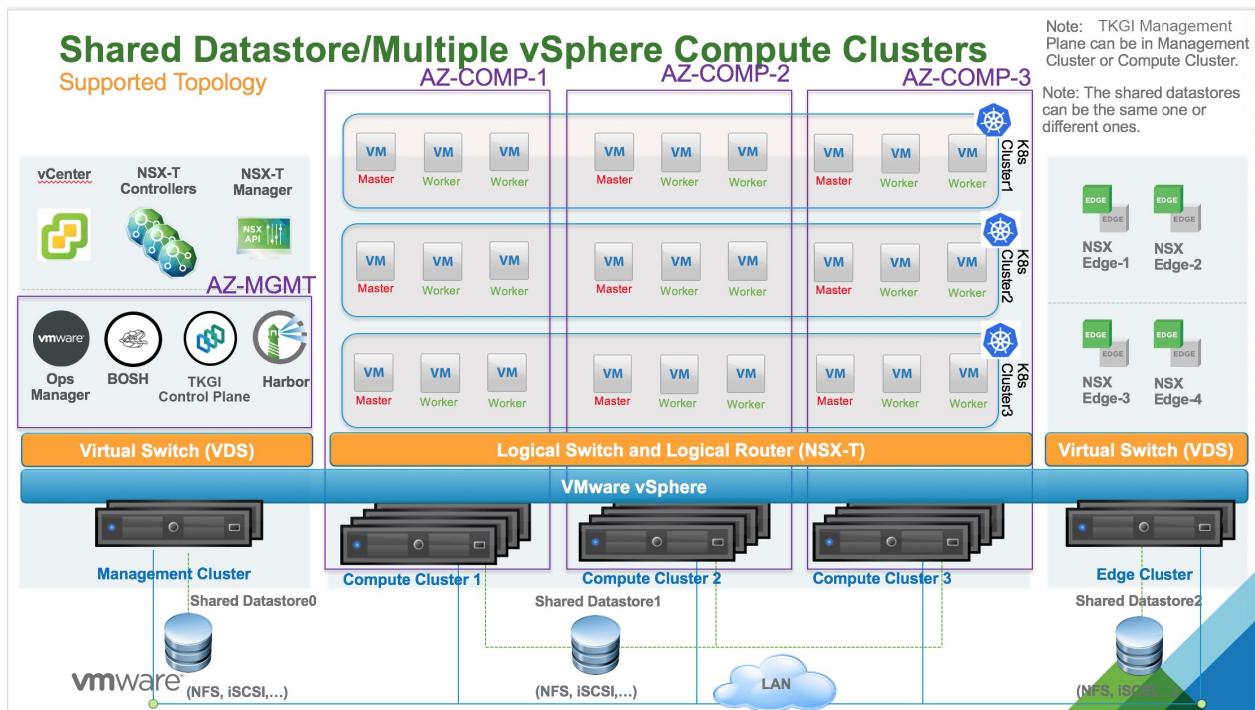
You can insert a shared NFS, iSCSI (VMFS), or FC (VMFS) datastore across all vSAN-based vSphere compute clusters to support both static and dynamic PV provisioning.

Refer to the following diagram:



## Multiple vSphere Compute Clusters with Shared File System Datastore

The following diagram illustrates a vSphere environment with multiple compute clusters with VMFS over NFS, iSCSI, or FC datastores shared across all vSphere compute clusters. For this topology, Tanzu Kubernetes Grid Integrated Edition supports both static and dynamic PV provisioning. Dynamic PV provisioning is recommended.



In this topology, multiple vSphere compute clusters are used to host all Kubernetes clusters. A unique shared datastore that uses NFS, or VMFS over iSCSI/FC is used across all compute clusters. In the above diagram, this datastore is labeled **Shared Datastore1**.

One or more AZs can be instantiated. Each AZ is mapped to a compute cluster. The AZ is bound to a failure domain which is typically the physical rack where the compute cluster is hosted.

You can have multiple shared datastores connected across all the vSphere compute clusters. PVs, backed by respective VMDK files, can then be dispatched across those datastores to mitigate the impact of datastore failure. For StatefulSets, all PVs used by different instances of the replica land in the same datastore.

This topology has the following failover scenarios:

- **ESXi hosts are down:** No impact on PVs.
- **One shared datastore is down:** PVs on the down datastore are unreachable.

## Configuring and Using PersistentVolumes

This topic describes how to provision static and dynamic PersistentVolumes (PVs) for VMware Tanzu Kubernetes Grid Integrated Edition to run stateful apps.

For static PV provisioning, the PersistentVolumeClaim (PVC) does not need to reference a StorageClass. For dynamic PV provisioning, you must specify a StorageClass and define the PVC using a reference to that StorageClass.

For more information about storage management in Kubernetes, see [Persistent Volumes](#) in the [Kubernetes Concepts](#) documentation.

For more information about the supported vSphere topologies for PV storage, see [PersistentVolume Storage Options on vSphere](#).

## Provision a Static PV

To provision a static PV, you manually create a Virtual Machine Disk (VMDK) file to use as a storage backend for the PV. When the PV is created, Kubernetes knows which volume instance is ready for use. When a PVC or volumeClaimTemplate is requested, Kubernetes chooses an available PV in the system and allocates it to the Deployment or StatefulSets workload.

## Provision a Static PV for a Deployment Workload

To provision a static PV for a Deployment workload, the procedure is as follows:



**Note:** The examples in this section use the vSphere volume plugin. Refer to the [Kubernetes documentation](#) for information about volume plugins for other cloud providers.

1. `ssh` into an ESXi host in your vCenter cluster that has access to the datastore where you will host the static PV.
2. Create VMDK files, replacing `DATASTORE` with your datastore directory name:

```
[root@ESXi-1:~] cd /vmfs
[root@ESXi-1:/vmfs] cd volumes/
[root@ESXi-1:/vmfs/volumes] cd DATASTORE/
[root@ESXi-1:/vmfs/volumes/7e6c0ca3-8c4873ed] cd kubevols/
```

```
[root@ESXi-1:/vmfs/volumes/7e6c0ca3-8c4873ed/kubevols] vmkfstools -c 2G redis-master.vmdk
```

3. Define a PV using a YAML manifest file that contains a reference to the VMDK file. For example, on vSphere, create a file named `redis-master-pv.yaml` with the following contents:

```
apiVersion: v1
kind: PersistentVolume
metadata:
 name: redis-master-pv
spec:
 capacity:
 storage: 2Gi
 accessModes:
 - ReadWriteOnce
 persistentVolumeReclaimPolicy: Retain
 vsphereVolume:
 volumePath: "[NFS-LAB-DATASTORE] kubevols/redis-master"
 fsType: ext4
```

4. Define a PVC using a YAML manifest file. For example, create a file named `redis-master-claim.yaml` with the following contents:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
 name: redis-master-claim
spec:
 accessModes:
 - ReadWriteOnce
 resources:
 requests:
 storage: 2Gi
```

5. Define a deployment using a YAML manifest file that references the PVC. For example, create a file named `redis-master.yaml` with the following contents:

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: redis-master
...
spec:
 template:
 spec:
 volumes:
 - name: redis-master-data
 persistentVolumeClaim:
 claimName: redis-master-claim
```

## Provision a Static PV for a StatefulSets Workload

To provision a static PV for a StatefulSets workload with three replicas, the procedure is as follows:



**Note:** The examples in this section use the vSphere volume plugin. Refer to the

[Kubernetes documentation](#) for information about volume plugins for other cloud providers.

1. Create VMDK files, replacing `DATASTORE` with your datastore directory name:

```
[root@ESXi-1:~] cd /vmfs
[root@ESXi-1:/vmfs] cd volumes/
[root@ESXi-1:/vmfs/volumes] cd DATASTORE/
[root@ESXi-1:/vmfs/volumes/7e6c0ca3-8c4873ed] cd kubevols/
[root@ESXi-1:/vmfs/volumes/7e6c0ca3-8c4873ed/kubevols] vmkfstools -c 10G mysql-pv-1.vmdk
[root@ESXi-1:/vmfs/volumes/7e6c0ca3-8c4873ed/kubevols] vmkfstools -c 10G mysql-pv-2.vmdk
[root@ESXi-1:/vmfs/volumes/7e6c0ca3-8c4873ed/kubevols] vmkfstools -c 10G mysql-pv-3.vmdk
```

2. Define a PV for the first replica using a YAML manifest file that contains a reference to the VMDK file. For example, on vSphere, create a file named `mysql-pv-1.yaml` with the following contents:

```
apiVersion: v1
kind: PersistentVolume
metadata:
 name: mysql-pv-1
spec:
 capacity:
 storage: 10Gi
 accessModes:
 - ReadWriteOnce
 persistentVolumeReclaimPolicy: Retain
 vsphereVolume:
 volumePath: "[NFS-LAB-DATASTORE] kubevols/mysql-pv-1"
 fsType: ext4
```

3. Define a PV for the second replica using a YAML manifest file that contains a reference to the VMDK file. For example, on vSphere, create a file named `mysql-pv-2.yaml` with the following contents:

```
apiVersion: v1
kind: PersistentVolume
metadata:
 name: mysql-pv-2
spec:
 capacity:
 storage: 10Gi
 accessModes:
 - ReadWriteOnce
 persistentVolumeReclaimPolicy: Retain
 vsphereVolume:
 volumePath: "[NFS-LAB-DATASTORE] kubevols/mysql-pv-2"
 fsType: ext4
```

4. Define a PV for the third replica using a YAML manifest file that contains a reference to the VMDK file. For example, on vSphere, create a file named `mysql-pv-3.yaml` with the following contents:

```

apiVersion: v1
kind: PersistentVolume
metadata:
 name: mysql-pv-3
spec:
 capacity:
 storage: 10Gi
 accessModes:
 - ReadWriteOnce
 persistentVolumeReclaimPolicy: Retain
 vsphereVolume:
 volumePath: "[NFS-LAB-DATASTORE] kubevols/mysql-pv-3"
 fsType: ext4

```

5. Define a StatefulSets object using a YAML manifest file. For example, create a file named `mysql-statefulsets.yaml` with the following contents:

```

apiVersion: apps/v1
kind: StatefulSet
metadata:
 name: mysql
spec:
 selector:
 matchLabels:
 app: mysql
 serviceName: mysql
 replicas: 3
...
volumeClaimTemplates:
 - metadata:
 name: data
 spec:
 accessModes: ["ReadWriteOnce"]
 resources:
 requests:
 storage: 10Gi

```



**Note:** In previous steps you created a total of three PVs. The `spec.replicas: 3` field defines three replicas. Each replica is attached to one PV.



**Note:** In the volumeClaimTemplates section, you must specify the required storage size for each replica. Do not refer to a StorageClass.

## Provision a Dynamic PV

Dynamic PV provisioning gives developers the freedom to provision storage when they need it without manual intervention from a Kubernetes cluster administrator. To enable dynamic PV provisioning, the Kubernetes cluster administrator defines one or more StorageClasses.

For dynamic PV provisioning, the procedure is to define and create a PVC that automatically triggers the creation of the PV and its backend VMDK file. When the PV is created, Kubernetes knows which volume instance is available for use. When a PVC or volumeClaimTemplate is requested,

Kubernetes chooses an available PV and allocates it to the Deployment or StatefulSets workload.

Tanzu Kubernetes Grid Integrated Edition supports dynamic PV provisioning by providing StorageClasses for all supported cloud providers, as well as an example PVC.



**Note:** For dynamic PVs on vSphere, you must create or map the VMDK file for the StorageClass on a shared file system datastore. This shared file system datastore must be accessible to each vSphere cluster where Kubernetes cluster nodes run. For more information, see [PersistentVolume Storage Options on vSphere](#).

## Provision a Dynamic PV for Deployment Workloads



**Note:** The examples in this section use the vSphere provisioner. Refer to the [Kubernetes documentation](#) for information about provisioners for other cloud providers.

For the Deployment workload with dynamic PV provisioning, the procedure is as follows:

1. Define a StorageClass using a YAML manifest file. For example, on vSphere, create a file named `redis-sc.yaml` with the following contents:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
 name: thin-disk
provisioner: kubernetes.io/vsphere-volume
```

2. Define a PVC using a YAML manifest file that references the StorageClass. For example, create a file named `redis-master-claim.yaml` with the following contents:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
 name: redis-master-claim
 annotations:
 volume.beta.kubernetes.io/storage-class: thin-disk
spec:
 accessModes:
 - ReadWriteOnce
 resources:
 requests:
 storage: 2Gi
```



**Note:** When you deploy the PVC on vSphere, the vSphere Cloud Provider plugin automatically creates the PV and associated VMDK file.

3. Define a Deployment using a YAML manifest file that references the PVC. For example, create a file named `redis-master.yaml` with the following contents:

```
apiVersion: apps/v1
```

```

kind: Deployment
metadata:
 name: redis-master
...
spec:
 template:
 spec:
 volumes:
 - name: redis-master-data
 persistentVolumeClaim:
 claimName: redis-master-claim

```

## Provision a Dynamic PV for StatefulSets Workloads



**Note:** The examples in this section use the vSphere provisioner. Refer to the [Kubernetes documentation](#) for information about provisioners for other cloud providers.

To provision a static PV for a StatefulSets workload with three replicas, the procedure is as follows:

1. Define a StorageClass using a YAML manifest file. For example, on vSphere, create a file named `mysql-sc.yaml` with the following contents:

```

kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
 name: my-storage-class
provisioner: kubernetes.io/vsphere-volume

```

2. Define a StatefulSets object using a YAML manifest file that references the StorageClass. For example, create a file named `mysql-statefulsets.yaml` with the following contents:

```

apiVersion: apps/v1
kind: StatefulSet
metadata:
 name: mysql
spec:
...
volumeClaimTemplates:
 - metadata:
 name: data
 spec:
 accessModes: ["ReadWriteOnce"]
 storageClassName: "my-storage-class"
 resources:
 requests:
 storage: 10Gi

```



**Note:** In the volumeClaimTemplates, specify the required storage size for each replica. Unlike static provisioning, you must explicitly refer to the desired StorageClass when you use dynamic PV provisioning.

## Specify a Default StorageClass

If you have or anticipate having more than one StorageClass for use with dynamic PVs for a Kubernetes cluster, you might want to designate a particular StorageClass as the default. This allows you to manage a storage volume without setting up specialized StorageClasses across the cluster.

If necessary, a developer can change the default StorageClass in the PVC definition. See the [Kubernetes documentation](#) for more information.

To specify a StorageClass as the default for a Kubernetes cluster, use the annotation `storageclass.kubernetes.io/is-default-class: "true"`.

For example:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
 name: thin-disk
 annotations:
 storageclass.kubernetes.io/is-default-class: "true"
provisioner: kubernetes.io/vsphere-volume
```



**Note:** The above example uses the vSphere provisioner. Refer to the [Kubernetes documentation](#) for information about provisioners for other cloud providers.

## Provision Dynamic PVs for Use with Tanzu Kubernetes Grid Integrated Edition

Perform the steps in this section to register one or more StorageClasses and define a PVC that can be applied to newly-created pods.

1. Download the StorageClass spec for your cloud provider by running the command for your cloud provider:
  - **AWS:** `wget https://raw.githubusercontent.com/cloudfoundry-incubator/kubo-ci/master/specs/storage-class-aws.yml`
  - **Azure:**
    - For Azure disk storage: `wget https://raw.githubusercontent.com/cloudfoundry-incubator/kubo-ci/master/specs/storage-class-azure.yml`
    - For Azure file storage: `wget https://raw.githubusercontent.com/cloudfoundry-incubator/kubo-ci/master/specs/storage-class-azure-file.yml`
  - **GCP:** `wget https://raw.githubusercontent.com/cloudfoundry-incubator/kubo-ci/master/specs/storage-class-gcp.yml`
  - **vSphere:** `wget https://raw.githubusercontent.com/cloudfoundry-incubator/kubo-ci/master/specs/storage-class-vsphere.yml` After downloading the vSphere StorageClass spec, replace the contents of the file with the following YAML to create the correct StorageClass for vSphere:

```

kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
 name: thin
 annotations:
 storageclass.kubernetes.io/is-default-class: "true"
provisioner: kubernetes.io/vsphere-volume

```

2. Apply the spec by running the following command:

```
kubectl create -f STORAGE-CLASS-SPEC.yml
```

Where **STORAGE-CLASS-SPEC** is the name of the file that you downloaded in the previous step.  
For example:

```
$ kubectl create -f storage-class-gcp.yml
```

3. Download the example PVC by running the following command:

```
wget https://raw.githubusercontent.com/cloudfoundry-incubator/kubo-ci/master/specs/persistent-volume-claim.yml
```

4. Apply the PVC by running the following command:

```
kubectl create -f persistent-volume-claim.yml
```

5. Confirm that you applied the PVC by running the following command:

```
kubectl get pvc -o wide
```

6. To use the dynamic PV, create a pod that uses the PVC. For an example, see the [pv-guestbook.yml configuration file](#) in the kubo-ci repository in GitHub.

## Supporting Windows Clusters

This section describes how to support Windows worker-based Kubernetes clusters provisioned by VMware Tanzu Kubernetes Grid Integrated Edition.

See the following topics:

- [Configuring Windows Worker-Based Kubernetes Clusters](#)
- [Creating a Windows Stemcell for vSphere Using Stembuild](#)
- [Using a Windows Pause Image for an Air-Gapped Environment](#)

## Configuring Windows Worker-Based Kubernetes Clusters

This topic describes configuring Windows worker-based Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition (TKGI).

## Overview

In Tanzu Kubernetes Grid Integrated Edition you can provision a Windows worker-based Kubernetes cluster on vSphere with NSX-T. Additionally, TKGI provides beta support for provisioning Windows worker-based Kubernetes clusters on vSphere with Flannel.

To provision a Windows worker-based Kubernetes cluster:

1. Verify your environment meets the Windows worker-based Kubernetes cluster [Prerequisites](#).
2. Configure a Windows Worker-Based Kubernetes Cluster.
3. Upload the Windows Server Stemcell.
4. Create a Windows Worker-Based Cluster.

For information about the architecture of TKGI Windows worker-based Kubernetes clusters, see [Windows Worker-Based Kubernetes Cluster High Availability](#) in *Tanzu Kubernetes Grid Integrated Edition Architecture*.



**Warning:** Support for Windows-based Kubernetes clusters is activated for TKGI on vSphere with NSX-T and as a beta feature on vSphere with Flannel.

Do not activate this feature if you are using TKGI with Google Cloud Platform (GCP), Azure, or Amazon Web Services (AWS).

## Prerequisites

Support for Windows-based Kubernetes clusters is activated for TKGI on vSphere with NSX-T and as a beta feature on vSphere with Flannel.

## vSphere with NSX-T Requirements

The following are required for creating a Windows worker-based Kubernetes cluster in Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T:

- Tanzu Kubernetes Grid Integrated Edition must be installed in a vSphere with NSX-T environment.
- Your vSphere environment meets the [vSphere with NSX-T Version Requirements](#).
- Tanzu Kubernetes Grid Integrated Edition has been configured as described in [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T](#).
- You must have a vSphere stemcell for Windows Server version 2019. For vSphere stemcell version requirements, see [Product Snapshot](#) in *Release Notes*.



**Note:** Windows stemcells for vSphere are not available on [VMware Tanzu Network](#). These stemcells must be created using your own Windows Server disk image (ISO file). To create a Windows stemcell for vSphere, complete the procedures in [Creating a Windows Stemcell for vSphere Using Stembuild](#).

- If your Tanzu Kubernetes Grid Integrated Edition installation is in an air-gapped environment,

you must prepare a Windows pause image in a private registry. For information about setting up a Windows pause image, see [Using a Windows Pause Image for an Air-Gapped Environment](#).

## vSphere with Flannel Requirements (Beta)

The following are required for creating a Windows worker-based Kubernetes cluster in Tanzu Kubernetes Grid Integrated Edition on vSphere with Flannel:

- Tanzu Kubernetes Grid Integrated Edition must be installed in a vSphere with Flannel environment.
- Your vSphere environment meets the [vSphere Prerequisites and Resource Requirements](#).
- Tanzu Kubernetes Grid Integrated Edition has been configured as described in [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#).
- You must have a vSphere stemcell for Windows Server version 2019. For vSphere stemcell version requirements, see [Product Snapshot](#) in *Release Notes*.



**Note:** Windows stemcells for vSphere are not available on [VMware Tanzu Network](#). These stemcells must be created using your own Windows Server disk image (ISO file). To create a Windows stemcell for vSphere, complete the procedures in [Creating a Windows Stemcell for vSphere Using Stembuild](#) in the TAS for VMs [Windows] documentation.

- If your Tanzu Kubernetes Grid Integrated Edition installation is in an air-gapped environment, you must prepare a Windows pause image in a private registry. For information about setting up a Windows pause image, see [Using a Windows Pause Image for an Air-Gapped Environment](#).

## Configure a Windows Worker-Based Kubernetes Cluster

1. Configure a Windows worker plan as described in [Plans](#), below.
2. Configure Windows worker networking as described in [Networking](#), below.
3. Upload the Windows Server stemcell as described in [Upload the Windows Server Stemcell](#), below.
4. Click **Apply Changes** to complete the configuration changes.

## Plans

A plan defines a set of resource types used for deploying a cluster.

### Activate a Plan



**Note:** Before configuring your Windows worker plan, you must first activate and configure [Plan 1](#). See [Plans](#) in *Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T* for more information.

To activate and configure a plan, perform the following steps:

1. Click the plan that you want to activate. You must activate and configure either **Plan 11**, **Plan 12**, or **Plan 13** to deploy a Windows worker-based cluster.
2. Select **Active** to activate the plan and make it available to developers deploying clusters.

## Configuration for Plan 11

Select 'Active' to allow users of the PKS CLI to create a cluster using this template plan.

### Plan\*

- Inactive  
 Active

### Name \*

Plan-11-Windows

### Description \*

Example: This plan will configure a large kubernetes cluster for resource heavy workloads, or a high number of workloads.

### Cluster Services

- Enable HA Linux workers

### Master/ETCD Node Instances ( min: 1, max: 5 ) \*

3

### Master/ETCD VM Type\*

Automatic: m5.large (cpu: 2, ram: 7.5 GB, disk: 32 GB) ▾

### Master Persistent Disk Type\*

Automatic: 10 GB ▾

### Master/ETCD Availability Zones \*

- us-east-1a  
 us-east-1b  
 us-east-1c

3. Under **Name**, provide a unique name for the plan.
4. Under **Description**, edit the description as needed. The plan description appears in the Services Marketplace, which developers can access by using the TKGI CLI.
5. Select **Enable HA Linux workers** to activate high availability Linux worker clusters. A high availability Linux worker cluster consists of three Linux worker nodes.
  - ◊ Windows workers are mediated by one or three Linux workers.
  - ◊ For an illustration of how Linux workers connect Windows workers to their control plane node, see [Windows Worker-Based Kubernetes Cluster High Availability](#).
6. Under **Master/ETCD Node Instances**, select the default number of Kubernetes control plane/etcd nodes to provision for each cluster. You can enter [1](#), [3](#), or [5](#).



**Note:** If you deploy a cluster with multiple control plane/etcd node VMs, confirm that you have sufficient hardware to handle the increased load on disk write and network traffic. For more information, see [Hardware recommendations](#) in the etcd documentation.

In addition to meeting the hardware requirements for a multi-control plane node cluster, we recommend configuring monitoring for etcd to monitor disk latency, network latency, and other indicators for the health of the cluster. For more information, see [Configuring Telegraf in TKGI](#).



**WARNING:** To change the number of control plane/etcd nodes for a plan, you must ensure that no existing clusters use the plan. Tanzu Kubernetes Grid Integrated Edition does not support changing the number of control plane/etcd nodes for plans with existing clusters.

7. Under **Master/ETCD VM Type**, select the type of VM to use for Kubernetes control plane/etcd nodes. For more information, including control plane node VM customization options, see the [Control Plane Node VM Size](#) section of [VM Sizing for Tanzu Kubernetes Grid Integrated Edition Clusters](#).
8. Under **Master Persistent Disk Type**, select the size of the persistent disk for the Kubernetes control plane node VM.
9. Under **Master/ETCD Availability Zones**, select one or more AZs for the Kubernetes clusters deployed by Tanzu Kubernetes Grid Integrated Edition. If you select more than one AZ, Tanzu Kubernetes Grid Integrated Edition deploys the control plane VM in the first AZ and the worker VMs across the remaining AZs. If you are using multiple control plane nodes, Tanzu Kubernetes Grid Integrated Edition deploys the control plane and worker VMs across the AZs in round-robin fashion.



**Note:** Tanzu Kubernetes Grid Integrated Edition does not support changing the AZs of existing control plane nodes.

10. Under **Maximum number of workers on a cluster**, set the maximum number of Kubernetes

worker node VMs that Tanzu Kubernetes Grid Integrated Edition can deploy for each cluster. Enter any whole number in this field.

Maximum number of workers on a cluster ( min: 1 ) \*

50

Worker Node Instances ( min: 1 ) \*

5

Worker VM Type\*

Automatic: r5.xlarge (cpu: 4, ram: 30.5 GB, disk: 128 GB) ▾

Worker Availability Zones \*

- us-east-2a
- us-east-2b
- us-east-2c

11. Under **Worker Node Instances**, specify the default number of Kubernetes worker nodes the TKGI CLI provisions for each cluster. The **Worker Node Instances** setting must be less than, or equal to, the **Maximum number of workers on a cluster** setting.

For high availability, create clusters with a minimum of three worker nodes, or two per AZ if you intend to use PersistentVolumes (PVs). For example, if you deploy across three AZs, you should have six worker nodes. For more information about PVs, see [PersistentVolumes](#) in *Maintaining Workload Uptime*. Provisioning a minimum of three worker nodes, or two nodes per AZ is also recommended for stateless workloads.

For more information about creating clusters, see [Creating Clusters](#).



**Note:** Changing a plan's **Worker Node Instances** setting does not alter the number of worker nodes on existing clusters. For information about scaling an existing cluster, see [Scale Horizontally by Changing the Number of Worker Nodes Using the TKGI CLI](#) in *Scaling Existing Clusters*.

12. Under **Worker VM Type**, select the type of VM to use for Kubernetes worker node VMs. For more information, including worker node VM customization options, see [Worker Node VM Number and Size](#) in *VM Sizing for Tanzu Kubernetes Grid Integrated Edition Clusters*.



**Note:** Tanzu Kubernetes Grid Integrated Edition requires a **Worker VM Type** with an ephemeral disk size of 32 GB or more.



**Note:** BOSH does not support persistent disks for Windows VMs. If specifying **Worker Persistent Disk Type** on a Windows worker is a requirement for you, submit feedback by sending an email to [pcf-windows@pivotal.io](mailto:pcf-windows@pivotal.io).

13. Under **Worker Availability Zones**, select one or more AZs for the Kubernetes worker nodes. Tanzu Kubernetes Grid Integrated Edition deploys worker nodes equally across the AZs you select.
14. Under **Kubelet customization - system-reserved**, enter resource values that Kubelet can use to reserve resources for system daemons. For example, `memory=250Mi, cpu=150m`. For more information about system-reserved values, see the [Kubernetes documentation](#).

#### Kubelet customization - system-reserved

#### Kubelet customization - eviction-hard

#### Kubelet customization - Windows pause image location \*

`mcr.microsoft.com/k8s/core/pause:1.2.0`

#### Errand VM Type\*

Automatic: micro (cpu: 1, ram: 1 GB, disk: 8 GB) ▾

15. Under **Kubelet customization - eviction-hard**, enter threshold limits that Kubelet can use to evict pods when they exceed the limit. Enter limits in the format `EVICTION-SIGNAL=QUANTITY`. For example, `memory.available=100Mi, nodefs.available=10%, nodefs.inodesFree=5%`. For more information about eviction thresholds, see the [Kubernetes documentation](#).



**WARNING:** Use the Kubelet customization fields with caution. If you enter values that are invalid or that exceed the limits the system supports, Kubelet might fail to start. If Kubelet fails to start, you cannot create clusters.

16. Under **Kubelet customization - Windows pause image location**, enter the location of your Windows pause image. The **Kubelet customization - Windows pause image location** default value of `mcr.microsoft.com/k8s/core/pause:3.6` configures Tanzu Kubernetes Grid Integrated Edition to pull the Windows pause image from the Microsoft Docker registry. The Microsoft Docker registry cannot be accessed from within air-gapped environments. If you want to deploy Windows pods in an air-gapped environment you must upload a Windows pause image to an accessible private registry, and configure the **Kubelet customization - Windows pause image location** field with the URI to this accessible Windows pause image. For more information about uploading a Windows pause image to a private registry, see [Using a Windows Pause Image for an Air-Gapped Environment](#).
17. Under **Errand VM Type**, select the size of the VM that contains the errand. The smallest instance possible is sufficient, as the only errand running on this VM is the one that applies the **Default Cluster App** YAML configuration.
18. (Optional) Under **(Optional) Add-ons - Use with caution**, enter additional YAML configuration to add custom workloads to each cluster in this plan. You can specify multiple files using `--` as a separator. For more information, see [Adding Custom Linux Workloads](#).

#### (Optional) Add-ons - Use with caution

#### Admission Plugins

- PodSecurityPolicy
- SecurityContextDeny



**Note:** Windows in Kubernetes does not support privileged containers. See [Feature Restrictions](#) in the Kubernetes documentation for additional information.

19. (Optional) Activate or deactivate the **SecurityContextDeny** admission controller plugin. For more information see [Using Admission Control Plugins for Tanzu Kubernetes Grid Integrated Edition Clusters](#). Windows in Kubernetes does not support the **PodSecurityPolicy** Admission Plugin feature. See [API](#) in the Kubernetes documentation for additional information.
20. Click **Save**.

## Networking

To configure networking, do the following:

1. Click **Networking**.
2. Under **Container Networking Interface**, select:
  - **NSX-T** for Windows worker based clusters on vSphere with NSX-T.
  - **Flannel** for Windows worker based clusters on vSphere without NSX-T beta.

## Networking Configurations

Container Networking Interface\*

Antrea (Switching from Flannel to Antrea is only supported on upgrade)

Flannel

Kubernetes Pod Network CIDR Range \*

10.200.0.0/16

Kubernetes Service Network CIDR Range \*

10.10.200.0/24

NSX-T

HTTP/HTTPS Proxy (for vSphere and AWS only)\*

Disabled

Enabled

Allow outbound internet access from Kubernetes cluster vms (IaaS-dependent)

Enable outbound internet access

**Save**



**Note:** Antrea is not supported for the TKGI Windows-worker on vSphere without NSX-T beta feature.

3. (Optional) Enter values for **Kubernetes Pod Network CIDR Range** and **Kubernetes Service Network CIDR Range**.

- For Windows worker-based clusters the **Kubernetes Service Network CIDR Range** setting must be **10.220.0.0/16**.



**Note:** vSphere on Flannel does not support networking Windows containers.

HTTP/HTTPS Proxy (for vSphere and AWS only)\*

Disabled  
 Enabled

HTTP Proxy URL

HTTP Proxy Credentials

Username  
 Password

HTTPS Proxy URL

HTTPS Proxy Credentials

Username  
 Password

No Proxy

4. (Optional) Configure Tanzu Kubernetes Grid Integrated Edition to use a proxy.

Production environments can deny direct access to public Internet services and between internal services by placing an HTTP or HTTPS proxy in the network path between Kubernetes nodes and those services.

Configure Tanzu Kubernetes Grid Integrated Edition to use your proxy and activate the following:

- ◊ TKGI API access to public Internet services and other internal services.
- ◊ Tanzu Kubernetes Grid Integrated Edition-deployed Kubernetes nodes access to public Internet services and other internal services.
- ◊ Tanzu Kubernetes Grid Integrated Edition Telemetry ability to forward Telemetry data to the CEIP and Telemetry program.



**Note:** This setting does not set the proxy for running Kubernetes workloads or pods.

5. To complete your global proxy configuration for all outgoing HTTP/HTTPS traffic from your

Kubernetes clusters, perform the following steps:

1. To proxy outgoing HTTP traffic, enter the URL of your HTTP proxy endpoint under **HTTP Proxy URL**. For example, `http\://myproxy.com:1234`.
2. (Optional) If your outgoing HTTP proxy uses basic authentication, enter the username and password in the **HTTP Proxy Credentials** fields.
3. To proxy outgoing HTTPS traffic, enter the URL of your HTTP proxy endpoint under **HTTPS Proxy URL**. For example, `http\://myproxy.com:1234`.



**Note:** Using an HTTPS connection to the proxy server is not supported. HTTP and HTTPS proxy options can only be configured with an HTTP connection to the proxy server. You cannot populate either of the proxy URL fields with an HTTPS URL. The proxy host and port can be different for HTTP and HTTPS traffic, but the proxy protocol must be HTTP.

4. (Optional) If your HTTPS proxy uses basic authentication, enter the username and password in the **HTTPS Proxy Credentials** fields.
5. Under **No Proxy**, enter the comma-separated list of IP addresses that must bypass the proxy to allow for internal Tanzu Kubernetes Grid Integrated Edition communication.

The **No Proxy** list should include `127.0.0.1` and `localhost`.

Also include the following in the **No Proxy** list:

- Your Tanzu Kubernetes Grid Integrated Edition environment's CIDRs, such as the service network CIDR where your Tanzu Kubernetes Grid Integrated Edition cluster is deployed, the deployment network CIDR, the node network IP block CIDR, and the pod network IP block CIDR.
- The FQDN of any registry, such as the Harbor API FQDN, or component communicating with Tanzu Kubernetes Grid Integrated Edition, using a hostname instead of an IP address.
- Any additional IP addresses or domain names that should bypass the proxy.

The **No Proxy** property for AWS accepts wildcard domains denoted by a prefixed `\*`. or ..

For example:

```
127.0.0.1,localhost, *.example1.com, .example2.com, example3.com,
198.51.100.0/24, 203.0.113.0/24, 192.0.2.0/24
```



**Note:** By default the `10.100.0.0/8` and `10.200.0.0/8` IP

address ranges, `.internal`, `.svc`, `.svc.cluster.local`, `.svc.cluster`, and your Tanzu Kubernetes Grid Integrated Edition FQDN are not proxied. This allows internal Tanzu Kubernetes Grid Integrated Edition communication.

Do not use the `_` character in the **No Proxy** field. Entering an underscore character in this field can cause upgrades to fail.

Because some jobs in the VMs accept `\*` as a wildcard, while others only accept `.`, we recommend that you define a wildcard domain using both of them. For example, to denote `example.com` as a wildcard domain, add both `\*.example.com` and `example.com` to the **No Proxy** property.

6. Under **Allow outbound internet access from Kubernetes cluster vms (IaaS-dependent)**, ignore the **Enable outbound internet access** checkbox.
7. Click **Save**.

## Upload the Windows Server Stemcell

1. When prompted by Ops Manager to upload a stemcell, follow the instructions and provide your previously created vSphere stemcell for Windows Server version 2019.

## Create a Windows Worker-Based Cluster

1. To create a Windows worker-based cluster follow the steps in [Creating Clusters](#).

## Creating a Windows Stemcell for vSphere Using Stembuild

This topic describes how to use the tool Stembuild to create a Windows stemcell for BOSH, for use by Tanzu Kubernetes Grid Integrated Edition (TKGI) on vSphere.

A **BOSH stemcell** is a versioned operating system image.

You must create a BOSH stemcell for Windows before you can deploy Windows workers in Kubernetes clusters using Tanzu Kubernetes Grid Integrated Edition on vSphere.

## Overview of Stembuild

Stembuild is a binary that you use to build BOSH stemcells for Windows Server 2019.

Stembuild creates a BOSH stemcell from a base Windows image. The Stembuild CLI has two commands, `construct` and `package`, which you run against a Windows Server 2019 VM. These commands are used to create a stemcell in [Construct and Package the BOSH Stemcell](#) below.

## Overview of Windows Stemcell Creation

To create a Windows stemcell for vSphere, you create a base Windows VM from a volume-licensed ISO and subsequently maintain that base template with all Windows-recommended security updates,

but without the BOSH dependencies.

The Windows VM with security updates serves as the base for all future stemcells produced from clones of that base VM. This enables you to build new stemcells without having to run Windows updates from scratch each time. You can also use a “snapshot” feature to maintain an updated Windows image that does not contain the BOSH dependencies.

VMware recommends installing any available critical updates and then rebuilding the stemcell from a clone of the original VM.

The BOSH stemcell that you create in this topic is based on Windows Server 2019. If you already have a BOSH stemcell for Windows on vSphere, see [Monthly Stemcell Upgrades](#) below.

For more information, see [Best Practices for Stembuild for Tanzu Application Service & Tanzu Kubernetes Grid Integrated Edition] (<https://tanzu.vmware.com/content/practitioners/best-practices-for-stembuild-for-tanzu-application-service-tanzu-kubernetes-grid-integrated-edition>) in *VMware Tanzu Tech Tutorials*.

To construct, package and upload a BOSH Stemcell for Windows to TKGI, complete the following:

1. [Prerequisites](#)
2. [Create and Configure a Base VM for the BOSH Stemcell](#)
3. [Construct and Package the BOSH Stemcell](#)
4. [Update Ops Manager With the Updated Stemcell](#)

## Prerequisites

Before you create a BOSH Windows stemcell for Tanzu Kubernetes Grid Integrated Edition on vSphere, you must have:

- A vSphere environment. To ensure the VM hardware used by the stemcell is compatible with your deployment environment’s ESXi/ESX host and vCenter Server versions, see [ESXi/ESX hosts and compatible virtual machine hardware versions list (2007240)] (<https://kb.vmware.com/s/article/2007240>) in the VMware Knowledge Base.
- An ISO for a Windows Server 2019 Server Core installation, build number: 17763, from [Microsoft Developer Network \(MSDN\)](#) or [Microsoft Volume Licensing Service Center \(VLSC\)](#). The Windows Server 2019 ISO must be a clean, base ISO file. You can use an evaluation copy for testing, but VMware does not recommend an evaluation copy for production because the licensing expires. For more information, see the [Windows Server documentation](#) or the [Microsoft Volume Licensing Service Center](#) website.



**Note:** A clean ISO file has no custom scripts or tooling. For example, the ISO must have no logging or antivirus tools installed.

- Download the following from [Stemcells \(Windows\)](#) on VMware Tanzu Network:
  - A Windows stemcell
  - A `stembuild` command line interface (CLI) from a 2019.x release

Refer to [Product Snapshot](#) in *Release Notes* for the compatible version of each to download.

- Microsoft Local Group Policy Object Utility (LGPO) downloaded to the same folder as your `stembuild` CLI.
- The minimum vCenter user permissions required to use `stembuild` for vSphere stemcells, specifically:
  - ◊ `VirtualMachine.GuestOperations.Modify`
  - ◊ `VirtualMachine.GuestOperations.Execute`
  - ◊ `VirtualMachine.GuestOperations.Query`
  - ◊ `VirtualMachine.Config.AddRemoveDevice`
  - ◊ `VirtualMachine.Interact.SetCDMedia`
  - ◊ `VApp.Export`
  - ◊ `System.Anonymous*`
  - ◊ `System.Read*`
  - ◊ `System.View*`

Permissions marked with an \* are generated upon creating a new user in vCenter and cannot be set within the vCenter UI.

## Create and Configure a Base VM for the BOSH Stemcell

Before using Stembuild to create a stemcell, you need to create a Windows Server 2019 VM and update the VM with the latest Windows updates.

To do this, follow these procedures in the TAS for VMs [Windows] documentation, in order:

1. [Create a Base VM for the BOSH Stemcell](#)
2. [Configure the Base VM](#)
3. [Clone the Base VM](#)

## Construct and Package the BOSH Stemcell

To create, configure, and package a BOSH Stemcell, follow these procedures, in order:

1. [Construct the BOSH Stemcell](#), in the TAS for VMs [Windows] documentation.
2. [Remove Hidden Devices](#), below.
3. [Package the BOSH Stemcell](#), in the TAS for VMs [Windows] documentation.

## Remove Hidden Devices

To ensure your BOSH Windows stemcell can work properly, confirm the stemcell does not have any hidden devices:

1. Open the vSphere Management console.
2. Confirm a network adapter is not assigned to the target VM.

3. Power on the target VM.
4. Log into the target VM.
5. Start PowerShell.
6. Confirm the VM has hidden devices:

```
Get-PnpDevice -Class net | ? Status -eq Unknown
```

7. If there are hidden devices, clean up those devices:

```
$Devs = Get-PnpDevice -Class net | ? Status -eq Unknown
ForEach ($Dev in $Devs) {
 Write-Host "Removing $($Dev.FriendlyName)" -ForegroundColor Cyan
 $RemoveKey = "HKLM:\SYSTEM\CurrentControlSet\Enum\$($Dev.InstanceId)"
 Get-Item $RemoveKey | Select-Object -ExpandProperty Property | %{
 Remove-ItemProperty -Path $RemoveKey -Name $_ -Verbose
 }
}
```

8. Power off the target VM.



**Note:** The `ovs-windows` job requires a VM with a net-adapter named `Ethernet0`. Remove hidden devices from the target VM to ensure the `Ethernet0` net-adapter name is not taken before the `ovs-windows` job starts.

## Update Ops Manager With the Updated Stemcell

To update Ops Manager with the new BOSH Windows stemcell:

1. Open Ops Manager.
2. Navigate to the **Stemcell Library**.
3. Replace the existing stemcell in the Ops Manager stemcell library with your new updated stemcell.
4. Deploy the TKGI tile.

## Monthly Stemcell Upgrades

Microsoft typically releases Windows updates with security patches on the second Tuesday of each month.

After each Microsoft Windows security update, you should update your BOSH stemcell by following these procedures, in order:

1. [Configure the Base VM in the TAS for VMs \[Windows\]](#) documentation.
2. [Construct and Package the BOSH Stemcell](#), above.
3. [Update Ops Manager With the Updated Stemcell](#), above.

## Known Issues

For known issues with stemcell creation, see [Known Issues in the TAS for VMs \[Windows\]](#) documentation.

## Using a Windows Pause Image for an Air-Gapped Environment

This topic describes configuring a private registry and a Windows pause image for an air-gapped environment for Windows worker-based Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition (TKGI).

### Overview

To deploy a Windows pod, Kubelet deploys a Windows container image fetched from a Docker registry.

Microsoft restricts distribution of Windows container base images and the fetched Windows container image is typically pulled from the Microsoft Docker registry. This registry is inaccessible from within an air-gapped environment.

To deploy Windows pods in an air-gapped environment you must have a Windows container image in a private Docker registry:

- [Prepare Your Private Docker Registry](#)
- [Prepare a Windows Pause Image for an Air-Gapped Environment](#)
- [Configure Tanzu Kubernetes Grid Integrated Edition to Use the Windows Pause Image](#)

### Prepare Your Private Docker Registry

Your private Docker registry must meet the following requirements:

- The registry must be accessible from your Tanzu Kubernetes Grid Integrated Edition environment.
- The registry must be configured to support Microsoft Windows images. For an example of a Windows-supporting registry, see [\[Pushing Images\]](#) (<https://goharbor.io/docs/2.0.0/working-with-projects/working-with-images/pulling-pushing-images/>) in the VMware Harbor documentation.

Follow the instructions for the Docker registry you chose to configure an accessible Windows image-supporting registry.

### Prepare a Windows Pause Image for an Air-Gapped Environment

To prepare a Windows pause image for an air-gapped environment, perform the following:

1. Create an accessible Windows Server 2019 machine in your environment.
2. Install Docker on this Windows Server 2019 machine.
3. Configure the machine's Docker daemon to allow non-redistributable artifacts to be pushed to your private registry. For information about configuring your Docker daemon, see [Allow](#)

[push of nondistributable artifacts in the Docker documentation.](#)

4. Open a command line on the Windows machine.
5. To download a Windows container image from the Microsoft Docker registry, run the following command:

```
docker pull mcr.microsoft.com/oss/kubernetes/pause:3.6
```

6. To tag the Windows container image, run the following command:

```
docker tag mcr.microsoft.com/oss/kubernetes/pause:3.6 REGISTRY-ROOT/windows/pause:3.6
```

Where `REGISTRY-ROOT` is your private registry's URL.

7. To upload the Windows container image to your accessible private registry, run the following command:

```
docker push PAUSE-IMAGE-URI
```

Where `PAUSE-IMAGE-URI` is the URI to the Windows pause image in your private registry.

Your pause image URI should follow the pattern: `my.private.registry/windows/pause:3.6`.

## Configure Tanzu Kubernetes Grid Integrated Edition to Use the Windows Pause Image

To configure Tanzu Kubernetes Grid Integrated Edition to fetch your accessible Windows container image when deploying Windows pods, perform the following:

1. Open the Tanzu Kubernetes Grid Integrated Edition tile.
2. Click the Windows worker Plan that you want to configure to use your accessible private registry.
3. Modify the **Kubelet customization - Windows pause image location** property to be your pause image URI.

For example:

```
my.private.registry/windows/pause:3.6
```

4. Click **Save**.

## Deploying Workloads

This section describes how to deploy workloads to Kubernetes clusters provisioned by VMware Tanzu Kubernetes Grid Integrated Edition.

See the following topics:

- [Deploying and Exposing Basic Linux Workloads](#)
- [Deploying and Exposing Basic Windows Workloads](#)

- [Adding Custom Linux Workloads](#)
- [Using Helm with Tanzu Kubernetes Grid Integrated Edition](#)

## Deploying and Exposing Basic Linux Workloads

This topic describes how to configure, deploy, and expose basic workloads in VMware Tanzu Kubernetes Grid Integrated Edition.

### Overview

A load balancer is a third-party device that distributes network and application traffic across resources. Using a load balancer can prevent individual network components from being overloaded by high traffic.



**Note:** The procedures in this topic create a dedicated load balancer for each workload. If your cluster has many apps, a load balancer dedicated to each workload can be an inefficient use of resources. An ingress controller pattern is better suited for clusters with many workloads.

Refer to the following Tanzu Kubernetes Grid Integrated Edition documentation topics for additional information about deploying and exposing workloads:

- For the different types of load balancers used in a deployment, see [Load Balancers in TKGI](#).
- For ingress routing on GCP, AWS, Azure, or vSphere without NSX-T, see [Configuring Ingress Routing](#).
- For ingress routing on vSphere with NSX-T, see [Configuring Ingress Resources and Load Balancer Services](#).

### Prerequisites

This topic references standard Kubernetes primitives. If you are unfamiliar with Kubernetes primitives, review the Kubernetes [Workloads](#) and [Services](#), [Load Balancing](#), and [Networking](#) documentation before following the procedures below.

### vSphere without NSX-T Prerequisites

If you use vSphere without NSX-T, you can choose to configure your own external load balancer or expose static ports to access your workload without a load balancer. See [Deploy Workloads without a Load Balancer](#) below.

### GCP, AWS, Azure, and vSphere with NSX-T Prerequisites

If you use Google Cloud Platform (GCP), Amazon Web Services (AWS), Azure, or vSphere with NSX-T integration, your cloud provider can configure a public-cloud external load balancer for your workload. See either [Deploy Workloads on vSphere with NSX-T](#) or [Deploy Workloads on GCP, AWS, or Azure, Using a Public-Cloud External Load Balancer](#) below.

### AWS Prerequisites

If you use AWS, you can also expose your workload using a public-cloud internal load balancer.

Perform the following steps before you create a load balancer:

1. In the [AWS Management Console](#), create or locate a public subnet for each availability zone (AZ) that you are deploying to. A public subnet has a route table that directs internet-bound traffic to the internet gateway.
2. On the command line, run `tkgi cluster CLUSTER-NAME`, where `CLUSTER-NAME` is the name of your cluster.
3. Record the unique identifier for the cluster.
4. In the [AWS Management Console](#), tag each public subnet based on the table below, replacing `CLUSTER-UUID` with the unique identifier of the cluster. Leave the **Value** field empty.

| Key                                                                            | Value |
|--------------------------------------------------------------------------------|-------|
| <code>kubernetes.io/cluster/service-instance_</code> <code>CLUSTER-UUID</code> | empty |

 **Note:** AWS limits the number of tags on a subnet to 100.

After completing these steps, follow the steps below in [Deploy AWS Workloads Using an Internal Load Balancer](#).

## Deploy Workloads on vSphere with NSX-T

If you use vSphere with NSX-T, follow the steps below to deploy and expose basic workloads using the NSX-T load balancer.

### Configure Your Workload

1. Open the Kubernetes service configuration file for your workload in a text editor.
2. To expose the workload through a load balancer, confirm that the Service object is configured to be `type: LoadBalancer`.
3. To deactivate load balancer SNAT mode, add the following `annotations` tag to the services `metadata` section of the manifest:

```
annotations:
 ncp/transparent-lb: True
```

For example:

```

apiVersion: v1
kind: Service
metadata:
 labels:
 name: nginx
 name: nginx
 annotations:
 ncp/transparent-lb: True
```

```

spec:
 ports:
 - port: 80
 selector:
 app: nginx
 type: LoadBalancer

```



**Note:** You can deactivate load balancer SNAT for only Layer 4 cluster load balancers with auto-scaling deactivated in a single-tier Policy API topology. To deactivate auto-scaling, see [cni\\_configurations Extensions Parameters](#) in *Creating and Managing Network Profiles (NSX-T Only)*.

4. Confirm that the Kubernetes service configuration of the workload is set to `type: LoadBalancer`.
5. Confirm that the `type` property of the Kubernetes service for each workload is similarly configured.



**Note:** For an example of a fully configured Kubernetes service, see the `type: LoadBalancer` configuration for the nginx app example in the kubo-ci repository in GitHub.

For more information about configuring the `LoadBalancer` Service type see [Type LoadBalancer](#) in the *Service* section of the Kubernetes documentation.

## Deploy and Expose Your Workload

1. To deploy the service configuration for your workload, run the following command:

```
kubectl apply -f SERVICE-CONFIG
```

Where `SERVICE-CONFIG` is your workload's Kubernetes service configuration.

For example:

```
$ kubectl apply -f nginx.yml
```

This command creates three pod replicas, spanning three worker nodes.

2. Deploy your applications, deployments, config maps, persistent volumes, secrets, and any other configurations or objects necessary for your applications to run.
3. Wait until your cloud provider has created and connected a dedicated load balancer to the worker nodes on a specific port.

## Access Your Workload

1. To determine the load balancer IP address and port number of your exposed workload, run the following command:

```
kubectl get svc SERVICE-NAME
```

Where `SERVICE-NAME` is the specified service `name` of your workload configuration.  
For example:

```
$ kubectl get svc nginx
```

2. Retrieve the external IP address and port of the load balancer from the returned listing.
3. To access the app, run the following command:

```
curl http://EXTERNAL-IP:PORT
```

Where:

- ◊ `EXTERNAL-IP` is the IP address of the load balancer.
- ◊ `PORT` is the port number.



**Note:** This command should be run on a server with network connectivity and visibility to the IP address of the worker node.

## Deploy Workloads on GCP, AWS, or Azure, Using a Public-Cloud External Load Balancer

If you use GCP, AWS, or Azure, follow the steps below to deploy and expose basic workloads using a load balancer configured by your cloud provider.

### Configure Your Workload

1. Open the Kubernetes service configuration file for your workload in a text editor.
2. To expose the workload through a load balancer, confirm that the Service object is configured to be `type: LoadBalancer`.

For example:

```

apiVersion: v1
kind: Service
metadata:
 labels:
 name: nginx
 name: nginx
spec:
 ports:
 - port: 80
 selector:
 app: nginx
 type: LoadBalancer

```

3. Confirm that the Kubernetes service configuration of the workload is set to `type: LoadBalancer`.

4. Confirm that the `type` property of the Kubernetes service for each workload is similarly configured.



**Note:** For an example of a fully configured Kubernetes service, see the `type: LoadBalancer` configuration for the nginx app example in the [kubo-ci](#) repository in GitHub.

For more information about configuring the `LoadBalancer` Service type see [Type LoadBalancer](#) in the *Service* section of the Kubernetes documentation.

## Deploy and Expose Your Workload

1. To deploy the service configuration for your workload, run the following command:

```
kubectl apply -f SERVICE-CONFIG
```

Where `SERVICE-CONFIG` is your workload's Kubernetes service configuration.

For example:

```
$ kubectl apply -f nginx.yml
```

This command creates three pod replicas, spanning three worker nodes.

2. Deploy your applications, deployments, config maps, persistent volumes, secrets, and any other configurations or objects necessary for your applications to run.
3. Wait until your cloud provider has created and connected a dedicated load balancer to the worker nodes on a specific port.

## Access Your Workload

1. To determine the load balancer IP address and port number of your exposed workload, run the following command:

```
kubectl get svc SERVICE-NAME
```

Where `SERVICE-NAME` is the specified service `name` of your workload configuration.

For example:

```
$ kubectl get svc nginx
```

2. Retrieve the external IP address and port of the load balancer from the returned listing.
3. To access the app, run the following command:

```
curl http://EXTERNAL-IP:PORT
```

Where:

- ◊ `EXTERNAL-IP` is the IP address of the load balancer.

- ◊ `PORT` is the port number.



**Note:** This command should be run on a server with network connectivity and visibility to the IP address of the worker node.

## Deploy AWS Workloads Using an Internal Load Balancer

If you use AWS, follow the steps below to deploy, expose, and access basic workloads using an internal load balancer configured by your cloud provider.

### Configure Your Workload

1. Open the Kubernetes service configuration file for your workload in a text editor.
2. To expose the workload through a load balancer, confirm that the Service object is configured to be `type: LoadBalancer`.
3. In the services metadata section of the manifest, add the following `annotations` tag:

```
annotations:
 service.beta.kubernetes.io/aws-load-balancer-internal: 0.0.0.0/0
```

For example:

```

apiVersion: v1
kind: Service
metadata:
 labels:
 name: nginx
 annotations:
 service.beta.kubernetes.io/aws-load-balancer-internal: 0.0.0.0/0
 name: nginx
spec:
 ports:
 - port: 80
 selector:
 app: nginx
 type: LoadBalancer

```

4. Confirm that the Kubernetes service configuration for the workload is set to `type: LoadBalancer`.
5. Confirm that the `annotations` and `type` properties of the Kubernetes service for each workload are similarly configured.



**Note:** For an example of a fully configured Kubernetes service, see the `type: LoadBalancer` configuration for the `nginx` app example in the `kubo-ci` repository in GitHub.

For more information about configuring the `LoadBalancer` Service type see [Type LoadBalancer](#) in

the *Service* section of the Kubernetes documentation.

## Deploy and Expose Your Workload

1. To deploy the service configuration for your workload, run the following command:

```
kubectl apply -f SERVICE-CONFIG
```

Where `SERVICE-CONFIG` is the Kubernetes service configuration of your workload.

For example:

```
$ kubectl apply -f nginx.yml
```

This command creates three pod replicas, spanning three worker nodes.

2. Deploy your applications, deployments, config maps, persistent volumes, secrets, and any other configurations or objects necessary for your applications to run.
3. Wait until your cloud provider has created and connected a dedicated load balancer to the worker nodes on a specific port.

## Access Your Workload

1. To determine the load balancer IP address and port number of your exposed workload, run the following command:

```
kubectl get svc SERVICE-NAME
```

Where `SERVICE-NAME` is the specified service `name` of your workload configuration.

For example:

```
$ kubectl get svc nginx
```

2. Retrieve the external IP and port of the load balancer from the returned listing.
3. To access the app, run the following command:

```
curl http://EXTERNAL-IP:PORT
```

Where:

- ◊ `EXTERNAL-IP` is the IP address of the load balancer.
- ◊ `PORT` is the port number.



**Note:** Run this command on a server with network connectivity and visibility to the IP address of the worker node.

## Deploy Workloads for a Generic External Load Balancer

Follow the steps below to deploy and access basic workloads using a generic external load balancer, such as F5.

In this approach you will access your workloads with a generic external load balancer.

Using a generic external load balancer requires a static port in your Kubernetes cluster. To do this we must expose your workloads with a `NodePort`.

## Configure Your Workload

To expose a static port on your workload, perform the following steps:

1. Open the Kubernetes service configuration file for your workload in a text editor.
2. To expose the workload without a load balancer, confirm that the Service object is configured to be `type: NodePort`.  
For example:

```

apiVersion: v1
kind: Service
metadata:
 labels:
 name: nginx
 name: nginx
spec:
 ports:
 - port: 80
 selector:
 app: nginx
 type: NodePort

```

3. Confirm that the Kubernetes service configuration of the workload is set to `type: NodePort`.
4. Confirm that the `type` property of the Kubernetes service for each workload is similarly configured.



**Note:** For an example of a fully configured Kubernetes service, see the [type: LoadBalancer configuration](#) for the nginx app example in the kubo-ci repository in GitHub.

For more information about configuring the `NodePort` Service type see [Type NodePort](#) in the Service section of the Kubernetes documentation.

## Deploy and Expose Your Workload

1. To deploy the service configuration for your workload, run the following command:

```
kubectl apply -f SERVICE-CONFIG
```

Where `SERVICE-CONFIG` is the Kubernetes service configuration of your workload.

For example:

```
$ kubectl apply -f nginx.yml
```

This command creates three pod replicas, spanning three worker nodes.

2. Deploy your applications, deployments, config maps, persistent volumes, secrets, and all other configurations or objects necessary for your applications to run.
3. Wait until your cloud provider has connected your worker nodes on a specific port.

## Access Your Workload

1. Retrieve the IP address for a worker node with a running app pod.



**Note:** If you deployed more than four worker nodes, some worker nodes might not contain a running app pod. Select a worker node that contains a running app pod.

You can retrieve the IP address for a worker node with a running app pod in one of the following ways: \* On the command line, run the following command:

```
kubectl get nodes -L spec.ip
```

- ◊ On the Ops Manager command line, run the following command to find the IP address:

```
bosh vms
```

This IP address will be used when configuring your external load balancer.

2. To see a listing of port numbers, run the following command:

```
kubectl get svc SERVICE-NAME
```

Where `SERVICE-NAME` is the specified service `name` of your workload configuration.

For example:

```
$ kubectl get svc nginx
```

3. Find the node port number in the `3XXXX` range. You use this port number when configuring your external load balancer.
4. Configure your external load balancer to map your application Uri to the IP and port number that you collected above. Refer to your load balancer documentation for instructions.

## Deploy Workloads without a Load Balancer

If you do not use an external load balancer, you can configure your service to expose a static port on each worker node. The following steps configure your service to be reachable from outside the cluster at `http://NODE-IP:NODE-PORT`.

### Configure Your Workload

To expose a static port on your workload, perform the following steps:

1. Open the Kubernetes service configuration file for your workload in a text editor.
2. To expose the workload without a load balancer, confirm that the Service object is configured to be `type: NodePort`.  
For example:

```

apiVersion: v1
kind: Service
metadata:
 labels:
 name: nginx
 name: nginx
spec:
 ports:
 - port: 80
 selector:
 app: nginx
 type: NodePort

```

3. Confirm that the Kubernetes service configuration of the workload is set to `type: NodePort`.
4. Confirm that the `type` property of the Kubernetes service for each workload is similarly configured.



**Note:** For an example of a fully configured Kubernetes service, see the [type: LoadBalancer configuration](#) for the nginx app example in the kubo-ci repository in GitHub.

For more information about configuring the `NodePort` Service type see [Type NodePort](#) in the [Service](#) section of the Kubernetes documentation.

## Deploy and Expose Your Workload

1. To deploy the service configuration for your workload, run the following command:

```
kubectl apply -f SERVICE-CONFIG
```

Where `SERVICE-CONFIG` is the Kubernetes service configuration of your workload.

For example:

```
$ kubectl apply -f nginx.yml
```

This command creates three pod replicas, spanning three worker nodes.

2. Deploy your applications, deployments, config maps, persistent volumes, secrets, and any other configurations or objects necessary for your applications to run.
3. Wait until your cloud provider has connected your worker nodes on a specific port.

## Access Your Workload

1. Retrieve the IP address for a worker node with a running app pod.



**Note:** If you deployed more than four worker nodes, some worker nodes might not contain a running app pod. Select a worker node that contains a running app pod.

You can retrieve the IP address for a worker node with a running app pod in one of the following ways: \* On the command line, run the following command:

```
kubectl get nodes -l spec.ip
```

- On the Ops Manager command line, run the following command to find the IP address:

```
bosh vms
```

2. To see a listing of port numbers, run the following command:

```
kubectl get svc SERVICE-NAME
```

Where `SERVICE-NAME` is the specified service `name` of your workload configuration.

For example:

```
$ kubectl get svc nginx
```

3. Find the node port number in the `3XXXX` range.
4. To access the app, run the following command:

```
curl http://NODE-IP:NODE-PORT
```

Where:

- `NODE-IP` is the IP address of the worker node.
- `NODE-PORT` is the node port number.



**Note:** Run this command on a server with network connectivity and visibility to the IP address of the worker node.

## Deploying and Exposing Basic Windows Workloads

This topic describes deploying Windows worker-based Kubernetes clusters in Tanzu Kubernetes Grid Integrated Edition (TKGI).

### Overview

In Tanzu Kubernetes Grid Integrated Edition, you can deploy Windows-based workloads to Kubernetes clusters on vSphere with NSX-T. Additionally, TKGI provides beta support for deploying Windows-based workloads to Kubernetes clusters on vSphere without NSX-T.

To deploy a new Windows-based workload to a new pod, do the following:

1. Access Your Windows-Based Cluster
2. Configure a Pod Deployment Manifest
3. Deploy the Pod
4. Configure a Service Manifest
5. Deploy the Service



**Warning:** Support for Windows-based Kubernetes clusters is enabled for TKGI on vSphere with NSX-T and as a beta feature for TKGI on vSphere without NSX-T.

Do not enable this feature if you are using TKGI with Google Cloud Platform (GCP), Azure, or Amazon Web Services (AWS).

## Prerequisites

You can deploy Windows workloads to Windows-based clusters only.

You must configure the Tanzu Kubernetes Grid Integrated Edition tile to support Windows-based clusters before you can use Windows-based clusters in TKGI. For instructions on configuring the Tanzu Kubernetes Grid Integrated Edition tile, see [Configuring Windows Worker-Based Clusters](#).

## Access Your Windows-Based Cluster

Your command line must have access to your Windows-based cluster to deploy Windows VMs and workloads to the cluster.

1. To determine which of your existing clusters is Windows-based, use the following command:

```
tkgi clusters
```

For example:

```
$ tkgi clusters

Name Plan Name UUID
Status Action

windows-k8s Plan-11-Windows-Beta 881543kd-64fg-7826-hea6-3h7g1o04kh0e
 succeeded Create

second-windows-k8s Plan-11-Windows-Beta 951547d1-67kg-9631-bju8-7h9s3o98br0q
 succeeded Create
```

Only clusters configured on Plans 11, 12, or 13 are Windows-based.

2. To access your Windows-based cluster, run the following command:

```
tkgi get-credentials CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of your Windows-based cluster.

For example:

```
$ tkgi get-credentials windows-k8s
Fetching credentials for cluster windows-k8s.
Context set for cluster windows-k8s.

You can now switch between clusters by using:
$kubectl config use-context <cluster-name>
```

The `tkgi get-credentials` command creates a local `kubeconfig`, allowing you to manage the cluster from the command line. For more information about the `tkgi get-credentials` command, see [Retrieving Cluster Credentials and Configuration](#).

- To verify you have established access to the correct cluster, run the following command:

```
kubectl cluster-info
```

- (Optional) To review the existing pods in the cluster, run the following command:

```
kubectl get pods
```

## Deploy a Windows Worker Pod

A pod deployment manifest file configures the VMs deployed to a pod.

### Configure a Pod Deployment Manifest

You must create a Windows worker deployment manifest before deploying your new Windows worker pod.

- To create a Windows worker deployment manifest, create a new YAML file containing the following:

```

apiVersion: apps/v1
kind: Deployment
metadata:
 labels:
 app: POD-NAME
 name: POD-NAME
spec:
 replicas: 1
 selector:
 matchLabels:
 app: POD-NAME
```

```

template:
 metadata:
 labels:
 app: POD-NAME
 name: POD-NAME
 spec:
 containers:
 - name: CONTAINER-NAME
 image: CONTAINER-FILE:VERSION
 env:
 - name: PORT
 value: "80"
 ports:
 - name: http
 containerPort: 80
 nodeSelector:
 kubernetes.io/os: windows
 tolerations:
 - key: "windows"
 operator: "Equal"
 value: "2019"
 effect: "NoSchedule"

```

Where:

- ◊ `POD-NAME` is the name of your pod.
- ◊ `CONTAINER-NAME` is the internal name of your container.
- ◊ `CONTAINER-FILE` is the filename of your container.
- ◊ `VERSION` is the version of the image to use, for example `latest`.

For example:

```

apiVersion: apps/v1
kind: Deployment
metadata:
 labels:
 app: win-webserver
 name: win-webserver
spec:
 replicas: 1
 selector:
 matchLabels:
 app: win-webserver
 template:
 metadata:
 labels:
 app: win-webserver
 name: win-webserver
 spec:
 containers:
 - name: windowswebserver
 image: stefanscherer/webserver-windows:0.4.0
 env:
 - name: PORT
 value: "80"
 ports:

```

```

- name: http
 containerPort: 80
nodeSelector:
 kubernetes.io/os: windows
tolerations:
- key: "windows"
 operator: "Equal"
 value: "2019"
 effect: "NoSchedule"

```

## Deploy the Pod

1. To deploy a new Windows worker pod, run the following command:

```
kubectl apply -f POD-CONFIG-FILE
```

Where `POD-CONFIG-FILE` is the filename of the Windows worker deployment manifest created above.

2. To confirm the status of the new pod, and the creation of new Windows worker nodes, run the following commands:

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

For example:

```
$ kubectl apply -f first-k8s.yml deployment.extensions/win-webserver
created
$ kubectl get pods -o wide

NAME READY STATUS RESTARTS AGE IP
NODE
NOMINATED NODE READINE
SS GATES

win-webserver-795g866cd7-58oct 1/1 Running 0 88s 10.200
.42.4 0983934a-6d69-8e5g-g3k1-98r8r561345j <none>
$ kubectl get nodes -o wide

NAME STATUS ROLES AGE VERSION
ON INTERNAL-IP EXTERNAL-IP OS-IMAGE
RNEL-VERSION CONTAINER-RUNTIME
KE

0983934a-6d69-8e5g-g3k1-98r8r561345j Ready <none> 19d v1.24
.1 10.85.41.118 10.85.41.118 Windows Server 2019 Datacenter 10
.0.17763.503 containerd://18.9.0

6d69934a-7d43-9g3g-h4d1-54r9r971395j Ready <none> 19d v1.24
.1 10.85.41.115 10.85.41.115 Ubuntu 16.04.6 LTS
15.0-50-generic containerd://18.9.0

7636d69a-2e75-510g-k6m1-76r3r371729k Ready <none> 19d v1.24
.1 10.85.41.117 10.85.41.117 Windows Server 2019 Datacenter 10
.0.17763.503 containerd://18.9.0
```

```
406d694a-9g96-2d3g-f3j1-32r1r441342x Ready <none> 19d v1.24
.1 10.85.41.116 10.85.41.116 Windows Server 2019 Datacenter 10
.0.17763.503 containerd://18.9.0
```

In the preceding example a new pod is created, and creation and status of the new pod and new nodes verified.



**Note:** The `ping` command does not work reliably for Windows workers. For more information, see [Pinging Windows Workers Does Not Work](#) in *Release Notes*.

## Deploy a Service to a Windows Worker Pod

A service deployment manifest file configures your service, defining how your service will run and how it will be exposed.

### Configure a Service Manifest

You must create a Windows service deployment manifest before deploying your Windows worker workload.

1. To create a Windows service deployment manifest, create a new YAML file containing the following:

```

apiVersion: v1
kind: Service
metadata:
 name: APP-NAME
 labels:
 app: APP-NAME
spec:
 ports:
 # the port that this service should serve on
 - port: 80
 targetPort: 80
 selector:
 app: APP-NAME
 type: LoadBalancer
```

Where `APP-NAME` is the name of your Windows service.

For example:

```

apiVersion: v1
kind: Service
metadata:
 name: win-webserver
 labels:
 app: win-webserver
spec:
```

```

ports:
 # the port that this service should serve on
 - port: 80
 targetPort: 80
 selector:
 app: win-webserver
 type: LoadBalancer

```

## Deploy the Service

1. To expose your service on a LoadBalancer, run the following command:

```
kubectl apply -f SERVICE-CONFIG-FILE
```

Where `SERVICE-CONFIG-FILE` is the filename of your Windows service deployment manifest created above.

For example:

```

$ kubectl get services

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S)
AGE
kubernetes ClusterIP 10.100.200.1 <none> 443/TCP
20d

$ kubectl apply -f first-k8s-service.yml

service/win-webserver created
$ kubectl get services

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S)
AGE
kubernetes ClusterIP 10.100.200.1 <none> 443/TCP
20d

win-webserver LoadBalancer 10.100.200.221 <none> 80:32073/T
CP 5s

$ curl 10.85.41.118:32073

```

```

License.txt;
ProgramData/;
Users/;
WcSandboxState/;
Windows/;
var/;
webserver.exe;

```

In the preceding example a new service is created, verified, and validated.

## Adding Custom Linux Workloads

This topic describes how to add custom workloads to VMware Tanzu Kubernetes Grid Integrated Edition clusters.

Custom workloads define what a cluster includes out of the box. For example, you can use custom workloads to configure metrics or logging.

## Create YAML Configuration

Create a YAML configuration for your custom workloads. Consult the following example from the [Kubernetes documentation](#):

```
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
kind: Deployment
metadata:
 name: nginx-deployment
spec:
 selector:
 matchLabels:
 app: nginx
 replicas: 2 # tells deployment to run 2 pods matching the template
 template: # create pods using pod definition in this template
 metadata:
 # unlike pod-nginx.yaml, the name is not included in the meta data as a unique name is
 # generated from the deployment name
 labels:
 app: nginx
 spec:
 containers:
 - name: nginx
 image: nginx:1.7.9
 ports:
 - containerPort: 80
```

## Apply Custom Workloads

To apply custom Kubernetes workloads to every cluster created on a plan, enter your YAML configuration in the **(Optional) Add-ons - Use with caution** field in the pane for configuring a plan in the Tanzu Kubernetes Grid Integrated Edition tile.

For more information, see the *Plans* section of the *Installing Tanzu Kubernetes Grid Integrated Edition* topic for your IaaS. For example, [Plans](#) in *Installing Tanzu Kubernetes Grid Integrated Edition on vSphere*.

## Using Helm with Tanzu Kubernetes Grid Integrated Edition

This topic describes how to install the Helm package manager on Tanzu Kubernetes Grid Integrated

Edition (TKGI), and how to use Helm to install software to Kubernetes clusters deployed by TKGI.

## Overview

Helm is a package manager you can use to deploy TKGI components, such as Wavefront, and your TKGI Kubernetes apps.

Helm includes the following components:

| Component           | Role   | Location                                          |
|---------------------|--------|---------------------------------------------------|
| <code>helm</code>   | Client | Runs on your local workstation                    |
| <code>tiller</code> | Server | (Helm 2 only) Runs inside your Kubernetes cluster |

To install Helm, see [Install and Configure Helm](#).

To use Helm to deploy components or your apps to TKGI clusters, see [Deploy Apps and Components Using Helm](#).

## Install and Configure Helm

You can use either [Helm 3](#) or its predecessor [Helm 2](#) as your TKGI Helm package manager. Helm 3 is more easily installed than Helm 2 and requires less configuration.

To install and configure Helm 3 for TKGI, see [Install and Configure Helm 3](#) below. To install Helm 2, see [Install and Configure Helm 2](#) below.

### Install and Configure Helm 3

To install and configure Helm 3, follow the [Step 1: Install And Configure Helm](#) instructions in the Bitnami TKGI documentation.

### Install and Configure Helm 2

To use Helm 2 with TKGI, you must first configure the Tiller component to give it access to the Kubernetes API. Tiller runs inside the Kubernetes cluster.

To grant API access to Tiller and install Helm 2:

1. Create a role-based access control (RBAC) configuration file named `rbac-config.yaml` that contains the following:

```
apiVersion: v1
kind: ServiceAccount
metadata:
 name: tiller
 namespace: kube-system

apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
 name: tiller
roleRef:
 apiGroup: rbac.authorization.k8s.io
```

```

kind: ClusterRole
name: cluster-admin
subjects:
- kind: ServiceAccount
 name: tiller
 namespace: kube-system

```

2. Create the service account and role by running the following command:

```
kubectl create -f rbac-config.yaml
```

3. Download and install the latest v2 patch release of the [Helm CLI](#).
4. Deploy Helm 2 using the service account by running the following command:

```
helm init --service-account tiller
```

5. Verify that the permissions are configured by running the following command:

```
helm ls
```

There should be no output from the above command.

To apply more granular permissions to the Tiller service account, see the [Helm RBAC](#) documentation.

For more information about securing Helm 2, see the Bitnami article [Exploring the Security of Helm](#).

## Deploy Components and Apps Using Helm

You can use Helm to deploy third-party components or your own apps to TKGI clusters.

### Deploy Apps Listed in Artifact HUB

To deploy a third-party component on Artifact HUB to a TKGI cluster:

1. Download the component's Helm chart from the official repositories in the [Artifact HUB](#).
2. Complete the deployment instructions for the component.

For specific instructions on deploying Wavefront, which you might need to do for Windows worker-based clusters, see the [Wavefront](#) section of the *Monitoring Windows Worker Clusters and Nodes* topic.

### Deploy Your Own Apps

To deploy your app to a TKGI cluster using Helm:

1. Package the app as a *Helm chart*, the package format that the `helm install` command uses.
  - For example Helm charts, see [Concourse Helm Chart](#), [DataDog Helm Chart](#), or the charts archived in [Helm Charts](#) on GitHub.
  - For information on how to create a Helm chart for your app, see [Charts](#) in the Helm documentation.
2. Run the `helm install` command, passing in the location of your chart. For more information,

see the [Helm Docs](#).

# Logging and Monitoring Tanzu Kubernetes Grid Integrated Edition

This section describes how to monitor VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) Linux and Windows environments.

To monitor Linux clusters and workloads, see the following topics:

- [Monitoring TKGI and TKGI-Provisioned Clusters](#)
- [Monitoring Workers and Workloads](#)

To monitor Windows clusters and workloads, see:

- [Monitoring Windows Worker Clusters and Workers](#)
- [Logging Windows Worker Clusters and Workers](#)

## Monitoring TKGI and TKGI-Provisioned Clusters on Linux

This topic lists VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) components and integrations you can use to capture logs and metrics about TKGI and TKGI-provisioned cluster VMs on Linux.

### Overview

To monitor TKGI and TKGI-provisioned cluster VMs, you can enable one or more of the following components and integrations:

| Name                        | Link                                                           |
|-----------------------------|----------------------------------------------------------------|
| Syslog                      | See <a href="#">Syslog</a> below.                              |
| Telegraf (metrics)          | See <a href="#">Telegraf</a> below.                            |
| Healthwatch                 | See <a href="#">Healthwatch</a> below.                         |
| VMware vRealize Log Insight | See <a href="#">vRealize Log Insight (vSphere Only)</a> below. |

Syslog, Telegraf, and VMware vRealize Log Insight integrations are enabled in the **Tanzu Kubernetes Grid Integrated Edition** tile > **Host Monitoring** section. Healthwatch is deployed to Ops Manager as the Healthwatch Exporter for TKGI tile.

These components and integrations are visible only to TKGI admins. They are not visible to cluster users, such as developers.

For information about monitoring Kubernetes workloads on Linux, see [Monitoring Workers and](#)

## Workloads.

For information about logging and monitoring Kubernetes clusters, workers and workloads on Windows, see:

- [Monitoring Windows Worker Clusters and Workers](#)
- [Logging Windows Worker Clusters and Workers](#)

## Logs: Syslog and vRLI

You can configure Syslog or vRealize Log Insight (vSphere only) to publish logs from the TKGI control plane and TKGI-provisioned cluster VMs.

You might need to inspect Syslog or vRealize Log Insight (vRLI) logs when troubleshooting or auditing your TKGI environment. For information about key TKGI events and the log entries they generate, see [Auditing Tanzu Kubernetes Grid Integrated Edition Logs](#).

### Syslog

Syslog sends log messages from all BOSH-deployed VMs in a TKGI environment, including Kubernetes cluster audit logs, to a syslog endpoint. To configure Syslog, see [Syslog](#) in the *Installing* topic for your IaaS.

If you do not use Syslog, you can retrieve logs from BOSH-deployed VMs by downloading them as described in [Downloading Logs from VMs](#). However, retrieving these logs through Syslog is recommended.

### vRealize Log Insight (vSphere Only)

The vRealize Log Insight (vRLI) integration for TKGI pulls logs from all BOSH jobs and containers running in the cluster, including node logs from core Kubernetes and BOSH processes, Kubernetes event logs, and pod `stdout` and `stderr`.

To configure the vRLI integration, see [VMware vRealize Log Insight Integration](#) in the *Installing* topic for vSphere with Flannel or vSphere with NSX-T.

For information about vRLI, see [vRealize Log Insight](#).

## Metrics: Telegraf

Telegraf sends metrics from TKGI API, control plane node, and worker node VMs to a monitoring service, such as Wavefront or Datadog.

In the **Tanzu Kubernetes Grid Integrated Edition** tile > **Host Monitoring**, you can configure Telegraf to collect metrics from one or more the following sources:

| Source   | Includes metrics from...                                                       |
|----------|--------------------------------------------------------------------------------|
| TKGI API | <ul style="list-style-type: none"> <li>• Node Exporter (Prometheus)</li> </ul> |

---

|                                                              |                                                                                                                                                                                                   |
|--------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Control Plane nodes</b><br>(not visible to cluster users) | One or more of the following: <ul style="list-style-type: none"><li>• Node Exporter (Prometheus)</li><li>• Kubernetes API server</li><li>• Kubernetes controller manager</li><li>• etcd</li></ul> |
| <b>Worker nodes</b>                                          | One or more of the following: <ul style="list-style-type: none"><li>• Node Exporter (Prometheus)</li><li>• kubelet</li></ul>                                                                      |

---

To configure Telegraf, see [Configuring Telegraf in TKGI](#).

For more information about Node Exporter, see [About Node Exporter](#) below.

## About Node Exporter

Node Exporter exports hardware and operating system metrics in Prometheus format.

In the **Host Monitoring** pane of the Tanzu Kubernetes Grid Integrated Edition tile, you can enable the Node Exporter BOSH job separately on control plane nodes, worker nodes, and the TKGI API VM.

Node Exporter exposes metrics on *localhost* only. For a list of Node Exporter metrics, see the [Node Exporter GitHub repository](#).

## Healthwatch

You can use the [Healthwatch](#) Healthwatch Exporter for TKGI tile to monitor the health of the TKGI Control Plane and your Linux and Windows cluster control plane nodes.

Healthwatch enables you to monitor the functionality of your TKGI environment and can be configured to expose metrics to a service or database external to your Ops Manager foundation. For more information, see [Overview of the Healthwatch Exporter for TKGI Tile](#).

To configure cluster discovery in Healthwatch, see [Configuring TKGI Cluster Discovery](#) in the Healthwatch documentation.

## Auditing Tanzu Kubernetes Grid Integrated Edition Logs

This topic summarizes key auditable events in TKGI, and the content of the log entries that the events generate. Operators can use this information to audit event logs to see what users took what actions at what times. This is helpful for security, compliance, and troubleshooting.

Log content can either be [downloaded](#) or configured to be transported via syslog.

## TKGI API events

The following log entry examples are produced by TKGI API events and correspond to key actions taken by a user logged into the TKGI CLI.

## Cluster Creation

**create-cluster**

|                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>         | A user has issued a create cluster command.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>Identifying String</b>  | Action 'create-cluster'                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| <b>Example Log Entries</b> | <pre>2019-05-16 14:59:34.897 INFO 7594 - [nio-9021-exec-7] io.pivotal.pks.cluster.ClusterService : Action 'create-cluster' by user 'admin', cluster name: 'logs', plan name: 'small'. Details: class ClusterParameters { kubernetesMasterHost: logs.lathrop.cf-app.com kubernetesMasterPort: 8443 workerHaproxyIpAddresses: null kubernetesWorkerInstances: 3 authorizationMode: null nsxtNetworkProfile: null } 2019-05-16 14:59:34.911 INFO 7594 - [nio-9021-exec-7] io.pivotal.pks.telemetry.Agent : Telemetry - addCluster: cluster request: class ClusterRequest { name: logs planName: small networkProfileName: null parameters: class ClusterParameters { kubernetesMasterHost: logs.lathrop.cf- app.com kubernetesMasterPort: 8443 workerHaproxyIpAddresses: null kubernetesWorkerInstances: 3 authorizationMode: null nsxtNetworkProfile: null } }, cluster entity: ClusterEntity{name='logs', uuid='f4e2b775-8be3- 41b8-abe8-67f2265b957e', owner='admin', brokerOperationId='{"BoshTaskID":479,"BoshContextID":"256c3b65-2eae-48f7- 81f0-caed7472fa5f","OperationType":"create","PostDeployErrand": {}, "PreDeleteErrand":{}}, "Errands": [{"Name": "apply-addons", "Instances": null},  {"Name": "vrops-errand", "Instances": null}, {"Name": "telemetry- agent", "Instances": null} ] }', lastActionDescription='Creating cluster', planId='8A0E21A8-8072-4D80-B365-D1F502085560', lastAction='CREATE', lastActionState='in progress', masterIps='[In Progress]', parameters=io.pivotal.pks.cluster.data.ClusterParametersEntity@6efbedb6', networkProfileUuid=null', computeProfileUuid=null', taskStartedAt=2019-05- 16T14:59:34.804}, plan: class Plan { id: 8A0E21A8-8072-4D80-B365- D1F502085560 name: small description: Example: This plan will configure a lightweight kubernetes cluster. Not recommended for production workloads. workerInstances: 3 masterInstances: 1 allowPrivilegedContainers: false }</pre> |

## Cluster Deletion

**delete-cluster**

|                            |                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>         | A user has issued a delete cluster command.                                                                                                                                                                                                                                                                                                                     |
| <b>Identifying String</b>  | delete deployment for instance                                                                                                                                                                                                                                                                                                                                  |
| <b>Example Log Entries</b> | <pre>2019-06-04T14:16:52-06:00 10.0.10.10 broker/rs2 [on-demand-service-broker] [2f71a161-5755-4a0d-9c21-5b8405209594] 2019/06/04 20:16:52.493286 BOSH task ID 132 status: processing delete deployment for instance 67f77801-3d15-4d65- b501-38a643055e69: Description: delete deployment service-instance_67f77801- 3d15-4d65-b501-38a643055e69 Result:</pre> |

## Successful Login

**UserAuthenticationSuccess**

|                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>         | A user has successfully logged into Tanzu Kubernetes Grid Integrated Edition.                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>Identifying String</b>  | UserAuthenticationSuccess                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>Example Log Entries</b> | [2019-05-16 17:12:48.833] uaa - 7777 [https-jsse-nio-8443-exec-2] .... INFO - Audit: UserAuthenticationSuccess ('admin'): principal=0074aab6-6ff7-4b4c-b821-49526a96ebcb, origin=[remoteAddress=207.126.127.114, clientId=pks_cli], identityZoneId=[uaa] [2019-05-16 17:12:48.873] uaa - 7777 [https-jsse-nio-8443-exec-2] .... INFO - Audit: TokenIssuedEvent ('["pks.clusters.admin"]'): principal=0074aab6-6ff7-4b4c-b821-49526a96ebcb, origin=[client=pks_cli, user=admin], identityZoneId=[uaa] |

## Unsuccessful Login

**UserAuthenticationFailure**

|                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>         | A user has failed a login attempt into Tanzu Kubernetes Grid Integrated Edition.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Identifying String</b>  | UserAuthenticationFailure                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>Example Log Entries</b> | [2019-05-16 17:15:31.363] uaa - 7777 [https-jsse-nio-8443-exec-8] .... INFO - Audit: UserAuthenticationFailure ('admin'): principal=0074aab6-6ff7-4b4c-b821-49526a96ebcb, origin=[remoteAddress=207.126.127.114, clientId=pks_cli], identityZoneId=[uaa] [2019-05-16 17:15:31.371] uaa - 7777 [https-jsse-nio-8443-exec-8] .... INFO - Audit: PrincipalAuthenticationFailure ('null'): principal=admin, origin=[207.126.127.114], identityZoneId=[uaa] [2019-05-16 17:15:33.387] uaa - 7777 [https-jsse-nio-8443-exec-6] .... INFO - Audit: ClientAuthenticationSuccess ('Client authentication success'): principal=pks_client, origin=[remoteAddress=127.0.0.1, cl |

## Successful Cluster Credential Retrieval

**ClientAuthenticationSuccess**

|                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>         | A user has successfully gained access to a cluster in Tanzu Kubernetes Grid Integrated Edition.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Identifying String</b>  | ClientAuthenticationSuccess                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Example Log Entries</b> | [2019-05-16 17:15:31.363] uaa - 7777 [https-jsse-nio-8443-exec-8] .... INFO - Audit: UserAuthenticationFailure ('admin'): principal=0074aab6-6ff7-4b4c-b821-49526a96ebcb, origin=[remoteAddress=207.126.127.114, clientId=pks_cli], identityZoneId=[uaa] [2019-05-16 17:15:31.371] uaa - 7777 [https-jsse-nio-8443-exec-8] .... INFO - Audit: PrincipalAuthenticationFailure ('null'): principal=admin, origin=[207.126.127.114], identityZoneId=[uaa] [2019-05-16 17:15:33.387] uaa - 7777 [https-jsse-nio-8443-exec-6] .... INFO - Audit: ClientAuthenticationSuccess ('Client authentication success'): principal=pks_client, origin=[remoteAddress=127.0.0.1, cl |

## User Creation

**UserCreatedEvent**

|                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>         | An administrator has successfully created a new user for Tanzu Kubernetes Grid Integrated Edition.                                                                                                                                                                                                                                                                                                                                                          |
| <b>Identifying String</b>  | UserCreatedEvent                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>Example Log Entries</b> | Jun 04 16:00:07 10.0.10.10 uaa/rs2: [2019-06-04 22:00:07.293] uaa - 18840 [https-jsse-nio-8443-exec-6] .... INFO - Audit: UserCreatedEvent ('["user_id=dc803130-15dc-4279-8b42-868fc80b8ca1","username=USERNAME2"]'): principal=dc803130-15dc-4279-8b42-868fc80b8ca1, origin=[client=admin, details=(remoteAddress=35.192.67.34, tokenType=bearertokenValue=, sub=admin, iss=https://api.tkgi.hawthorne.cf-app.com:8443/oauth/token)], identityZoneId=[uaa] |

## User Deletion

**UserDeletedEvent**

|                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>         | An administrator has successfully deleted a user for Tanzu Kubernetes Grid Integrated Edition.                                                                                                                                                                                                                                                                                                                                                              |
| <b>Identifying String</b>  | UserDeletedEvent                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>Example Log Entries</b> | Jun 04 16:00:07 10.0.10.10 uaa/rs2: [2019-06-04 22:00:07.293] uaa - 18840 [https-jsse-nio-8443-exec-6] .... INFO - Audit: UserCreatedEvent ('["user_id=dc803130-15dc-4279-8b42-868fc80b8ca1","username=USERNAME2"]'): principal=dc803130-15dc-4279-8b42-868fc80b8ca1, origin=[client=admin, details=(remoteAddress=35.192.67.34, tokenType=bearertokenValue=, sub=admin, iss=https://api.tkgi.hawthorne.cf-app.com:8443/oauth/token)], identityZoneId=[uaa] |

## Telemetry Collection

**Telemetry Ping**

|                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>         | The optional telemetry system has successfully reached an external host for collecting product data for Tanzu Kubernetes Grid Integrated Edition.                                                                                                                                                                                                                                                                                                                                                    |
|                            | To learn more about the Tanzu Kubernetes Grid Integrated Edition telemetry program, see <a href="#">Telemetry</a> .                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Identifying String</b>  | telemetry-server                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Example Log Entries</b> | 2019-06-04T15:41:05-06:00 10.0.10.10 telemetry-server/rs2 2019-06-04 21:41:05 +0000 [debug]: #0 generating helo 2019-06-04T15:41:05-06:00 10.0.10.10 telemetry-server/rs2 2019-06-04 21:41:05 +0000 [debug]: #0 checking ping 2019-06-04T15:41:05-06:00 10.0.10.10 telemetry-server/rs2 2019-06-04 21:41:05 +0000 [debug]: #0 generating pong 2019-06-04T15:41:05-06:00 10.0.10.10 telemetry-server/rs2 2019-06-04 21:41:05 +0000 [debug]: #0 connection established address="10.0.11.21" port=33366 |

## Kubernetes Audit Log Events

The Kubernetes control plane emits a standard log format every time a user takes action to query or change the state of the Kubernetes API. An example audit event log entry is below.

```
{
 "kind": "Event",
 "apiVersion": "audit.k8s.io/v1",
 "level": "Request",
 "auditID": "dc2bb4e9-4b85-42da-82a3-5ee47091207d",
 "stage": "ResponseStarted",
 "requestURI": "/apis/policy/v1/poddisruptionbudgets?resourceVersion=370506\u0026timeout=7m54s\u0026timeoutSeconds=474\u0026watch=true",
 "verb": "watch",
 "user": {
 "username": "system:kube-scheduler",
 "uid": "system:kube-scheduler",
 "groups": ["system:authenticated"]
 },
 "sourceIPs": ["10.0.11.10"],
 "userAgent": "kube-scheduler/v1.15.4 (linux/amd64) kubernetes/67d2fcf/schedule",
 "objectRef": {
 "resource": "poddisruptionbudgets",
 "apiGroup": "policy",
 "apiVersion": "v1"
 },
 "responseStatus": {
 "metadata": {},
 "code": 200
 },
 "requestReceivedTimestamp": "2019-12-11T21:47:28.097065Z",
 "stageTimestamp": "2019-12-11T21:47:28.097491Z",
 "annotations": {
 "authorization.k8s.io/decision": "allow",
 "authorization.k8s.io/reason": "RBAC: allowed by ClusterRoleBinding \"system:kube-scheduler\" of ClusterRole \"system:kube-scheduler\" to User \"system:kube-scheduler\""
 }
}
```

For more information about Kubernetes Audit Event Log format see the [Kubernetes documentation](#).

## Related Links

- For information about configuring syslog log transport, see [Installing Tanzu Kubernetes Grid Integrated Edition](#).
- For information about downloading TKGI logs, see [Downloading Logs from VMs](#).
- For information about Kubernetes Audit Log format, see [Kubernetes documentation](#)

## Downloading Logs from VMs

This topic explains how to download logs from BOSH-deployed VMs in your VMware Tanzu Kubernetes Grid Integrated Edition environment using the BOSH Command Line Interface (CLI).

## Overview

In Tanzu Kubernetes Grid Integrated Edition, you can download logs from any BOSH-deployed VM, such as the TKGI API VM or Kubernetes cluster VMs.

You might need to download these logs when troubleshooting or auditing your TKGI environment.

## Download Logs

To download logs from a BOSH-deployed VM:

1. Gather credential and IP address information for your BOSH Director, SSH into the Ops Manager VM, and use the BOSH CLI to log in to the BOSH Director from the Ops Manager VM. For more information, see [Advanced Troubleshooting with the BOSH CLI](#).
2. After logging in to the BOSH Director, list the names of your BOSH deployments by running:

```
bosh -e ENVIRONMENT deployments
```

Where `ENVIRONMENT` is your BOSH environment alias.

3. Identify the names of the VMs that you want to retrieve logs from by listing the VMs in your target BOSH deployment:

```
bosh -e ENVIRONMENT -d DEPLOYMENT vms
```

Where:

- ◊ `ENVIRONMENT` is the BOSH environment alias.
- ◊ `DEPLOYMENT` is your target BOSH deployment name. Kubernetes cluster deployment names begin with `service-instance_` and include a unique identifier.

For example, the following command lists the VMs in a Kubernetes cluster:

```
$ bosh -e tkgi -d service-instance_ae681cd1-7ff4-4661-b12c-49a5b543f16f vms
```

4. Download logs from a VM:

```
bosh -e ENVIRONMENT -d DEPLOYMENT logs VM-NAME
```

For example:

```
$ bosh -e tkgi
-d service-instance_ae681cd1-7ff4-4661-b12c-49a5b543f16f
logs master/000a1111-222b-3333-4cc5-de66f7a8899b
```

For more information about log files, see [View Log Files](#) in *Using BOSH Diagnostic Commands in Tanzu Kubernetes Grid Integrated Edition*.

# Configuring Telegraf in TKGI

This topic describes how to configure Telegraf in VMware Tanzu Kubernetes Grid Integrated Edition (TKGI).

## Overview

You can configure Telegraf to collect metrics from TKGI API, control plane node, and worker node VMs and send the metrics to a monitoring service, such as Wavefront or Datadog.

For more information about these metrics, see [Metrics: Telegraf](#) in *Monitoring TKGI and TKGI-Provisioned Clusters*.

## Collect Metrics Using Telegraf

To collect metrics using Telegraf:

1. Create a configuration file for your output plugin. See [Create a Configuration File](#) below.
2. Configure Telegraf in the Tanzu Kubernetes Grid Integrated Edition tile. See [Configure Telegraf in the Tile](#) below.

## Create a Configuration File

To connect a monitoring service to TKGI, you must create a configuration file for the service. The configuration file is written in a TOML format and consists of key-value pairs. After you create your configuration file, you can enter the file into the Tanzu Kubernetes Grid Integrated Edition tile to connect the service.

To create a configuration file for your monitoring service:

1. Locate the required format for your monitoring service in the [README.md](#) file for your service in [telegraf](#) in GitHub. For example, if you want to collect metrics from etcd, the etcd documentation recommends using the open-source Prometheus monitoring service.
2. Create your configuration file using the required format of your monitoring service. For example, if you want to create a configuration file for an HTTP output plugin, create a file similar to the following:

```
[[outputs.http]]
 url="https://example.com"
 method="POST"
 data_format="json"
 [[processors.override]]
 [processors.override.tags]
 director = "bosh-director-1"
```



**Note:** You can add tags to your configuration file to label etcd metrics. For example, the above code snippet adds a `bosh-director-1` tag to the etcd metrics. If you have multiple BOSH Directors, VMware recommends adding tags to filter your metrics in your monitoring service.

## Configure Telegraf in the Tile

To configure TKGI to use Telegraf for metric collection:

1. Navigate to the **Tanzu Kubernetes Grid Integrated Edition** tile > **Settings** > **Host Monitoring**.

2. Under **Enable Telegraf Outputs?**, select **Yes**.

Enable Telegraf Outputs?\*

No

Yes

Prometheus input plugin Metric version\*

Prometheus input plugin Metric version = 2 ▾

Enable node exporter on TKGI API \*

Enable node exporter on master \*

Include etcd metrics \*

Enable node exporter on worker \*

Include Kubernetes Controller Manager metrics \*

Include Kubernetes API Server metrics

\*

Include Kubernetes Scheduler metrics

\*

Include kubelet metrics \*

Include telegraf metrics when telegraf enabled \*

Setup Telegraf Outputs \*

[[outputs.discard]]



3. Configure Telegraf output settings as described in the table below.

| Configuration Setting                                | Description...to send these metrics to your monitoring service                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Prometheus input plugin Metric version</b>        | Controls the metrics mapping from Prometheus to telegraf when scraping metrics using the Prometheus input plugin. The Prometheus input plugin scrapes the following metrics: <code>node_exporter</code> , <code>kube_apiserver</code> , <code>kube_controller_manager</code> , <code>kube_scheduler</code> , and <code>etcd metrics</code> .<br><br>Your Prometheus client must be configured with the matching <code>metric_version</code> setting. For more information, see <a href="#">Prometheus Input Plugin</a> in the telegraf GitHub repository. |
| <b>Enable node exporter on TKGI API</b>              | Enable to send Node Exporter metrics from the TKGI API VM.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Enable node exporter on control plane</b>         | Enable to send Node Exporter metrics from Kubernetes control plane nodes.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Include etcd metrics</b>                          | Enable to send etcd server and debugging metrics.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>Enable node exporter on worker</b>                | Enable to send Node Exporter metrics from Kubernetes worker nodes.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>Include Kubernetes Controller Manager metrics</b> | Enable to send Kubernetes controller manager metrics.<br>These metrics provide information about the state of each cluster.                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>Include Kubernetes API Server metrics</b>         | Enable to send Kubernetes API Server metrics.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>Include Kubernetes Scheduler metrics</b>          | Enable to send Kubernetes Scheduler metrics. For more information, see <a href="#">Configure Include Kubernetes Scheduler Metrics</a> .                                                                                                                                                                                                                                                                                                                                                                                                                   |
| <b>Include kubelet metrics</b>                       | Enable to send kubelet metrics for all workloads running in all your Kubernetes clusters.<br>If you enable <b>Include kubelet metrics</b> , be prepared for a high volume of metrics.                                                                                                                                                                                                                                                                                                                                                                     |

|                                                       |                                                                                                                                                                                              |
|-------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Include telegraf metrics when telegraf enabled</b> | Enable to send telegraf process memory status, agent metrics, and write metrics. For more information, see <a href="#">Telegraf Internal Input Plugin</a> in the telegraf GitHub repository. |
|-------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|



**Note:** The telegraf output configuration options are visible to TKGI admins only.

Components you enable in this step will be visible to TKGI admins only.

- In **Setup Telegraf Outputs**, replace the default value `[[outputs.discard]]` with the contents of the configuration file that you created in [Create a Configuration File](#) above. See the following example for an HTTP output plugin:

```
[outputs.http]
url="https://example.com"
method="POST"
data_format="json"
[[processors.override]]
[processors.override.tags]
director = "bosch-director-1"
```

- In **Setup Telegraf Agent**, replace the default Telegraf agent property values with your custom values for interval, buffering and debugging related properties. For more information about the configurable Telegraf agent properties, see [Agent configuration](#) in the Telegraf documentation.
- Click **Save**.
- To deploy the Tanzu Kubernetes Grid Integrated Edition tile, return to the Ops Manager Installation Dashboard and click **Review Pending Changes > Apply Changes**.

## Troubleshoot etcd

VMware recommends working with Support to troubleshoot control plane/etcdb node VMs. The monitoring and metrics data you retrieve from the control plane/etcdb node VMs can help the Support team diagnose and troubleshoot errors.

## Monitoring Linux Workers and Workloads

This topic lists VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) components and integrations you can use to capture logs and metrics about your Kubernetes worker nodes and workloads.

### Overview

To monitor Kubernetes worker nodes and workloads in your TKGI deployment, you can enable one or more of the following components and integrations in the **Tanzu Kubernetes Grid Integrated Edition** tile > **In-Cluster Monitoring**:

| Name                | Type                                | Link                                                                                                                                                                                 |
|---------------------|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Sink resources      | TKGI component                      | Sink Resources, below.                                                                                                                                                               |
| Wavefront           | External integration                | <a href="#">VMware Tanzu Kubernetes Grid Integrated Edition Integration</a> in the Wavefront documentation.<br><br>Or send metrics to Wavefront with a <a href="#">metric sink</a> . |
| vRealize Operations | External integration, with cAdvisor | <a href="#">cAdvisor on GitHub</a><br><a href="#">VMware vRealize Operations Management Pack for Container Monitoring</a>                                                            |

When running on worker nodes, these components and integrations are visible to both TKGI admins and cluster users, such as developers.

To enable sink resources, Wavefront, or vRealize Operations integration, follow the instructions in *In-Cluster Monitoring* for your IaaS:

- [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#)
- [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T](#)
- [Installing Tanzu Kubernetes Grid Integrated Edition on GCP](#)
- [Installing Tanzu Kubernetes Grid Integrated Edition on AWS](#)
- [Installing Tanzu Kubernetes Grid Integrated Edition on Azure](#)

For information about monitoring TKGI and TKGI-provisioned cluster VMs on Linux, see [Monitoring TKGI and TKGI-Provisioned Clusters](#).

For information about logging and monitoring Kubernetes clusters, workers and workloads on Windows, see:

- [Monitoring Windows Worker Clusters and Workers](#)
- [Logging Windows Worker Clusters and Workers](#)

## Sink Resources

In TKGI, you can deploy log sinks and metric sinks to monitor your Kubernetes worker nodes and workloads that are running on them.

To deploy a log or a metric sink:

1. Enable sink resources in the **Tanzu Kubernetes Grid Integrated Edition** tile > **In-Cluster Monitoring**. You can enable both log and metric sink resources or only one of them.
2. (Optional) Enable Node Exporter on worker nodes by selecting the **Enable node exporter on workers** checkbox.
3. Create sink resources. For instructions, see [Creating and Managing Sink Resources](#).

For more information about sink resources, see:

- Conceptual information: [Sink Architecture in Tanzu Kubernetes Grid Integrated Edition](#)
- Sink resource types, outputs, and identifying strings: [Monitoring Clusters with Log Sinks](#)

# Sink Architecture in Tanzu Kubernetes Grid Integrated Edition

This topic describes how VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) implements sinks for collecting logs and metrics from Kubernetes worker nodes and workloads.

For step-by-step instructions on creating sinks in TKGI, see [Creating and Managing Sink Resources](#).

## Overview

A sink collects logs or metrics about Kubernetes worker nodes in a TKGI deployment and workloads that are running on them.

For more information, see:

- [Sink Types](#) below
- [Sink Architecture](#) below

## Sink Types

You can create two types of sinks:

- Log sinks
- Metric sinks

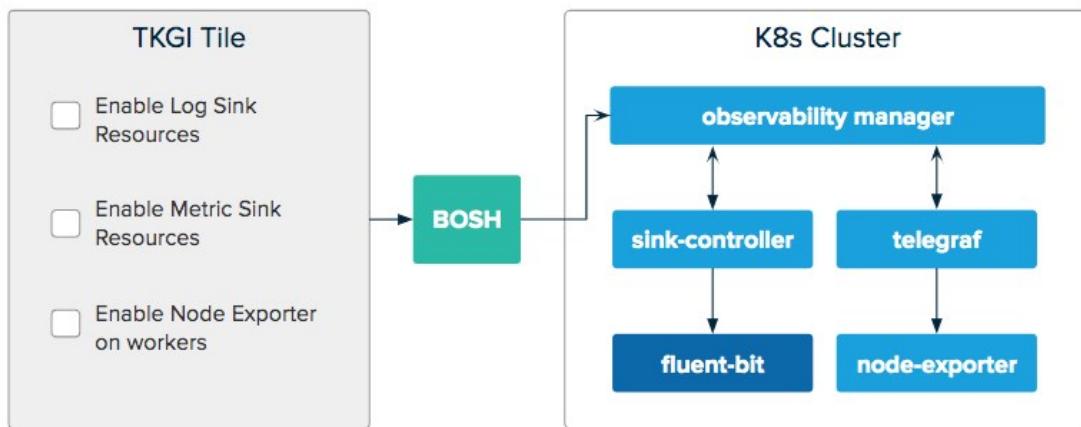
See the table below for information about these sink types.

| Sink Type   | Sink Resource                  | Description                                                                                                                                                                                                                                                                                                                     |
|-------------|--------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Log sink    | <code>ClusterLogSink</code>    | Forwards logs from a cluster to a log destination. Logs are transported using one of the following: <ul style="list-style-type: none"> <li>• The Syslog Protocol defined in <a href="#">RFC 5424</a></li> <li>• WebHook</li> <li>• Fluent Bit output plugins</li> </ul>                                                         |
| Log sink    | <code>LogSink</code>           | Forwards logs from a namespaced subset within a <code>ClusterLogSink</code> resource to a log destination. Logs are transported using one of the following: <ul style="list-style-type: none"> <li>• The Syslog Protocol defined in <a href="#">RFC 5424</a></li> <li>• WebHook</li> <li>• Fluent Bit output plugins</li> </ul> |
| Metric sink | <code>ClusterMetricSink</code> | Collects and writes metrics from a cluster to specified outputs using input and output plugins.                                                                                                                                                                                                                                 |
| Metric sink | <code>MetricSink</code>        | Collects and writes metrics from a namespace within a cluster to specified outputs using input and output plugins.                                                                                                                                                                                                              |

## Sink Architecture

TKGI-provisioned Kubernetes clusters include an observability manager that manages log sink and metric sink configurations within a cluster.

The following diagram details TKGI cluster observability architecture:



[View a larger version of this image.](#)

In the **Tanzu Kubernetes Grid Integrated Edition** tile > **In-Cluster Monitoring**:

- **Enable Metric Sink Resources** enables metric sinks.
- **Enable Log Sink Resources** enables log sinks.
- **Enable node exporter on workers** forwards additional infrastructure metrics.

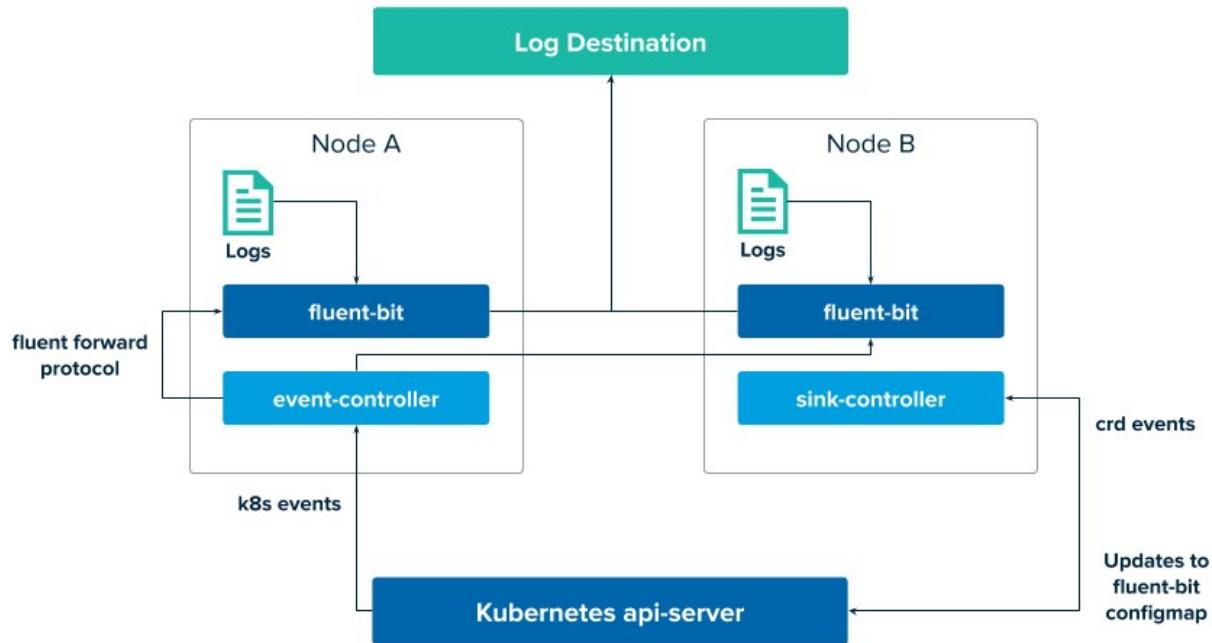
Setting these checkboxes in Ops Manager directs how BOSH configures the observability manager.

For more information about enabling log sinks and metrics sinks, see [\(Optional\) In-Cluster Monitoring](#) in the *Installing* topic for your IaaS.

## Log Sink Architecture

The TKGI log sink aggregates workload logs and forwards them to a common log destination.

The following diagram details TKGI log sink architecture:



[View a larger version of this image.](#)

Logs are monitored and aggregated by a Fluent Bit [DaemonSet](#) running as a pod on each worker node.

An event-controller collects Kubernetes API events and sends them to a second Fluent Bit daemon pod for aggregation.

All aggregated log entries are marshaled to a common log destination.

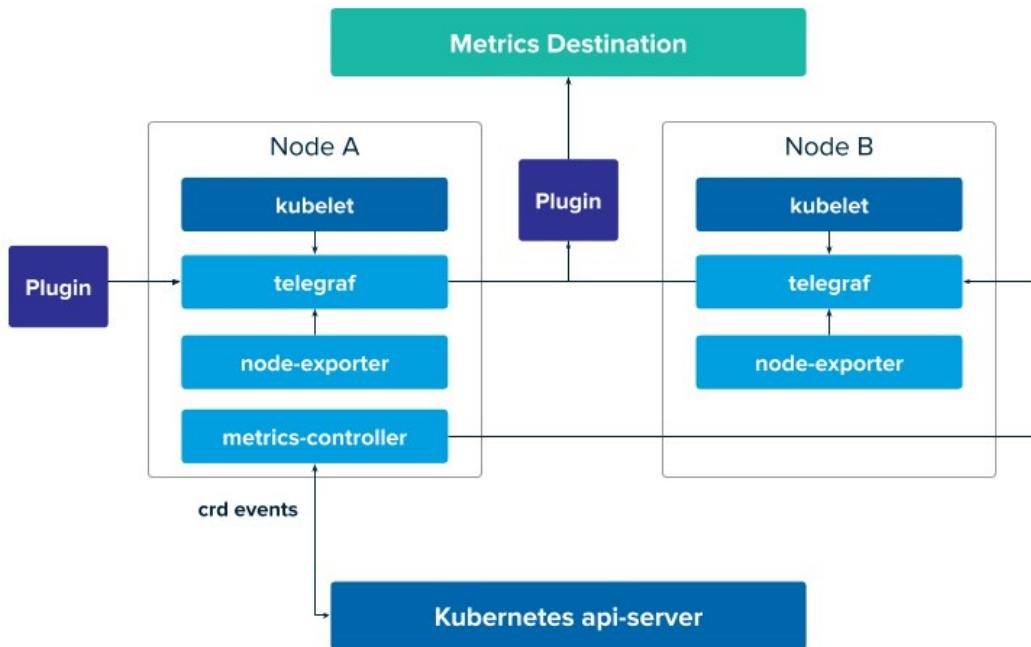


**Note:** When sinks are added or removed, all of the Fluent Bit pods are refreshed with new sink information.

## Metric Sink Architecture

The TKGI metric sink aggregates workload metrics and forwards them to a common metrics destination.

The following diagram details TKGI metric sink architecture:



[View a larger version of this image.](#)

A metric sink collects and writes metrics from a cluster to specified outputs using input and output plugins.

Workload metrics are monitored by a set of third-party plugins. The plugins forward the metrics to a Telegraf service pod.

A pair of kubelets monitors Kubernetes and forwards Kubernetes metrics to a pair of Telegraf service pods.

If Node Exporter is enabled on the worker nodes in the Tanzu Kubernetes Grid Integrated Edition tile, a Node Exporter [DaemonSet](#) is included in all clusters. For more information about Node Exporter metrics, see the [Node Exporter](#) repository in GitHub.

To define the collected unstructured metrics, a metric-controller monitors Kubernetes for custom resource definitions and forwards those definitions to the Telegraf services.

The Telegraf services collect, process, and aggregate gathered metrics. All aggregated metrics are marshaled to an additional plugin for forwarding to a third-party application.



**Note:** When sinks are added or removed, all of the Telegraf pods are refreshed with new sink information.

## Creating and Managing Sink Resources

This topic describes how to create and manage sink resources for a Kubernetes cluster provisioned with VMware Tanzu Kubernetes Grid Integrated Edition (TKGI), or for a namespace within a cluster.

### Overview

Sinks collect logs and metrics about Kubernetes worker nodes in your TKGI deployment and workloads that are running on them.

You can create two types of sinks:

- Log sinks
- Metric sinks

For more conceptual information about sinks, see [Sink Architecture in Tanzu Kubernetes Grid Integrated Edition](#).

## Prerequisites

Before creating a sink resource:

1. Review [Sink Types](#) in *Sink Architecture in Tanzu Kubernetes Grid Integrated Edition*.
2. Configure sink resources in the **Tanzu Kubernetes Grid Integrated Edition** tile > **In-Cluster Monitoring**:
  - If you want to create a `ClusterLogSink` or `LogSink` resource, select the **Enable Log Sink Resources** checkbox.
  - If you want to create a `ClusterMetricSink` or `MetricSink` resource, select the **Enable Metric Sink Resources** checkbox.
  - If you want to use Node Exporter to send worker node metrics to metric sinks of kind `ClusterMetricSink` as described in [Create a ClusterMetricSink Resource for Node Exporter Metrics](#) below, select the **Enable node exporter on workers** checkbox.

For more information about these configuration settings, see the TKGI installation topic for your IaaS:

- [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#)
  - [Installing Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T Integration](#)
  - [Installing Tanzu Kubernetes Grid Integrated Edition on GCP](#)
  - [Installing Tanzu Kubernetes Grid Integrated Edition on AWS](#)
  - [Installing Tanzu Kubernetes Grid Integrated Edition on Azure](#)
3. Install the Kubernetes CLI, `kubectl`. For installation instructions, see [Installing the Kubernetes CLI](#).

## Create LogSink or ClusterLogSink Resources

To create `ClusterLogSink` or `LogSink` resources, you can:

- [Create a Syslog ClusterLogSink or LogSink Resource](#)
- [Create a Webhook ClusterLogSink or LogSink Resource](#)
- [Create a ClusterLogSink or LogSink Resource with a Fluent Bit Output Plugin](#)



**Note:** Log sinks created in TKGI do not support UDP connections.



**Note:** TKGI requires a secure connection for log forwarding when using `ClusterLogSink` and `LogSink` resources of type `syslog` or `webhook`. To forward logs using an unsecured connection, see [Unsecured ClusterLogSink and LogSink Log Forwarding](#) below.

## Create a Syslog ClusterLogSink or LogSink Resource

`ClusterLogSink` and `LogSink` resources of type `syslog` deliver logs using the TCP-based syslog protocol.

By default, TKGI uses a system root certificate authority (CA) certificate to secure `syslog` `ClusterLogSink` and `LogSink` log forwarding connections, but you can optionally use a custom CA certificate to secure the connections.

To define a `syslog` `ClusterLogSink` or `LogSink` resource, perform the following steps:

1. Create a YAML file that specifies your log destination in one of the following formats:
  - ◊ To use the default system root CA certificate to secure log forwarding connections:

```
apiVersion: pksapi.io/v1beta1
kind: YOUR-SINK-RESOURCE
metadata:
 name: YOUR-SINK
 namespace: YOUR-NAMESPACE
spec:
 type: syslog
 host: YOUR-LOG-DESTINATION
 port: YOUR-LOG-DESTINATION-PORT
 enable_tls: true
 insecure_skip_verify: false
```

Where:

- `YOUR-SINK-RESOURCE` is the sink resource that you want to create. This must be either `ClusterLogSink` or `LogSink`. For information about these sink resources, see [Overview](#).
- `YOUR-SINK` is a name that you choose for your sink.
- `YOUR-NAMESPACE` is the name of your namespace. Omit this line if you are creating `ClusterLogSink`.
- `YOUR-LOG-DESTINATION` is the URL or IP address of your log management service.
- `YOUR-LOG-DESTINATION-PORT` is the port number of your log management service.



**Note:** `enable_tls` must be `true`.

- To use a custom CA certificate to secure log forwarding connections:

```
apiVersion: pksapi.io/v1beta1
kind: YOUR-SINK-RESOURCE
metadata:
 name: YOUR-SINK
 namespace: YOUR-NAMESPACE
spec:
 type: syslog
 host: YOUR-LOG-DESTINATION
 port: YOUR-LOG-DESTINATION-PORT
 enable_tls: true
 insecure_skip_verify: false
 fluent_bit_ca_cert: YOUR-CA-CERT
```

Where:

- **YOUR-SINK-RESOURCE** is the sink resource that you want to create. This must be either `ClusterLogSink` or `LogSink`. For information about these sink resources, see [Overview](#).
- **YOUR-SINK** is a name that you choose for your sink.
- **YOUR-NAMESPACE** is the name of your namespace. Omit this line if you are creating `ClusterLogSink`.
- **YOUR-LOG-DESTINATION** is the URL or IP address of your log management service.
- **YOUR-LOG-DESTINATION-PORT** is the port number of your log management service.
- **YOUR-CA-CERT** is your custom CA certificate to secure the `syslog` connection. The custom CA certificate must include a `SAN` field. If the certificate does not include a `SAN` field, Fluent Bit will not send logs.



**Note:** `enable_tls` must be `true`.

2. (Optional) To filter the logging output, include a `filters` section in your configuration. For more information, see [Define a Filter for LogSink and ClusterLogSink Resources](#) below.
3. Save the YAML file with an appropriate file name. For example, `my-cluster-log-sink.yml`.
4. Apply the `ClusterLogSink` or `LogSink` resource to your cluster by running the following command:

```
kubectl apply -f YOUR-SINK.yml
```

Where `YOUR-SINK.yml` is the name of your YAML file.

For example:

```
$ kubectl apply -f my-cluster-log-sink.yml
```

## Create a Webhook ClusterLogSink or LogSink Resource

`ClusterLogSink` and `LogSink` resources of type `webhook` batch logs into one-second units, wrap the resulting payload in JSON, and use the `POST` method to deliver the logs to the address of your log management service.

To define a webhook `ClusterLogSink` or `LogSink` resource, perform the following steps:

1. Create a YAML file that specifies your log destination in the following format:

```
apiVersion: pksapi.io/v1beta1
kind: YOUR-SINK-RESOURCE
metadata:
 name: YOUR-SINK
 namespace: YOUR-NAMESPACE
spec:
 type: webhook
 url: YOUR-LOG-DESTINATION
```

Where:

- `YOUR-SINK-RESOURCE` is the sink resource you want to create. This must be either `ClusterLogSink` or `LogSink`. For information about these sink resources, see [Overview](#).
  - `YOUR-SINK` is a name you choose for your sink.
  - `YOUR-NAMESPACE` is the name of your namespace. Omit this line if you are creating `ClusterLogSink`.
  - `YOUR-LOG-DESTINATION` is the URL or IP address of your log management service.
2. (Optional) To filter the logging output, include a `filters` section in your configuration. For more information, see [Define a Filter for LogSink and ClusterLogSink Resources](#) below.
  3. Save the YAML file with an appropriate filename. For example, `my-cluster-log-sink.yml`.
  4. Apply the `ClusterLogSink` or `LogSink` resource to your cluster by running the following command:

```
kubectl apply -f YOUR-SINK.yml
```

Where `YOUR-SINK.yml` is the name of your YAML file.

For example:

```
$ kubectl apply -f my-cluster-log-sink.yml
```

## Create a ClusterLogSink or LogSink Resource with a Fluent Bit Output Plugin

`ClusterLogSink` and `LogSink` resources with a Fluent Bit output plugin deliver logs to the output

plugin that you specify in your resource configuration.

To define a `ClusterLogSink` or `LogSink` resource with a Fluent Bit output plugin, perform the following steps:

1. Create a YAML file that specifies your log destination in the following format:

```
apiVersion: pksapi.io/v1beta1
kind: YOUR-SINK-RESOURCE
metadata:
 name: YOUR-SINK
 namespace: YOUR-NAMESPACE
spec:
 type: http
 output_properties:
 Host: example.com
 Format: json
 Port: 443
 tls: on
 tls.verify: off
```

Where:

- `YOUR-SINK-RESOURCE` is the sink resource you want to create. This must be either `ClusterLogSink` or `LogSink`. For information about these sink resources, see [Overview](#).
- `YOUR-SINK` is a name you choose for your log sink.
- `YOUR-NAMESPACE` is the name of your namespace. Omit this line if you are creating `ClusterLogSink`.



**Note:** This is a sample plugin configuration for `http`. For a full list of supported plugins, see the [Fluent Bit documentation](#).

2. (Optional) To filter the logging output, include a `filters` section in your configuration. For more information, see [Define a Filter for LogSink and ClusterLogSink Resources](#) below.
3. Save the YAML file with an appropriate filename. For example, `my-cluster-log-sink.yml`.
4. Apply the `ClusterLogSink` or `LogSink` resource to your cluster by running the following command:

```
kubectl apply -f YOUR-SINK.yml
```

Where `YOUR-SINK.yml` is the name of your YAML file.

For example:

```
$ kubectl apply -f my-cluster-log-sink.yml
```

## (Optional) Define a Filter for LogSink and ClusterLogSink Resources

You can set filters on your [LogSink](#) and [ClusterLogSink](#) resources:

- You can include or exclude all logs or all events from sink output. For more information, see [Exclude Logs or Events from Sink Output](#) below.
- If you are using Fluent Bit, you can filter logs using conditions and rules. For more information, see [Create a Fluent Bit ClusterLogSink or LogSink Filter](#) below.

## Exclude Logs or Events from Sink Output

The [LogSink](#) and [ClusterLogSink](#) resources allow users to set filters to include or exclude all logs or all events from sink output.

To filter all logs or all events from sink output:

1. Add a filter properties section to the YAML file that specifies the sink's log output destination and which types of log entities to include or exclude from the output:

```
apiVersion: pksapi.io/v1beta1
kind: YOUR-SINK-RESOURCE
metadata:
 name: YOUR-SINK
 namespace: YOUR-NAMESPACE
spec:
 type: syslog
 host: YOUR-LOG-DESTINATION
 port: YOUR-LOG-DESTINATION-PORT
 enable_tls: true
 filters:
 include-events: true
 include-logs: false
```

Where:

- [YOUR-SINK-RESOURCE](#) is the sink resource type that you created. This must be either [ClusterLogSink](#) or [LogSink](#).
- [YOUR-SINK](#) is the name you chose for your sink.
- [YOUR-NAMESPACE](#) is the name of your namespace. Omit this line for [ClusterLogSink](#) type sink resources.
- [YOUR-LOG-DESTINATION](#) is the URL or IP address of your log management service.
- [YOUR-LOG-DESTINATION-PORT](#) is the port number of your log management service.

The default values for these filter properties is **true**. If you do not specify [filters](#) properties, both logs and events are included in the sink's output.

For more information, see [Monitoring Clusters with Log Sinks](#).

## Fluent Bit ClusterLogSink or LogSink FilterSpecs

As you consider your sink, you might realize that the sink needs a filter more complex than “all logs”

or “all events” or even more complex than a single query. You might even need to route your log output to more than one destination target.

If you are using a Fluent Bit ClusterLogSink or LogSink sink resource, you can define a filterSpec with condition rules for filtering incoming logs. Your Fluent Bit sink resource will selectively direct matching log entries to the output destination for the sink. The output target for a Fluent Bit sink is the [Output\\_properties](#) destination defined for the sink resource.

For example, you might want only the error log entries of your [Production](#) Pods routed to a specific service that you are monitoring and for the remaining [Production](#) Pod status log entries to be managed separately from your non-production Pod logs. For this scenario, you require the ability to create a complex filtering query and to control the routing destination of filtered log entries.

### Support for Multiple Filtering Rules

You can define multiple filtering rules in a filterSpec:

- Define multiple condition tests in a filterSpec filter:
  - Define filter conditions that filter based on log content.
  - Define filter conditions that filter based on the log metadata. You can filter log metadata by: Namespace, Host Name, Container Name, Pod Name, and Pod labels.

When you define multiple condition tests in a filterSpec filter, a log entry is considered a match only if all of the defined conditions in the filter are matched.

- Define multiple filters for a filterSpec:

When you define multiple filters in your filterSpec configuration, a log entry is considered a match if one or more of the filters is matched.

### Support for Multiple Target Destinations

If you have multiple output target destinations, you need to define a separate sink for each destination. When you have multiple sinks, log entry output is directed to only one destination, no matter how many sinks the log entry is a match for.

Which destination a log entry is routed to depends on which sinks the entry matches:

- If one or more of the sinks is configured without a filterSpec, all log entries are directed to the output target of the first sink without a filterSpec.
- If all of the sinks are configured with a filterSpec, the log entries are directed to the output target as follows:

| Log Entry Match Scenario                  | Log Entry Target Destination                                                                          |
|-------------------------------------------|-------------------------------------------------------------------------------------------------------|
| Matches a single sink                     | The log entry is directed to the output target destination for the sink.                              |
| Matches multiple LogSinks                 | The log entry is directed to only the output target destination for the first matched LogSink.        |
| Matches multiple ClusterLogSinks          | The log entry is directed to only the output target destination for the first matched ClusterLogSink. |
| Matches both LogSinks and ClusterLogSinks | The log entry is directed to only the output target destination for the first matched LogSink.        |

## Create a Fluent Bit ClusterLogSink or LogSink FilterSpec

To define a filter for a `ClusterLogSink` or `LogSink` resource, perform the following steps:

1. Configure `include-logs` as `true` in the `filters` section of your Sink resource configuration file.
2. Add a `filterSpec` section to the `filters` section in your Sink resource configuration file:

```
filters:
 include-logs: true
 filterSpec:
 - filter:
 - condition: CONDITION-TYPE
 key: KEY
 value: VALUE
 Output_properties:
 output_target
```

Where:

- ◊ `CONDITION-TYPE` is the type of conditional test to apply to the log entries.
- ◊ `KEY` is the name of the key to validate.
- ◊ `VALUE` is the value of the key to validate.

For more information, see [Define a Filter Condition](#) below.

3. Create additional condition sections for each condition you want to apply to your filter.
4. Create additional filter sections as needed.

## Define a Filter Condition

When you define a Fluent Bit Sink resource filter, you must specify the filter's `condition` type, `key`, and `value`:

- The filter `condition` type is the type of rule to apply to the filter. For example, common `condition` types include `key_value_equals` and `key_value_does_not_equal`.
- The filter `key` indicates which property in the log entry JSON to test against.
- The filter `value` is the value of the key to test for.

To build a filter for your `filterSpec`, see:

- [Chose a Filter Condition Type](#)
- [Create a Filter Key](#)

For example, consider a situation where you wish to include only production Pod log entries in your logs. If the administrator has tagged production Pods as a `production`, the output JSON for error log entries might have a format similar to the following:

```
"log":"[ERROR]counter 2: 2439 Wed Feb 23 06:51:08 UTC 2022",
```

```

 "kubernetes":
 {
 "pod_name": "counter-2",
 "namespace_name": "ns-1",
 "pod_id": "05184f35-44ba-45ed-8f75-5016321619ce",
 "labels": {"environment": "production"},
 ...
 }

```

All of the following example filter settings for `condition`, `key`, and `value` match the example JSON-structured log entry above:

- Match against error condition:
  - condition: `key_value_matches`
  - key: `log`
  - value: `^/[ERROR/]`
- Match against pod\_id:
  - condition: `key_value_equals`
  - key: `$kubernetes['pod_id']`
  - value: `05184f35-44ba-45ed-8f75-5016321619ce`
- Match against environment = production:
  - condition: `key_value_equals`
  - key: `$kubernetes['labels']['environment']`
  - value: `production`

The resulting filter would be written as:

```

filters:
 include-logs: true
filterSpec:
 - filter:
 - condition: key_value_matches
 key: log
 value: ^/[ERROR/]
 - filter:
 - condition: key_value_equals
 key: $kubernetes['pod_id']
 value: 05184f35-44ba-45ed-8f75-5016321619ce
 - filter:
 - condition: key_value_equals
 key: $kubernetes['labels']['environment']
 value: production

```

## Choose a Filter Condition Type

The following are the supported filter condition types supported by TKGI:

| Condition | Key | Value | Description |
|-----------|-----|-------|-------------|
|-----------|-----|-------|-------------|

|                                                        |                |                  |                                                                              |
|--------------------------------------------------------|----------------|------------------|------------------------------------------------------------------------------|
| <code>key_exists</code>                                | string:ke<br>y | none             | <code>true</code> if key exists                                              |
| <code>key_does_not_exist</code>                        | string:ke<br>y | none             | <code>true</code> if key does not exist                                      |
| <code>a_key_matches</code>                             | regexp:k<br>ey | none             | <code>true</code> if a key matches regex                                     |
| <code>no_key_matches</code>                            | regexp:k<br>ey | none             | <code>true</code> if not keys match regex                                    |
| <code>key_value_equals</code>                          | string:ke<br>y | string:val<br>ue | <code>true</code> if key exists and its value equals value                   |
| <code>key_value_does_not_equal</code>                  | string:ke<br>y | string:val<br>ue | <code>true</code> if key exists and its value does not equal value           |
| <code>key_value_matches</code>                         | string:ke<br>y | regexp:val<br>ue | <code>true</code> if key exists and its value matches value                  |
| <code>key_value_does_not_match</code>                  | string:ke<br>y | regexp:val<br>ue | <code>true</code> if key exists and its value does not match value           |
| <code>matching_keys_have_matching_values</code>        | regexp:k<br>ey | regexp:val<br>ue | <code>true</code> if all keys matching key have values matching value        |
| <code>matching_keys_do_not_have_matching_values</code> | regexp:k<br>ey | regexp:val<br>ue | <code>true</code> if all keys matching key do not have values matching value |

## Create a Filter Key

The filter key follows the pattern:

```
key: $ROOTNODE-NAME [OBJECT-NAME] [PARAMETER-NAME] `
```

Where:

- `ROOTNODE-NAME` is the name of the root node in the JSON log entry.
- `OBJECT-NAME` is name of a child node in the root node.
- `PARAMETER-NAME` is name of a parameter in the child node.

For examples of how to create a Filter Key, see [Define a Filter Condition](#) above.

## (Optional) Unsecured ClusterLogSink and LogSink Log Forwarding

By default, TKGI uses a secure connection for log forwarding when using `ClusterLogSink` and `LogSink` resources of type `syslog` or `webhook`.

For debugging purposes on a local machine, you might want to temporarily forward logs using an unsecured connection. To do this, you must:

1. Deactivate sink forwarding validation by running the following command:

```
kubectl delete validatingwebhookconfigurations validator.pksapi.io
```

2. Set `enable_tls` to `false` in your log destination YAML file.



**Warning:** Deactivating secure log forwarding is not recommended.

## Create ClusterMetricSink and MetricSink Resources

ClusterMetricSink and MetricSink resources collect metrics from different sources:

- `ClusterMetricSink` resources collect metrics from a cluster. If you want to collect pod usage metrics, you have to use a ClusterMetricSink.
- `MetricSink` resources collect metrics from a namespace within a cluster. If you want to isolate a certain workload's metrics to its own output, you have to use a namespaced MetricSink.

### ClusterMetricSink Resources

#### How It Works

By default, a `ClusterMetricSink` resource collects metrics from a cluster using the [Kubernetes Input Plugin] (<https://github.com/influxdata/telegraf/tree/1.13.4/plugins/inputs/kubernetes>) and writes them to one or more outputs that you specify in your `ClusterMetricSink` configuration:

- ClusterMetricSink DaemonSet collect pod usage metrics using Telegraf agents running on every node in a cluster.
- The ClusterMetricSink is able to fetch pod metrics using the Kubernetes input plugin, which gives access to pod usage metrics for each kubernetes node. Pod metrics come from the underlying container runtime, which does not isolate metrics based on namespace.

Do not use a ClusterMetricSink to isolate a specific workload's metrics to its own output. A ClusterMetricSink cannot isolate one input from all the others because all of a ClusterMetricSink's configurations are located in a shared ConfigMap. This means that each ClusterMetricSink's input will also go to all other ClusterMetricSinks' outputs.

### MetricSink Resources

A `MetricSink` resource runs in a single node only and collects metrics from a namespace within a cluster using `prometheus.io/scrape` annotations set to `true`. A MetricSink creates a unique telegraf agent pod and ConfigMap in the namespace, then writes the metrics to one or more outputs that you specify in your `MetricSink` configuration.

Do not use MetricSinks to collect pod usage metrics. Pod metrics come from the underlying container runtime, which does not isolate metrics based on namespace, so only ClusterMetricSink is able to fetch pod metrics.

For a list of supported output plugins, see [Output Plugins](#) in the telegraf GitHub repository.

#### When to Use MetricSink vs. ClusterMetricSink

### Case 1. Isolating Output => MetricSink

If you want to isolate a certain workload's metrics to its own output, you have to use a namespaced MetricSink. A MetricSink creates a unique telegraf agent pod and ConfigMap in the namespace.

ClusterMetricSinks cannot isolate one input from all the others because all ClusterMetricSinks' configurations are located in a shared ConfigMap. This means that each ClusterMetricSink's inputs will also go to all other ClusterMetricSinks' outputs.

Note that MetricSinks do not have access to pod usage metrics, as the source of those metrics does not filter by namespace and gives usage for all pods on a node.

### Case 2. Pod Usage Metrics (CPU, Memory) => ClusterMetricSink

If you want to get pod usage metrics, you have to use a ClusterMetricSink.

To get pod usage metrics, Telegraf agents need to run on every node in a cluster, and this is what the ClusterMetricSink DaemonSet does.

Pod metrics come from the underlying container runtime, which does not isolate metrics based on namespace, so only ClusterMetricSink is able to fetch pod metrics.

ClusterMetricSinks use the Kubernetes input plugin, which gives access to pod usage metrics for each kubernetes node. MetricSinks only run in a single node, so they do not have access to pod usage metrics.

## Create a ClusterMetricSink or MetricSink Resource

To define a `ClusterMetricSink` or `MetricSink` resource, perform the following steps:

1. Create a YAML file in the following format:

```
apiVersion: pksapi.io/v1beta1
kind: YOUR-SINK-RESOURCE
metadata:
 name: YOUR-SINK
 namespace: YOUR-NAMESPACE
spec:
 inputs:
 outputs:
 - type: YOUR-OUTPUT-PLUGIN
 telegraf_agent_config:
 interval: INTERVAL-LENGTH
```

Where:

- ❖ `YOUR-SINK-RESOURCE` is the sink resource you want to create. This must be either `ClusterMetricSink` or `MetricSink`. For information about these sink resources, see [Overview](#).
- ❖ `YOUR-SINK` is a name you choose for your sink.
- ❖ `YOUR-NAMESPACE` is the name of your namespace. Omit this line if you are creating `ClusterMetricSink`.
- ❖ `YOUR-OUTPUT-PLUGIN` is the name of the output plugin you want to use for your metrics.

- ❖ (Optional) `INTERVAL-LENGTH` is the interval, in seconds, separating sink collection of input data. The assigned `INTERVAL-LENGTH` must be a positive integer less than 9223372037. The default is `10`.



**Note:** You can leave the `inputs` field blank. For `ClusterMetricSink`, this field is configured to include metrics from the kubelet by default. For `MetricSink`, the field includes all `prometheus.io/scrape` annotations set to `true` by default.

For example:

```
apiVersion: pksapi.io/v1beta1
kind: ClusterMetricSink
metadata:
 name: http
spec:
 inputs:
 outputs:
 - type: http
 url: https://example.com
 method: POST
 data_format: json
```

This will send all cluster metrics provided by the kubernetes input plugin via json POST to <https://example.com>

## Create a ClusterMetricSink Resource for Node Exporter Metrics

To define a `clusterMetricSink` resource for collecting Node Exporter metrics, perform the following steps:

1. Activate Node Exporter on your cluster workers by selecting the **Enable node exporter on workers** checkbox in the **Tanzu Kubernetes Grid Integrated Edition** tile > **In-Cluster Monitoring**.
2. Create a YAML file in the following format:

```
apiVersion: pksapi.io/v1beta1
kind: ClusterMetricSink
metadata:
 name: YOUR-SINK
spec:
 inputs:
 - monitor_kubernetes_pods: true
 type: prometheus
 outputs:
 - type: YOUR-OUTPUT-PLUGIN
```

Where:

- ❖ `YOUR-SINK` is a name you choose for your sink.
- ❖ `YOUR-OUTPUT-PLUGIN` is the name of the output plugin you want to use for your metrics.

For example:

```
apiVersion: pksapi.io/v1beta1
kind: ClusterMetricSink
metadata:
 name: http
spec:
 inputs:
 - monitor_kubernetes_pods: true
 type: prometheus
 outputs:
 - type: http
 url: https://example.com
 method: POST
 data_format: json
```

3. Save the YAML file with an appropriate filename. For example, `my-cluster-metric-sink.yml`.
4. Apply the `ClusterMetricSink` resource to your cluster by running the following command:

```
kubectl apply -f YOUR-SINK.yml
```

Where `YOUR-SINK.yml` is the name of your YAML file.

For example:

```
$ kubectl apply -f my-cluster-metric-sink.yml
```

## List Sinks

To list sinks for clusters and namespaces, use the commands in the following sections.

## ClusterLogSink and LogSink Resources

To list cluster log sinks, run the following command:

```
kubectl get clusterlogsinks
```

To list namespace log sinks, run the following command:

```
kubectl -n YOUR-NAMESPACE get logsinks
```

Where `YOUR-NAMESPACE` is the name of your namespace.

## ClusterMetricSink and MetricSink Resources

To list cluster metric sinks, run the following command:

```
kubectl get clustermetricsinks
```

To list namespace metric sinks, run the following command:

```
kubectl -n YOUR-NAMESPACE get metricsinks
```

Where `YOUR-NAMESPACE` is the name of your namespace.

## Delete Sinks

To delete sinks for clusters and namespaces, use the commands in the following sections.

## ClusterLogSink and LogSink Resources

To delete a cluster log sink, run the following command:

```
kubectl delete clusterlogsink YOUR-SINK
```

Where `YOUR-SINK` is the name of your sink.

To delete a namespace log sink, run the following command:

```
kubectl -n YOUR-NAMESPACE delete logsink YOUR-SINK
```

Where:

- `YOUR-NAMESPACE` is the name of your namespace.
- `YOUR-SINK` is the name of your log sink.

## ClusterMetricSink and MetricSink Resources

To delete a cluster metric sink, use the following command:

```
kubectl delete clustermetricsink YOUR-SINK
```

Where `YOUR-SINK` is the name of your sink.

To delete a namespace metric sink, use the following command:

```
kubectl -n YOUR-NAMESPACE delete metricsink YOUR-SINK
```

Where:

- `YOUR-NAMESPACE` is the name of your namespace.
- `YOUR-SINK` is the name of your metric sink.

## Monitoring Clusters with Log Sinks

This topic describes the log sink resources you can use to monitor Kubernetes clusters provisioned by VMware Tanzu Kubernetes Grid Integrated Edition and their workloads.

## Overview

You can use the following sink resources to collect logs from your Kubernetes clusters.

| Sink Resource               | Sink Type | Description                                                                                                                                                                                                                                                                                       |
|-----------------------------|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>ClusterLogSink</code> | Log sink  | <p>Forwards logs from a cluster to a log destination. Logs are transported using one of the following:</p> <ul style="list-style-type: none"> <li>• The Syslog Protocol defined in <a href="#">RFC 5424</a></li> <li>• WebHook</li> <li>• Fluent Bit output plugins</li> </ul>                    |
| <code>LogSink</code>        | Log sink  | <p>Forwards logs from a namespace within a cluster to a log destination. Logs are transported using one of the following:</p> <ul style="list-style-type: none"> <li>• The Syslog Protocol defined in <a href="#">RFC 5424</a></li> <li>• WebHook</li> <li>• Fluent Bit output plugins</li> </ul> |

## Log Sinks

`ClusterLogSink` and `LogSink` resources collect pod logs and events from the Kubernetes API in your Kubernetes clusters. For more information, see:

- [Log Format](#)
- [Notable Kubernetes API Events](#)

## Log Format

In Tanzu Kubernetes Grid Integrated Edition, you can create `ClusterLogSink` and `LogSink` resources of the following types:

- Syslog
- WebHook
- Fluent Bit output plugins

Your log format depends on the type of `ClusterLogSink` or `LogSink` you want to use. For example, if you use a `ClusterLogSink` or `LogSink` resource of type `syslog`, Tanzu Kubernetes Grid Integrated Edition formats your logs as described in the sections below.

### Syslog Format

All log entries collected by `ClusterLogSink` and `LogSink` resources of type `syslog` include a prefix in the following format:

APP-NAME/NAMESPACE/POD-ID

Where:

- APP-NAME is `pod.log` or `k8s.event`.
- NAMESPACE is the namespace associated with the pod log or Kubernetes event.
- POD-ID is the ID of the pod associated with the pod log or Kubernetes event.

## Pod Logs

Pod logs are distinguished by the string `pod.log` in the APP-NAME field.

The following is a sample pod log entry:

```
36 <14>1 2018-11-26T18:51:41.647825+00:00 cluster-name pod.log/rocky-raccoon/logspewe
r-6b58b6689d-dhddj - - [kubernetes@47450 app="logspewer" pod-template-hash="2614622458
" namespace_name="rocky-raccoon" object_name="logspewer-6b58b6689d-dhddj" container_na
me="logspewer"] 2018/11/26 18:51:41 Log Message 589910
```

Where:

- `cluster-name` is the human-readable cluster name used when creating the cluster.
- `pod.log` is the APP-NAME.
- `rocky-raccoon` is the NAMESPACE.
- `logspewer-6b58b6689d-dhddj` is the POD-ID.

## Kubernetes API Events

Kubernetes API events are distinguished by the string `k8s.event` in the APP-NAME field.

The following is an example Kubernetes API event log entry:

```
Nov 14 16:01:49 cluster-name k8s.event/rocky-raccoon/logspewer-6b58b6689d-j9n: Success
fully assigned rocky-raccoon/logspewer-6b58b6689d-j9nq7 to vm-38dfd896-bb21-43e4-67b0
-9d2f339adaf1
```

Where:

- `cluster-name` is the human-readable cluster name used when creating the cluster.
- `k8s.event` is the APP-NAME.
- `rocky-raccoon` is the NAMESPACE.
- `logspewer-6b58b6689d-j9n` is the POD-ID.

## Notable Kubernetes API Events

The following section lists Kubernetes API events that can help assess Kubernetes scheduling problems in Tanzu Kubernetes Grid Integrated Edition.

To monitor for these events, look for log entries that contain the **Identifying String** indicated below for each event.

## Failure to Retrieve Containers from Registry

### ImagePullBackOff

|                               |                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>            | Image pull back offs occur when the Kubernetes API cannot reach a registry to retrieve a container or the container does not exist in the registry. The scheduler might be trying to access a registry that is not available on the network. For example, access to Docker Hub is blocked by a firewall. Other reasons might include the registry is experiencing an outage or a specified container has been deleted or was never uploaded. |
| <b>Identifying String</b>     | Error:ErrImagePull                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Example Sink Log Entry</b> | Jan 25 10:18:58 gke-bf-test-default-pool-aa8027bc-rnf6<br>k8s.event/default/test-669d4d66b9-zd9h4/: Error: ErrImagePull                                                                                                                                                                                                                                                                                                                      |

## Malfunctioning Containers

### CrashLoopBackOff

|                               |                                                                                                                                                                                                                                             |
|-------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>            | Crash loop back offs imply that the container is not functioning as intended. There are several potential causes of crash loop back offs, which depend on the related workload. To investigate further, examine the logs for that workload. |
| <b>Identifying String</b>     | Back-off restarting failed container                                                                                                                                                                                                        |
| <b>Example Sink Log Entry</b> | Jan 25 09:26:44 cluster-name k8s.event/monitoring/cost-analyzer-prometheus-<br>se: Back-off restarting failed container                                                                                                                     |

## Successful Scheduling of Containers

### ContainerCreated

|                           |                                                                                                                                                                                                  |
|---------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>        | Operators can monitor the creation and successful start of containers to keep track of platform usage at a high level. Cluster users can track this event to monitor the usage of their cluster. |
| <b>Identifying String</b> | Started container                                                                                                                                                                                |

---

|                                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Example Sink Log Entries</b> | <pre>Jan 25 09:14:55 cluster-name 35.239.18.250 k8s.event/rocky- raccoon/logspewer-6b58b6689d/: Created pod: logspewer-6b58b6689d-sr96t Jan 25 09:14:55 cluster-name 35.239.18.250 k8s.event/rocky- raccoon/logspewer-6b58b6689d-sr9: Successfully assigned rocky-raccoon/ logspewer-6b58b6689d-sr96t to vm-efe48928-be8e-4db5-772c-426ee7aa52f2 Jan 25 09:14:55 cluster-name k8s.event/rocky-raccoon/logspewer-6b58b6689d- mkd: Killing container with id docker://logspewer:Need to kill Pod Jan 25 09:14:56 cluster-name k8s.event/rocky-raccoon/logspewer-6b58b6689d- sr9: Container image "oratos/logspewer:v0.1" already present on machine Jan 25 09:14:56 cluster-name k8s.event/rocky-raccoon/logspewer-6b58b6689d- sr9: Created container Jan 25 09:14:56 cluster-name k8s.event/rocky-raccoon/logspewer-6b58b6689d- sr9: Started container</pre> |
|---------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

---

## Failure to Schedule Containers

### FailedScheduling

---

|                                 |                                                                                                                                                           |
|---------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>              | This event occurs when a container cannot be scheduled. For instance, this might occur due to lack of node resources.                                     |
| <b>Identifying String</b>       | <code>Insufficient RESOURCE</code><br>where <code>RESOURCE</code> is a specific type of resource. For example, <code>cpu</code> .                         |
| <b>Example Sink Log Entries</b> | <pre>Jan 25 10:51:48 gke-bf-test-default-pool-aa8027bc-rnf6 k8s.event/default/test2-5c87bf4b65-7fdtd/: 0/1 nodes are available: 1 Insufficient cpu.</pre> |

---

## Related Links

For more information about log sinks, see:

- [Creating and Managing Sink Resources](#).  
Follow these instructions to create `ClusterLogSink` and `LogSink` resources, described in [Overview](#) above.
- [Sink Architecture in Tanzu Kubernetes Grid Integrated Edition](#).  
See this topic for conceptual information about sinks.

## Monitoring Windows Worker Clusters and Nodes

This topic describes how to set up monitoring tools to capture metrics from Windows worker-based Kubernetes clusters deployed by Tanzu Kubernetes Grid Integrated Edition (TKGI).

## Overview

You can use any of the following monitoring tools to capture metrics from TKGI Windows worker-based Kubernetes clusters:

- [Healthwatch](#)
- [Wavefront](#)

- Prometheus with Grafana

## Healthwatch

You can use the [Healthwatch](#) Healthwatch Exporter for TKGI tile to monitor the health of the TKGI Control Plane and your Linux and Windows cluster control plane nodes.

Healthwatch enables you to monitor the functionality of your TKGI environment and can be configured to expose metrics to a service or database external to your Ops Manager foundation. For more information, see [Overview of the Healthwatch Exporter for TKGI Tile](#).

To configure cluster discovery in Healthwatch, see [Configuring TKGI Cluster Discovery](#) in the Healthwatch documentation.

## Wavefront

Wavefront runs as an external service that you set up to monitor Windows worker-based clusters the same way that you set it up to monitor clusters running Linux worker nodes:

1. Install Helm on your local machine, if you do not already have it, by following [Install and Configure Helm](#) in the topic *Using Helm with Tanzu Kubernetes Grid Integrated Edition*.
2. Use the Helm CLI to deploy Wavefront to the target cluster:

```
helm install wavefront波浪前端/wavefront --namespace波浪前端 \
--set clusterName=CLUSTER-NAME \
--set wavefront.url=https://INSTANCE-NAME.wavefront.com \
--set wavefront.token=API-TOKEN \
--set collector.usePKSPrefix=true \
--set collector.useDaemonset=false
```

Where:

- CLUSTER-NAME is the name of your Kubernetes cluster.
- INSTANCE-NAME is the subdomain name for your Wavefront instance.
- API-TOKEN is the Wavefront API token for your Wavefront subscription.

3. Do one of the following:
  - Configure Wavefront Integration in Ops Manager using the [VMware Tanzu Kubernetes Grid Integrated Edition Integration](#) procedure in the Wavefront documentation.
  - Set up a metric sink to send metrics to Wavefront following the instructions in [Creating and Managing Sink Resources](#).

## Prometheus with Grafana

[Prometheus](#) and [Grafana](#) are open source tools you can use to monitor your Kubernetes Windows clusters and worker nodes, and to visualize and alert on events occurring in those clusters and worker nodes.

## Prerequisites

Before installing Prometheus ensure you have installed kubectl. For more information about installing kubectl, see [Install and Set Up kubectl](#) in the Kubernetes documentation.

For additional Prometheus and Grafana prerequisites, see [Prerequisites](#) in the *kube-prometheus* GitHub repository.

## Overview

To monitor Windows clusters and worker node metrics using Prometheus and Grafana:

1. [Install Prometheus and Grafana](#)
2. [Install the Windows Node Exporter](#)
3. [Administer Prometheus and Grafana](#)

## Install Prometheus and Grafana

To install Prometheus and Grafana:

1. [Download the Prometheus Source Code](#)
2. [Generate Your Prometheus Dashboard Configuration](#)
3. [Generate Monitoring Rules](#)
4. [Deploy Prometheus and Grafana](#)
5. [Verify Grafana is Running](#)

### Download the Prometheus Source Code

To download the Prometheus source code:

1. Clone or download the Prometheus operator source code from [Latest release](#) in the `prometheus-operator/kube-prometheus` GitHub repository.
2. Clone or download the Prometheus monitoring mixin for kubernetes source code from [Latest release](#) in the `kubernetes-monitoring/kubernetes-mixin` GitHub repository.

### Generate Your Prometheus Dashboard Configuration

To generate a Prometheus dashboard configuration:

1. Open a command line.
2. Change directory to the `kubernetes-mixin/dashboards` source code directory.
3. Edit the `dashboards.libsonnet` configuration file.
4. To add the Windows dashboard to the configuration add the following to the `dashboards.libsonnet` file:

```
(import 'windows.libsonnet')
```

5. Save the file.

6. Change directory to the `kubernetes-mixin` source code root directory.
7. To generate the Prometheus dashboard, run:

```
make dashboards_out
```

8. Review the `kubernetes-mixin/dashboards_out` directory and confirm the following Windows dashboard definition files were created:

```
k8s-resources-windows-cluster.json
k8s-resources-windows-namespace.json
k8s-resources-windows-pod.json
k8s-windows-cluster-rsrc-use.json
k8s-windows-node-rsrc-use.json
```

## Generate Monitoring Rules

To generate Prometheus monitoring rules:

1. Open a command line.
2. Copy the rule definition file from the `kubernetes_mixin/rules` directory to the `kube-prometheus/jsonnet/kube-prometheus/rules` directory:

```
cp KUBERNETES-MIXIN-PATH/rules/windows.libsonnet PROMETHEUS-OP-PATH/jsonnet/kube-prometheus/rules/windows.libsonnet
```

Where:

- `KUBERNETES-MIXIN-PATH` is the `kubernetes_mixin` source directory.
  - `PROMETHEUS-OP-PATH` is the Prometheus operator path.
3. Change directory to the `kube-prometheus/jsonnet/kube-prometheus/rules` source code directory.
  4. Edit the `rules.libsonnet` configuration file.
  5. To add the Windows rule to the configuration add the following to the `rules.libsonnet` file:

```
(import 'windows.libsonnet')
```

6. Save the file.
7. Change directory to the `kube-prometheus` Prometheus source code root directory.
8. To generate all rules, run:

```
make manifests
```

## Deploy Prometheus and Grafana

To deploy Prometheus and Grafana:

1. Get `kubeconfig` for your Windows cluster. For more information, see [Retrieving Cluster Credentials and Configuration](#).

2. Open a command line.
3. Change directory to the `kube-prometheus` Prometheus source code root directory.
4. To deploy Prometheus and Grafana, run:

```
kubectl create -f manifests/setup
until kubectl get servicemonitors --all-namespaces ; do date; sleep 1; echo "";
done
kubectl create -f manifests/
```

5. To forward the Prometheus and Grafana ports to localhost run:

```
kubectl --namespace monitoring port-forward svc/prometheus-k8s PROMETHEUS-PORT
kubectl --namespace monitoring port-forward svc/grafana GRAFANA-PORT
```

Where:

- ◊ `PROMETHEUS-PORT` is a localhost port for Prometheus, for example `9090`.
- ◊ `GRAFANA-PORT` is a localhost port for Grafana, for example `3000`.

For more information about installing Prometheus and Grafana, see [Quickstart](#) in the *prometheus-operator/kube-prometheus* GitHub repository.

## Verify Grafana is Running

To verify Grafana is running:

1. To open the Grafana Dashboard, open a browser to:

```
http://localhost:GRAFANA-PORT
```

Where `GRAFANA-PORT` is the localhost port for Grafana, for example `3000`.

2. To log in to the Grafana Dashboard authenticate with user `admin`, password `admin`.
3. To view Kubelet metrics from all of your windows workers, navigate to **Dashboard > Default > Kubernetes/Kubelet**.

## Install the Windows Node Exporter

To monitor Windows worker node metrics, you need endpoints, the Windows Node Exporter service, and the Windows Node Exporter service monitor.

To install and configure the Windows Node Exporter:

1. [Deploy the Windows Node Exporter](#)
2. [Configure the Windows Node Exporter](#)
3. [Set Up the Grafana Windows Node Dashboard](#)

### Deploy the Windows Node Exporter

The Windows Node Exporter must be installed on all of the Windows worker nodes you want to monitor.

To determine the names and IP addresses of all of your Windows worker nodes:

1. Open a command line.
2. Run the following:

```
kubectl get nodes -L spec.ip
```

3. Note the names and IP addresses of the Windows worker nodes.

To deploy the Windows Node Exporter to all of your Windows worker nodes, repeat the steps below for each Windows worker node you want to monitor:

1. Start an SSH session to the Windows worker.
2. To download the [[windows\\_exporter-0.13.0-amd64.msi](https://github.com/prometheus-community/windows_exporter/releases/download/v0.13.0/windows_exporter-0.13.0-amd64.msi)] ([https://github.com/prometheus-community/windows\\_exporter/releases/download/v0.13.0/windows\\_exporter-0.13.0-amd64.msi](https://github.com/prometheus-community/windows_exporter/releases/download/v0.13.0/windows_exporter-0.13.0-amd64.msi)) to the worker from v0.13.0 in the *prometheus-community/windows\_exporter* repository on GitHub:

```
Powershell Invoke-WebRequest https://github.com/prometheus-community/windows_exporter/releases/download/v0.13.0/windows_exporter-0.13.0-amd64.msi ^
-OutFile DOWNLOAD-PATH
```

Where `DOWNLOAD-PATH` is the full path to the output location and filename for the retrieved file. For example, `c:\windows_exporter-0.13.0-amd64.msi`.

3. To start the `windows_exporter` msi as a Windows service, open a command line:

```
msiexec /i DOWNLOAD-PATH ENABLED_COLLECTORS="cpu,cs,logical_disk,net,os,system,
container,memory"
```

Where `DOWNLOAD-PATH` is the full path to the output location and filename for the retrieved file. For example, `c:\windows_exporter-0.13.0-amd64.msi`.

## Configure the Windows Node Exporter

To support monitoring by Prometheus and Grafana, the Windows Node Exporter service must be configured to access your Windows worker node metrics.

To configure the Windows Node Exporter:

1. Create a file named `windows-exporter.yml`.
2. Populate the file with the following:

```
apiVersion: v1
kind: Endpoints
metadata:
 labels:
 k8s-app: APP-NAME
 name: ENDPOINT-NAME
 namespace: kube-system
subsets:
- addresses:
```

```

- ip: NODE-IP
 targetRef:
 kind: Node
 name: NODE-NAME
 ports:
 - name: http-metrics
 port: 9182
 protocol: TCP

apiVersion: v1
kind: Service
metadata:
 labels:
 k8s-app: APP-NAME
 name: SERVICE-NAME
 namespace: kube-system
spec:
 clusterIP: None
 ports:
 - name: http-metrics
 port: 9182
 protocol: TCP
 targetPort: 9182
 sessionAffinity: None
 type: ClusterIP

apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
 labels:
 k8s-app: APP-NAME
 name: SERVICEMONITOR-NAME
 namespace: monitoring
spec:
 endpoints:
 - interval: 30s
 port: http-metrics
 jobLabel: k8s-app
 namespaceSelector:
 matchNames:
 - kube-system
 selector:
 matchLabels:
 k8s-app: APP-NAME

```

Where:

- ◊ **APP-NAME** is the name for the Kubernetes app.
- ◊ **ENDPOINT-NAME** is the name for the endpoint.
- ◊ **SERVICE-NAME** is the name for the service.
- ◊ **SERVICEMONITOR-NAME** is the name for the service monitor.
- ◊ **NODE-IP** is the IP address of the Windows worker node to monitor.
- ◊ **NODE-NAME** is the name of the Windows worker node to monitor.

For example:

```
apiVersion: v1 kind: Endpoints metadata: labels: k8s-app: wmi-exporter name: wmi-exporter namespace: kube-system subsets: - addresses: - ip: 192.168.1.1 targetRef: kind: Node name: nodeone - ip: 192.168.1.1 targetRef: kind: Node name: nodetwo ports: - name: http-metrics port: 9182 protocol: TCP
```

**apiVersion: v1 kind: Service metadata: labels: k8s-app: wmi-exporter name: wmi-exporter namespace: kube-system spec: clusterIP: None ports: - name: http-metrics port: 9182 protocol: TCP targetPort: 9182 sessionAffinity: None type: ClusterIP**

```
apiVersion: monitoring.coreos.com/v1 kind: ServiceMonitor metadata: labels: k8s-app: wmi-exporter name: wmi-exporter namespace: monitoring spec: endpoints: - interval: 30s port: http-metrics jobLabel: k8s-app namespaceSelector: matchNames: - kube-system selector: matchLabels: k8s-app: wmi-exporter
```

3. Create an additional `subsets.addresses` item for each Windows worker node you want to monitor:

```
subsets:
- addresses:
 - ip: NODE-ONE-IP
 targetRef:
 kind: Node
 name: NODE-ONE-NAME
 - ip: NODE-TWO-IP
 targetRef:
 kind: Node
 name: NODE-TWO-NAME
 - ip: NODE-THREE-IP
 targetRef:
 kind: Node
 name: NODE-THREE-NAME
```

Where:

- ◊ `NODE-ONE-IP` is the IP address of the first Windows worker node to monitor.
- ◊ `NODE-ONE-NAME` is the name of the first Windows worker node to monitor.
- ◊ `NODE-TWO-IP` is the IP address of the second Windows worker node to monitor.
- ◊ `NODE-TWO-NAME` is the name of the second Windows worker node to monitor.
- ◊ `NODE-THREE-IP` is the IP address of the third Windows worker node to monitor.
- ◊ `NODE-THREE-NAME` is the name of the third Windows worker node to monitor.

4. Save your changes to the `windows-exporter.yml` file.
5. To create the Windows node exporter objects, run:

```
kubectl create -f windows-exporter.yml
```

6. Wait a few seconds for the service to be ready.
7. To verify your new targets in Prometheus, open a browser to:

```
http://localhost:_PROMETHEUS-PORT/targets
```

Where `PROMETHEUS-PORT` is the localhost port for Prometheus, for example `9090`.

## Set Up the Grafana Windows Node Dashboard

To import the dashboard files generated above to Grafana:

1. Open a browser to your Grafana dashboard.
2. To open the **Import** tab, click **+** on the Grafana side menu.
3. Click **Upload .json file**.
4. Upload the five dashboard files from the `kubernetes-mixin/dashboards_out` directory one at a time:

```
k8s-resources-windows-cluster.json
k8s-resources-windows-namespace.json
k8s-resources-windows-pod.json
k8s-windows-cluster-rsrc-use.json
k8s-windows-node-rsrc-use.json
```

5. To verify the dashboard, open a browser to your Grafana dashboard.

If you prefer to use a different Grafana dashboard, you can use and configure the preferred dashboard to monitor the Windows Node Exporter. For more information, see [Dashboards](#) at the Grafana Labs site.

## Administer Prometheus and Grafana

After completing the minimum installation steps described above:

- Change your Grafana installation's account authentication to not use the default `admin/admin` user/password pair.
- Review the Grafana and Prometheus documentation for any additional steps you should complete:
  - ◊ For information about managing Grafana, see [Administration](#) in the Grafana documentation.
  - ◊ For information about managing Prometheus, see [Customizing Kube-Prometheus](#) in the `kube-prometheus` documentation.

## Logging Windows Worker Workloads

This topic describes how to install and configure components and integrations to capture VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) Windows Worker Kubernetes cluster and worker node logs.

## Prerequisites

Before starting the tasks in this topic:

- Your environment must be a Windows 2019 data center with vRealize Log Insight (vRLI).
- You must have an accessible container image registry.
- Docker must be installed on the local machine you are working from. For more information, see [Prepare the Working Environment](#) below.

## Overview

This procedure describes how to send logs to vRLI from Windows workers on TKGI-provisioned Windows clusters using Fluent Bit.

To use Fluent Bit to send Windows worker logs to vRLI:

1. [Prepare a Fluent Bit Image](#)
2. [Install Fluent Bit](#)

## Prepare the Working Environment

To prepare your working environment:

1. Install [docker](#) on the local machine you are working from. If you are working from a TKGI-provisioned Windows worker node, Docker is already installed.
2. Follow the steps in [Configure Docker for Creating Windows Containers](#) below.

## Configure Docker for Creating Windows Containers

To configure Docker to create Windows containers:

1. To open the Docker Desktop menu, click the Docker icon in the Windows system tray notification area.
2. If the **Switch to Windows containers...** option is present on the Docker Desktop menu, click it. The Docker Desktop menu should now display the **Switch to Linux containers...** option.

For more information, see [Switch between Windows and Linux containers](#) in *Docker Desktop for Windows user manual* in the Docker documentation.

## Prepare a Fluent Bit Image

1. [Build a Windows Fluent Bit Docker Image](#)
2. [Configure Fluent Bit](#)

## Build a Windows Fluent Bit Docker Image

To build a Fluent Bit Windows Docker image:

1. Log in to the Windows 2019 machine where you are doing your work.
2. Download the Fluent Bit source code from the [fluent/fluent-bit](#) repository on GitHub.

3. Open the Windows Dockerfile located at `fluent-bit/dockerfiles/Dockerfile.windows` for editing.
4. Set the `FLUENTBIT_VERSION` parameter to `1.8`. Fluent Bit v1.8 supports the native syslog plugin.
5. Change the last line in the Windows Dockerfile from:

```
ENTRYPOINT ["fluent-bit.exe", "-i", "dummy", "-o", "stdout"]
```

to instead read:

```
ENTRYPOINT ["fluent-bit.exe", "--config", "/fluent-bit/etc/fluent-bit.conf"]
```

6. Save the modified Windows Dockerfile.
7. Configure Docker for creating Windows containers. For more information, see [Configure Docker for Creating Windows Containers](#) above.
8. To build a Fluent Bit container image, run:

```
docker.exe build -f Dockerfile.windows -t fluent-bit-1.8 .
```

While building the Fluent Bit container, Docker downloads the Fluent Bit Windows installation ZIP and *Microsoft Visual C++ Redistributable Update* and installs `vc_redist.x64.exe` in the new container:

- \* If the process fails while Docker downloads the Microsoft Visual C++ Redistributable Update or while installing `vc_redist.x64.exe` in the new container, see '[Cannot Find Path](#)' Error When Creating the Fluent Bit Docker Container in *Troubleshooting* below.
- \* If Docker fails to invoke `Invoke-WebRequest` while downloading the Fluent Bit Windows installation ZIP from fluentbit.io, see '[The remote name could not be resolved](#)' Error When Creating the Fluent Bit Docker Container in *Troubleshooting* below.

9. Push the Fluent Bit container image to your registry.

## Configure Fluent Bit

A Fluent Bit configuration defines the `ServiceAccount`, `ClusterRole`, and `ClusterRoleBinding` objects that ensure that the Fluent Bit Kubernetes filter can access and read metadata from the Kubernetes API server `kubernetes.default.svc.cluster.local:443`.

You must configure Fluent Bit using a YAML configuration file. For more information on configuring the Syslog output plugin, see [Syslog](#) in the Fluent Bit documentation.



**Note:** Because of Docker container drive mounting limitations, the Fluent Bit configuration below mounts the entire `E:` disk to the container, allowing Fluent Bit to read content from the container log path `E:\docker-root`.

To create a Fluent Bit deployment configuration YAML file:

1. Create a file named `fluent-bit.yml`.
2. Populate the file with the following:

```

apiVersion: v1
kind: ServiceAccount
metadata:
 name: fluent-bit-win
 namespace: pks-system

apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
 name: fluent-bit-read
rules:
- apiGroups: [""]
 resources:
 - namespaces
 - pods
 verbs: ["get", "list", "watch"]

apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
 name: fluent-bit-read
roleRef:
 apiGroup: rbac.authorization.k8s.io
 kind: ClusterRole
 name: fluent-bit-read
subjects:
- kind: ServiceAccount
 name: fluent-bit-win
 namespace: pks-system

apiVersion: v1
kind: ConfigMap
metadata:
 name: fluent-bit-windows-config
 labels:
 app: fluent-bit
 namespace: pks-system
data:
 fluent-bit.conf: |
 [SERVICE]
 Flush 5
 Log_Level debug
 Daemon off
 Parsers_File parsers.conf

 [INPUT]
 Name tail
 Tag kube./*
 Path C:\var\log\containers*.log
 Parser docker
 DB /var/log/flb_kube1.db
 Skip_Long_Lines On
 Refresh_Interval 60

 [FILTER]
 Name kubernetes
 Match kube./*
 Kube_URL https://kubernetes.default.svc.cluster.local:443

```

```

 Kube_CA_File /var/run/secrets/kubernetes.io/serviceaccount/ca.crt
 Kube_Token_File /var/run/secrets/kubernetes.io/serviceaccount/token
 Merge_Log On
 Merge_Log_Key log_processed
 K8S-Logging.Parser On
 K8S-Logging.Exclude Off
 Kube_Tag_Prefix kube.cvar.log.containers.

[FILTER]
 Name nest
 Match kube.*
 Operation lift
 Nested_Under kubernetes

[OUTPUT]
 Name syslog
 Match *
 Host OUTPUT-ADDRESS
 Port OUTPUT-PORT
 Mode MODE
 syslog_format rfc5424
 Syslog_Hostname_key host
 Syslog_Appname_key pod_name
 Syslog_Procid_key container_name
 Syslog_Message_key log
 # tls on #only needed when use tls mode
 # tls.verify off #only needed when use tls mode

parsers.conf: |
[PARSER]
 Name json
 Format json
 Time_Key time
 Time_Format %d/%b/%Y:%H:%M:%S %z

[PARSER]
 Name docker
 Format json
 Time_Key time
 Time_Format %Y-%m-%dT%H:%M:%S.%L
 Time_Keep On

apiVersion: apps/v1
kind: DaemonSet
metadata:
 labels:
 app: fluent-bit
 name: fluent-bit-windows
 namespace: pks-system
spec:
 selector:
 matchLabels:
 app: fluent-bit
 template:
 metadata:
 labels:
 app: fluent-bit
 spec:

```

```

nodeSelector:
 beta.kubernetes.io/os: windows
tolerations:
- key: "windows"
 operator: "Equal"
 value: "2019"
 effect: "NoSchedule"
containers:
- image: fluent-bit-1.8
 imagePullPolicy: IfNotPresent
 name: fluent-bit
 volumeMounts:
 - mountPath: /var/log
 name: varlog
 readOnly: false
 - mountPath: /fluent-bit/etc
 name: fluent-bit-windows-config
 - mountPath: "E:"
 name: rootcontainers
 volumes:
 - name: varlog
 hostPath:
 path: /var/log
 - name: rootcontainers
 hostPath:
 path: E:/
 - configMap:
 defaultMode: 420
 name: fluent-bit-windows-config
 name: fluent-bit-windows-config
 serviceAccountName: fluent-bit-win
updateStrategy:
 type: RollingUpdate

```

Where:

- ◊ **OUTPUT-ADDRESS** is the IP address of your vRealize Log Insight installation.
- ◊ **OUTPUT-PORT** is the port to use to communicate with your vRealize Log Insight installation. If you enable TLS, set the **Port** parameter to **1514**, otherwise set **Port** to **514** for most installations.
- ◊ **MODE** is the desired transport type. The available options are **tcp**, **tls**, and **udp**. If you set **MODE** as **tls**, you must also include the required **tls** parameters.
- ◊ **FLUENT-BIT-IMAGE** is the name of the Fluent Bit image in your registry. For example, **fluent-bit-1.8**.

3. Save the file.

## Install Fluent Bit

To install Fluent Bit:

1. Deploy Fluent Bit on the Windows Cluster
2. Validate the Fluent Bit Deployment Using a Sample App

## Deploy Fluent Bit on the Windows Cluster

To send container logs to vRLI, deploy Fluent Bit and related objects using your deployment configuration file.

To deploy Fluent Bit:

1. Deploy Fluent Bit using kubectl:

```
kubectl create -f CONFIG-FILE
```

Where `CONFIG-FILE` is the filename of your Fluent Bit deployment configuration file.

For example:

```
kubectl create -f fluent-bit.yml
```

## Validate the Fluent Bit Deployment Using a Sample App

1. [Configure a Sample App](#)
2. [Deploy the Sample App](#)
3. [Validate the Fluent Bit Deployment](#)

### Configure a Sample App

To confirm that logging is working correctly, use a sample app that outputs log entries frequently.

The `logspewer` sample app defined below outputs a log every 10 seconds when running.

To configure a `logspewer` sample app for testing:

1. Create a file named `sample.yml`.
2. Populate the file with the following:

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: logspewer
 namespace: pks-system
 labels:
 app: logspewer
spec:
 replicas: 1
 selector:
 matchLabels:
 app: logspewer
 template:
 metadata:
 labels:
 app: logspewer
 spec:
 containers:
 - name: logspewer
 image: pivotalgreenhouse/logspewer:latest
 env:
```

```

- name: INTERVAL_IN_SECONDS
 value: "10"
nodeSelector:
 kubernetes.io/os: windows
tolerations:
- key: "windows"
 operator: "Equal"
 value: "2019"
 effect: "NoSchedule"

```

3. Save the file.

## Deploy the Sample App

Deploy the test app using kubectl:

1. To deploy the test app:

```
kubectl create -f CONFIG-FILE
```

Where `CONFIG-FILE` is the filename of your sample app deployment configuration file.

For example:

```
kubectl create -f sample.yml
```

## Validate the Fluent Bit Deployment

Validate your Fluent Bit configuration and confirm that Fluent Bit is functioning using the test app you created.

To confirm the sample app's logs are being written to vRLI:

1. Open vRLI.
2. Open the **vRLI > Interactive Analytics** tab.
3. Search the list for “logspewer”. The “logspewer” items are the log entries generated by the sample app.

## Troubleshooting

---

### ‘Cannot Find Path’ Error When Creating the Fluent Bit Docker Container

#### Symptom

The error message `Copy-Item : Cannot find path 'C:\Windows\System32\msvcp140.dll'` is displayed while Docker installs the *Microsoft Visual C++ Redistributable Update*.

#### Description

When Docker builds the Fluent Bit container, it installs the *Microsoft Visual C++ Redistributable Update* to the container. To do this, it downloads the *Redistributable Update* as `vc_redist.x64.exe`, installs the update in the new Docker container, and copies three DLL files to the `/fluent-bit/bin/` directory.

If a `Copy-Item : Cannot find path` error is returned for either `msvcp140.dll`, `vccorlib140.dll`, or `vcruntime140.dll` the installation of `vc_redist.x64.exe` has failed.

## Workaround

To manually install the *Microsoft Visual C++ Redistributable Update* to the Fluent Bit container:

1. Download the *Microsoft Visual C++ Redistributable Update*, `vc_redist.x64.exe`, from Microsoft.
2. Install `vc_redist.x64.exe` on your local Windows 2019 machine.
3. Copy the following files from `C:\Windows\System32\` to the directory containing `Dockerfile.windows`: `msvcp140.dll`, `vccorlib140.dll`, and `vcruntime140.dll`.
4. Modify your `Dockerfile.windows` file:

1. Remove the following lines from the file:

```
RUN Write-Host ('Installing Visual C++ Redistributable Package'); `
 Start-Process /local/vc_redist.x64.exe -ArgumentList '/install', '/quiet', '/norestart' -NoNewWindow -Wait; `
 Copy-Item -Path /Windows/System32/msvcp140.dll -Destination /fluent-bit/bin/; `
 Copy-Item -Path /Windows/System32/vccorlib140.dll -Destination /fluent-bit/bin/; `
 Copy-Item -Path /Windows/System32/vcruntime140.dll -Destination /fluent-bit/bin/;
```

2. Replace those lines with the following:

```
RUN Write-Host ('Installing Visual C++ Redistributable Package'); `
 Start-Process /local/vc_redist.x64.exe -ArgumentList '/install', '/quiet', '/norestart' -NoNewWindow -Wait;

COPY msvcp140.dll /fluent-bit/bin/;
COPY vccorlib140.dll /fluent-bit/bin/;
COPY vcruntime140.dll /fluent-bit/bin/;
```

3. Save the file.
5. Re-run the Docker build command. For more information, see [Build a Windows Fluent Bit Docker Image](#) above.

## 'The remote name could not be resolved' Error When Creating the Fluent Bit Docker Container

### Symptom

At the `Installing Fluent Bit` step, you see the error `Invoke-WebRequest : The remote name`

could not be resolved: 'fluentbit.io'.

For example:

```
Step 10/16 : RUN Write-Host ('Installing Fluent Bit');
 $majorminor = ([Version]::Parse("$env:FLUENTBIT_VERSION")).ToString(2);

 Invoke-WebRequest -Uri "https://fluentbit.io/releases/$($majorminor)/td-a
gent-bit-$($env:FLUENTBIT_VERSION)-win64.zip" -OutFile td-agent-bit.zip;
 Expand-Archive -Path td-agent-bit.zip -Destination /local/fluent-bit;

 Move-Item -Path /local/fluent-bit/*/* -Destination /fluent-bit/;

---> Running in f56e62f47fb5

Installing Fluent Bit

Invoke-WebRequest : The remote name could not be resolved: 'fluentbit.io'

At line:1 char:184

+ ... oString(2); Invoke-WebRequest -Uri https://fluentbit.io/releases/$($m ...
+
+ ~~~~~
+ CategoryInfo : InvalidOperation: (System.Net.HttpWebRequest:HttpWebRequ
est) [Invoke-WebRequest], WebException

+ FullyQualifiedErrorId : WebCmdletWebResponseException,Microsoft.PowerShell.Comma
nds.InvokeWebRequestCommand

The command
 'powershell -Command $ErrorActionPreference = 'Stop';
 $ProgressPreference = 'SilentlyContinue';
 Write-Host ('Installing Fluent Bit');
 $majorminor = ([Version]::Parse("$env:FLUENTBIT_VERSION")).ToString(2);

 Invoke-WebRequest -Uri "https://fluentbit.io/releases/$($majorminor)/td-ag
ent-bit-$($env:FLUENTBIT_VERSION)-win64.zip" -OutFile td-agent-bit.zip;
 Expand-Archive -Path td-agent-bit.zip -Destination /local/fluent-bit;
 Move-Item -Path /local/fluent-bit/*/* -Destination /fluent-bit/';
returned a non-zero code: 1
```

## Description

To install Fluent Bit while building a Fluent Bit container, Docker uses `Invoke-WebRequest` to download the Fluent Bit Windows installation ZIP from fluentbit.io.

The PowerShell `Invoke-WebRequest` command is not consistently reliable within a docker container. For more information, see [docker build exception: Invoke-WebRequest : The remote name could not be resolved](#) in the Docker GitHub repository.

## Workaround

Change the script to download the Fluent Bit Windows installation ZIP using `DOCKER ADD`.

To modify the file:

1. Back up the Windows Dockerfile located at `fluent-bit/dockerfiles/Dockerfile.windows`.
2. Open the Windows Dockerfile for editing.
3. Locate the following lines:

```
RUN Write-Host ('Installing Fluent Bit'); `

$majorminor = ([Version]::Parse("$env:FLUENTBIT_VERSION")).ToString(2); `

Invoke-WebRequest -Uri "https://fluentbit.io/releases/$($majorminor)/td-agent-bit-$($env:FLUENTBIT_VERSION)-win64.zip" -OutFile td-agent-bit.zip; `

Expand-Archive -Path td-agent-bit.zip -Destination /local/fluent-bit; `

Move-Item -Path /local/fluent-bit/** -Destination /fluent-bit/;
```

4. Replace the lines with:

```
...
ARG FLUENTBIT_VERSION=1.8.0
ARG majorminor=1.8
...

ADD "https://fluentbit.io/releases/$majorminor/td-agent-bit-$FLUENTBIT_VERSION-win64.zip" /local/td-agent-bit.zip
RUN Write-Host ('Installing Fluent Bit'); `

Expand-Archive -Path td-agent-bit.zip -Destination /local/fluent-bit; `

Move-Item -Path /local/fluent-bit/** -Destination /fluent-bit/;
```

5. Save the file.
  6. Re-run the Docker build command. For more information, see [Build a Windows Fluent Bit Docker Image](#) above.
-

# Backing Up and Restoring Tanzu Kubernetes Grid Integrated Edition

This section describes how to back up and restore VMware Tanzu Kubernetes Grid Integrated Edition, including:

- [Backup and Restore Overview](#)
- [Backup and Restore Kubernetes Workloads](#)
- [Backup and Restore Kubernetes Clusters](#)
- [Backup and Restore TKGI Components](#)
- [Backup and Restore TKGI Infrastructure](#)

## Overview of Backing Up and Restoring TKGI

This topic provides an overview of the backup and restore process for TKGI, and provides high-level considerations for implementing your backup and restore strategy for TKGI.

## Layers and Tools

TKGI backup and restore comprises various layers and tools. The table summarizes these layers and tools from the top-down, that is, workload to infrastructure. Refer to the individual sections for details on performing backup and restore for that layer, as well as additional details on test scenarios.

| Layer                                   | Tool(s)                       | Comments                                                                                                                                                                                                                   |
|-----------------------------------------|-------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Backup and Restore Kubernetes Workloads | Velero, Restic                | Use Velero for stateless applications and Velero with Restic for stateful applications. Load balancer and ingress services depend on NSX-T backup.                                                                         |
| Backup and Restore Kubernetes Clusters  | BOSH Backup and Restore (BBR) | Use BBR to back up and restore Kubernetes clusters provisioned by TKGI, including the control plane nodes, etcd database, and worker nodes.                                                                                |
| Backup and Restore TKGI Components      | Ops Manager, BBR              | Use Ops Manager to back up and restore the BOSH Director and TKGI tile configurations. Use BBR to backup and restore the TKGI Management Plane virtual machines, including BOSH Director, TKGI Control Plane, and TKGI DB. |
| Backup and Restore TKGI Infrastructure  | NSX-T Manager, vCenter Server | Use the NSX Manager UI or CLI to backup and restore the NSX Manager DB. Use vCenter Server to backup and restore vCenter objects.                                                                                          |

## Considerations

When planning your backup and restore strategy and implementation for TKGI, keep in mind the following considerations.

### Develop a Backup and Restore Plan, and Test It Often

This documentation provides guidelines for implementing a robust backup and restore practice for TKGI. You must develop a plan of your own on how you intend to implement these tools and guidelines for your TKGI foundation. In addition, you must test the backup and restore of each layer of your TKGI foundation to ensure that the components and workloads are properly backed up and restored.

### Back Up Frequently and Regularly

You can only restore what is backed up. Key resources, such as stateful applications, cluster configurations, NSX-T objects, should be backed up on a frequent and regular schedule, such as once every 24 hours.

### Back Up Critical Components

Make sure you promptly back up critical components, including:

- When you deploy an application, take a backup of the application using Velero and Restic.
- When you provision a Kubernetes cluster using TKGI, take a backup of the cluster using BBR and networking objects using NSX-T.
- When you deploy, upgrade, or update TKGI components, take a backup of Ops Manager and the TKGI Management Plane using BBR.
- When you create one or more of the following NSX-T objects, take a backup using NSX-T.
  - Load balancer
  - Namespace
  - Logical switch (created when a TKGI cluster is provisioned)
  - Ingress
  - Network Policy

### Backup and Restore One Kubernetes Namespace at a Time

For optimal performance and assurance, only back up one Kubernetes namespace at a time using Velero. Likewise, you should only restore one Kubernetes namespace at a time using Velero.

### Restore What Breaks

The general approach is to restore what breaks. For example, if NSX-T crashes, you only need to restore NSX-T. If Ops Manager breaks, restore Ops Manager.

The exception is a Kubernetes cluster. If the cluster breaks, you need to restore the cluster using BBR and the applications using Velero. On the NSX-T side, you need to create a new namespace for the restored cluster. Once the cluster is restored, use `kubectl` to delete the old namespace. This will

force the creation of the NSX-T objects in the namespace. Refer to the scenarios for the TKGI cluster backup and restore for more details.

## Understand What Is Restored at Each Layer

Because there are several layers and tools involved in the backup and restore of TKGI, it can be confusing what is being backed up and restored within each layer.

**Velero** is used for backing up and restoring Kubernetes workloads. **Restic** is used with Velero for backup and restore of stateful workloads. Use Velero restore if something breaks in a workload application, and you need to restore it.

**BBR** is used to backup and restore Kubernetes clusters provisioned by TKGI. This includes the cluster nodes and etcd database. BBR might also restore stateless applications, depending on when the backup of the cluster was taken, but you shouldn't rely on it for such purposes. Use Velero for workload backup and restore.

For BBR to work restoring a cluster, the NSX-T objects need to be in the NSX-T database. BBR will only work if the objects are in NSX-T. BBR recreates the VM, not the logical switch. You need the logical switch to be present for each Kubernetes cluster.

If NSX-T crashes, you only need to restore NSX-T. Applications will continue to run, but you can't add anything.

## Backing Up and Restoring Tanzu Kubernetes Workloads Using Velero with Restic

This topic provides an overview of how to back up and restore Kubernetes workloads deployed to TKGI clusters using Velero and Restic.

## About Tanzu Kubernetes Workload Backup and Restore

Tanzu Kubernetes workload backup and restore is done using [Velero](#) and [Restic](#) software.

Velero is an open-source product that provides full backup and recovery of Kubernetes workloads. Restic is used as a companion tool with Velero to provide platform independent volume snapshots. For more information about Restic, see [Restic Integration](#) in the Velero documentation.

Together, Velero and Restic support backup and recovery for all types of Kubernetes workloads deployed to Tanzu clusters.

## Application Types

An application deployed to Kubernetes can be stateless or stateful. Velero with Restic can back up and restore both types, but the procedure might differ slightly. The examples in [Scenarios for Backing Up and Restoring TKGI Workloads](#) demonstrate the differences and process for each type.

In addition, an application might have a service attached or ingress enabled. See [Services and Ingress](#) for a summary of the available techniques for ensuring full restoration of functionality.

## Services and Ingress

If a service is attached, the IP address might need to be static for backup and restore to work as

expected.

Velero backup and restore is agnostic of NSX-T objects. To restore applications with same IP address, there are two mechanism available with TKGI:

- Create a Kubernetes load balancer service with a [static IP address](#) by specifying the `loadBalancerIP` in the YAML file.
- Deploy a Kubernetes cluster with a static IP address for pod ingress by defining [network profile](#).

## Storage Types

Tanzu Kubernetes clusters support two persistence providers: VMware Cloud Provider (vCP) and the Container Storage Interface (CSI).

vCP is VMware's Kubernetes storage plugin that allows cluster nodes on vSphere to interact with vCenter to support Kubernetes Persistent Volume objects. CSI is open-source software that enables Kubernetes implementations to provide a standard interface for storage systems, without having to create drivers for each system.

The provider is defined in the storage class. Velero with Restic backup and restore is slightly different depending on the type of persistence provider.

The VMware Cloud Provider (vCP) StorageClass provisioner only works with TKGI:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
 name: demo-sts-sc
provisioner: kubernetes.io/vsphere-volume
parameters:
 diskformat: thin
```

The Container Storage Interface (CSI) StorageClass provisioner works with TKGI, TKGM, and TKGS.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
 name: guestbook-sc-csi
 annotations:
 storageclass.kubernetes.io/is-default-class: "true"
provisioner: csi.vsphere.vmware.com
parameters:
 datastoreurl: "ds:///vmfs/volumes/vsan:52d8eb4842dbf493-41523be9cd4ff7b7/"
```

## Tanzu Kubernetes Workload Backup and Restore Requirements

This section lists the requirements for using Velero and Restic for Kubernetes workload backup and recovery.

### Cluster Configuration

To back up and recover TKGI Kubernetes workloads using Velero and Restic, TKGI installs a Velero

and a Restic Pod on each target Kubernetes cluster.

To configure your target clusters for Velero and Restic backup and recovery, complete the steps in [Modify the Host Path](#) in *Installing Velero and Restic*.

## Object Store

You need to provide an object store for Velero backups. Velero supports a number of object store providers. For more information, see [Providers](#) in the Velero documentation. Minio is an S3-compatible object store that is easy to install and use. This documentation uses the Minio server installed on a Linux VM as the backup destination for the example scenarios.

## Velero CLI

To perform Velero backup and recovery, you install the Velero CLI on a client VM. This can be the TKGI client VM or an administrator laptop.

## Velero Version

Refer to the [Release Notes](#) for the supported Velero version.

## Air-Gapped Environment

When you run the Velero CLI to install Velero on the Kubernetes cluster, Velero and Restic pods are deployed to the cluster. To install the pods, the instructions assume that the cluster environment has internet access, including the client where the Velero CLI is installed as well as the Kubernetes cluster. In case the environment has no internet access, you can use a private container registry to install Velero pods onto the target cluster. See [Air-gapped deployments](#).

## Get Started Using Velero and Restic

To get started with Velero and Restic:

- Review the scenarios in [Scenarios for Backing Up and Restoring TKGI Workloads](#).
- Complete the steps in [Installing Velero and Restic](#).

## Scenarios for Backing Up and Restoring TKGI Workloads

This section summarizes the scenarios and considerations for workload backup and restore using Velero and the Restic plugin.

## Kubernetes Workload Backup and Restore Scenarios Using Velero

The following scenarios and considerations summarize the various aspects of workload backup and restore using Velero for TKGI.

| App Type | Backup criteria | Service Type | Storage Provider | Tools | Example |
|----------|-----------------|--------------|------------------|-------|---------|
|----------|-----------------|--------------|------------------|-------|---------|

|                     |                      |                                                    |      |                |                                                     |
|---------------------|----------------------|----------------------------------------------------|------|----------------|-----------------------------------------------------|
| Stateless           | Kubernetes namespace | Load balancer with dynamic IP                      | None | Velero         | Backup and restore stateless Guestbook web app      |
| Stateless           | Kubernetes label     | Load balancer with dynamic IP                      | None | Velero         | Backup and restore stateless Guestbook web app      |
| Stateful            | Kubernetes namespace | Load balancer with dynamic IP                      | vCP  | Velero, Restic | Backup and restore stateful Guestbook web app       |
| Stateful            | Kubernetes label     | Load balancer with dynamic IP                      | vCP  | Velero, Restic | Backup and restore stateful Guestbook web app       |
| Stateful            | Kubernetes namespace | Load balancer with dynamic IP                      | CSI  | Velero, Restic | Backup and restore stateful Guestbook web app (CSI) |
| StatefulSet         | Kubernetes namespace | Headless cluster IP                                | vCP  | Velero, Restic | Backup and restore statefulset Cassandra DB         |
| StatefulSet         | Kubernetes label     | Headless cluster IP                                | vCP  | Velero, Restic | Backup and restore statefulset Cassandra DB         |
| StatefulSet         | Kubernetes namespace | Headless cluster IP                                | CSI  | Velero, Restic | Backup and restore statefulset Cassandra DB (CSI)   |
| Stateful            | Kubernetes namespace | Load balancer with static IP                       | vCP  | Velero, Restic | Backup and restore stateful Wordpress app           |
| Stateful            | Kubernetes namespace | Load balancer with static IP                       | CSI  | Velero, Restic | Backup and restore stateful Wordpress app (CSI)     |
| Stateless           | Kubernetes namespace | Ingress with static IP                             | None | Velero         | Backup and restore stateless Guestbook app          |
| Stateless, stateful | Whole cluster        | Load balancer with dynamic IP, headless cluster IP | vCP  | Velero, Restic | Backup and restore whole cluster                    |

## Installing Velero and Restic

This topic describes how to install Velero and Restic for backing up and restoring Tanzu Kubernetes Grid Integrated Edition (TKGI)-provisioned Kubernetes workloads. This topic also describes how to install MinIO for Velero with Restic backup storage.

## Prerequisites

Ensure the following before installing Velero and Restic for backing up and restoring TKGI:

- You have read: [Tanzu Kubernetes Workload Backup and Restore Requirements](#) in *Backing Up and Restoring Tanzu Kubernetes Workloads Using Velero with Restic*.
- You have a Linux VM with sufficient storage to store several workload backups. You will install MinIO on this VM. For more information, see [Quick start evaluation install with Minio](#) in the Velero documentation.
- You have a TKGI Client VM (Linux) where CLI tools are installed, such as the TKGI CLI, kubectl, and others. You will install the Velero CLI on this client VM. If you do not have such a VM, you can install the Velero CLI locally, but you must adjust the following installation steps accordingly.

- The Kubernetes environment has internet access and can be reached by the client VM. If the environment does not have internet access, refer to [Install Velero and Restic in an Air-Gapped Environment](#) below.

## Deploy an Object Store

Restic requires an object store as the backup destination for workload backups.

To deploy and configure a MinIO Server on a Linux Ubuntu VM as the Velero and Restic backend object store:

1. [Install MinIO](#)
2. [Enable MinIO as a Service](#)
3. [Create MinIO Bucket](#)
4. [Configure MinIO Bucket](#)

For more information about MinIO, see the [MinIO Quick Start Guide](#).

## Install MinIO

To install MinIO:

1. Install the MinIO app:

```
wget https://dl.min.io/server/minio/release/linux-amd64/minio
```

2. Grant execute permissions to the MinIO app:

```
chmod +x minio
```

3. Create a directory where MinIO data will be stored:

```
mkdir /DATA-MINIO
```

## Start MinIO

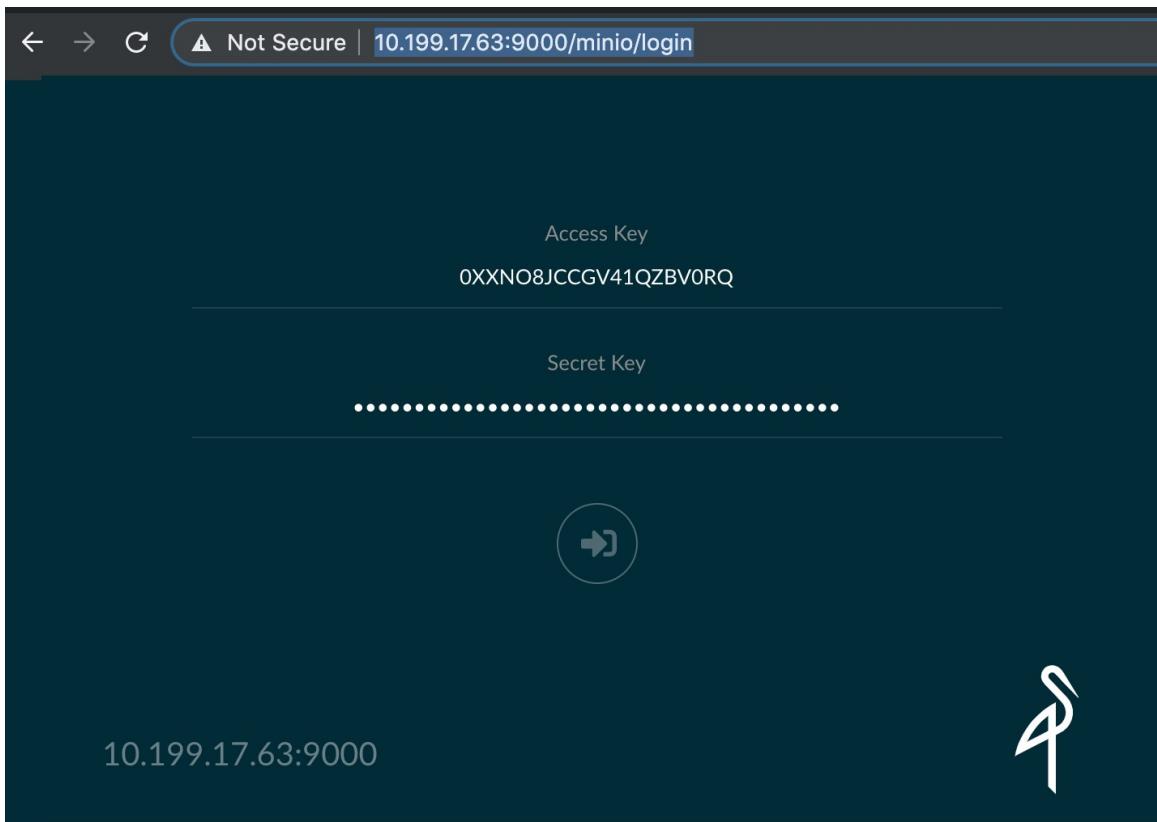
To prepare the MinIO server:

1. Start the MinIO server:

```
./minio server /DATA-MINIO
```

After the MinIO server has started, you are provided with the data store instance endpoint URL, AccessKey, and SecretKey.

2. Record the MinIO server endpoint URL, AccessKey, and SecretKey information for the data store instance.
3. Browse to the MinIO data store by opening a browser to the MinIO server endpoint URL. For example: <http://10.199.17.63:9000/minio/login/>.



[View a larger version of this image.](#)

- Log in to the MinIO server and provide the AccessKey and SecretKey.

[View a larger version of this image.](#)

## Enable MinIO as a Service

To enable MinIO as a service, configure MinIO for automatic startup:

- Download the `minio.service` script:

```
curl -O https://raw.githubusercontent.com/minio/minio-service/master/linux-systemd/minio.service
```

- Edit the `minio.service` script and add the following value for `ExecStart`:

```
ExecStart=/usr/local/bin/minio server /DATA-MINIO path
```

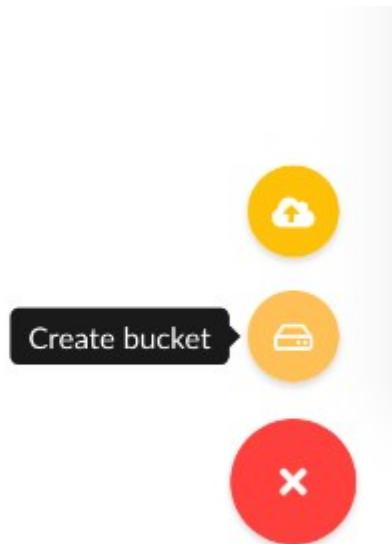
3. Save the revised script.
4. Configure the MinIO service by running the following commands:

```
cp minio.service /etc/systemd/system
cp minio /usr/local/bin/
systemctl daemon-reload
systemctl start minio
systemctl status minio
systemctl enable minio
```

## Create MinIO Bucket

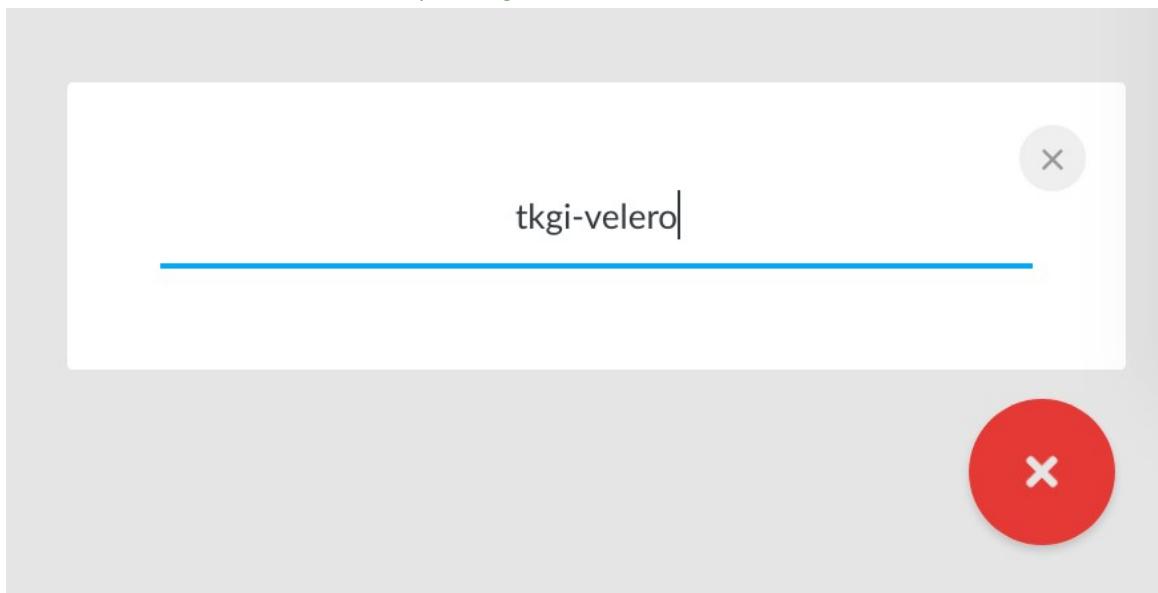
To create a MinIO bucket for TKGI workload backup and restore:

1. Launch the Mino browser and log in to your object store.
2. Click the Create Bucket icon.



[View a larger version of this image.](#)

3. Enter the bucket name, for example: `tkgi-velero`.



[View a larger version of this image.](#)

- Verify that the bucket was created.

The screenshot shows the MinIO Browser interface. On the left, there's a sidebar with a search bar labeled "Search Buckets...". Below it are two buckets listed: "tkgi-velero" and "velero". The main area is titled "tkgi-velero /" and shows a file named "Used: 309.15 KB". A table below lists files by Name, Size, and Last Modified. The table has three columns: "Name", "Size", and "Last Modified". There are three rows corresponding to the files in the bucket.

[View a larger version of this image.](#)

## Configure MinIO Bucket

By default, a new MinIO bucket is read-only, but for our Velero backup and restore the MinIO bucket must be read-write.

To set the new `tkgi-velero` bucket to read-write:

- Select the bucket and click on the dots link.
- Select **Edit Policy**.

The screenshot shows the MinIO Bucket Policies interface. It lists two buckets: "tkgi-velero" and "velero". A modal window titled "Edit policy" is open over the "tkgi-velero" bucket. Inside the modal, there are two buttons: "Edit" and "Delete".

[View a larger version of this image.](#)

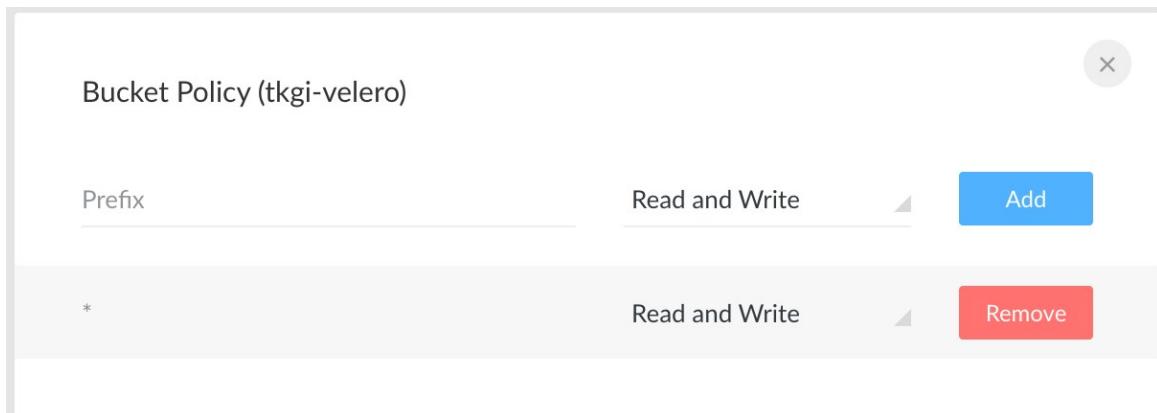
- Change the policy to `Read and Write`.

The screenshot shows the "Bucket Policy (tkgi-velero)" configuration. It has a "Prefix" input field and a "Read and Write" button. The "Read and Write" button is highlighted with a blue background.

[View a larger version of this image.](#)

- Click **Add**.

- To close the dialog box, click **X**.



[View a larger version of this image.](#)

## Install the Velero CLI on Your Workstation

To install the Velero CLI on your workstation:

- [Download the Velero CLI Binary](#)
- [Install the Velero CLI](#)

### Download the Velero CLI Binary

To download the Velero CLI Binary:

- Download the supported version of the signed Velero binary for your version of TKGI from the TKGI product downloads page at [VMware Customer Connect](#). For more information about the currently supported Velero versions, see the *Product Snapshot* section of the [Release Notes](#).



**Note:** You must use the Velero binary signed by VMware to be eligible for support from VMware.

### Install the Velero CLI

To install the Velero CLI on the TKGI client or on your local machine:

- Open a command line and change directory to the Velero CLI download.
- Unzip the download file:

```
gunzip velero-linux-v1.9.5+vmware.1.gz
```

- To check for the Velero binary:

```
ls -l
```

For example:

```
$ ls -l
```

```
-rwxrwxr-x 1 kubo kubo 69985692 Nov 14 02:55 velero-linux-v1.9.5+vmwa
re.1
```

- Grant execute permissions to the Velero CLI:

```
chmod +x velero-linux-v1.9.5+vmware.1
```

- Make the Velero CLI globally available by moving it to the system path:

```
cp velero-linux-v1.9.5+vmware.1 /usr/local/bin/velero
```

- Verify the installation:

```
velero version
```

For example:

```
$ velero version
```

```
Client: Version: v1.9.5
```

## Install Velero and Restic on the Target Kubernetes Cluster

To install the Velero and Restic pods on each Kubernetes cluster whose workloads you want to backup, complete the following:

- Prerequisites
- Set Up the `kubectl` Context
- Install Velero and Restic

## Prerequisites

The following steps require that:

- You have installed MinIO as your backup object store. For more information, see [Deploy an Object Store](#) above.
- Your Kubernetes cluster has internet access.

## Set Up the `kubectl` Context

The Velero CLI context will automatically follow the `kubectl` context. Before running Velero CLI commands to install Velero and Restic on the target cluster, set the `kubectl` context:

- Retrieve the name of the MinIO bucket. For example, `tkgi-velero`.
- Get the AccessKey and SecretKey for the MinIO bucket. For example, AccessKey: `0XXN08JCCGV41QZBV0RQ` and SecretKey: `c1Z1bf8Ljkvkmq7fHucrKCkxV39BRbcycGeXQDfx`.
- Verify `kubectl` works against the cluster. If needed, use `tkgi get-credentials`.

- Set the context for the target Kubernetes cluster so that the Velero CLI knows which cluster to work on by running:

```
tkgi get-credentials CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of the cluster. For example:

```
$ tkgi get-credentials cluster-1
```

```
Fetching credentials for cluster cluster-1. Password: ***** Context set for cluster cluster-1.
```

```
You can now switch between clusters by using: $kubectl config use-context <cluster-name>
```

You can also run `kubectl config use-context CLUSTER-NAME` to set context.

- To create a secrets file, create a file named `credentials-minio`. Update the file with the MinIO server access credentials that you collected above:

```
[default]
aws_access_key_id = ACCESS-KEY
aws_secret_access_key = SECRET-KEY
```

Where:

- `ACCESS-KEY` is the AccessKey that you collected above.
- `SECRET-KEY` is the SecretKey that you collected above.

For example:

```
[default]
aws_access_key_id = 0XXNO8JCCGV41QZBV0RQ
aws_secret_access_key = clz1bf8Ljkvkmq7fHucrKCkxV39BRbcycGeXQDfx
```

- Save the file.
- Verify that the file is in place:

```
ls
```

For example:

```
$ ls
credentials-minio
```

## Install Velero and Restic

To install Velero and Restic:

- Run the following command to install Velero and Restic on the target Kubernetes cluster:

```
velero install --image projects.registry.vmware.com/tkg/velero/velero:v1.9.5_vmware.1
--provider aws \
--plugins projects.registry.vmware.com/tkg/velero/velero-plugin-for-aws:v1.5.3_vmware.1 \
--bucket tkgi-velero \
--secret-file ./credentials-minio \
--use-volume-snapshots=false \
--use-restic \
--backup-location-config \
region=minio,s3ForcePathStyle="true",s3Url=http://10.199.17.63:9000,publicUrl=http://10.199.17.63:9000
```

For example:

```
$ velero install -image projects.registry.vmware.com/tkg/velero/velero:v1.9.5_vmware.1 -provider aws -plugins projects.registry.vmware.com/tkg/velero/velero-plugin-for-aws-v1.5.3_vmware.1

-bucket tkgi-velero -secret-file ./credentials-minio -use-volume-snapshots=false

-use-restic -backup-location-config

region=minio,s3ForcePathStyle="true",s3Url=http://10.199.17.63:9000,publicUrl=http://10.199.17.63:9000
```

CustomResourceDefinition/backups.velero.io: created ... Waiting for resources to be ready in cluster... ... DaemonSet/restic: created Velero is installed! Use 'kubectl logs deployment/velero -n velero' to view the status.

- Verify the installation of Velero and Restic:

```
kubectl logs deployment/velero -n velero
```

- Verify the `velero` namespace:

```
kubectl get ns
```

For example:

| NAME            | STATUS | AGE |
|-----------------|--------|-----|
| default         | Active | 13d |
| kube-node-lease | Active | 13d |
| kube-public     | Active | 13d |
| kube-system     | Active | 13d |
| pks-system      | Active | 13d |

|        |        |       |
|--------|--------|-------|
| velero | Active | 2m38s |
|--------|--------|-------|

- Verify the `velero` and `restic` pods.

```
kubectl get all -n velero
```

For example:

```
$ kubectl get all -n velero

NAME READY STATUS RESTARTS AGE
GE
pod/restic-25chx 0/1 CrashLoopBackOff 1 3
0s
pod/restic-rpxcp 0/1 CrashLoopBackOff 1 3
0s
pod/restic-wfxmg 0/1 CrashLoopBackOff 1 3
0s
pod/velero-8dc7498d9-9v7x4 1/1 Running 0 3
0s
```

## Modify the Host Path

To run the three-pod Restic DaemonSet on a Kubernetes cluster in TKGI, you must modify the Restic DaemonSet spec and modify the hostpath.

To modify the Restic DaemonSet:

- Verify the three-pod Restic DaemonSet:

```
kubectl get pod -n velero
```

For example:

```
$ kubectl get pod -n velero

NAME READY STATUS RESTARTS AGE
AGE
pod/restic-p5bdz 0/1 CrashLoopBackOff 4 3m8s
pod/restic-rbmnd 0/1 CrashLoopBackOff 4 3m8s
pod/restic-vcpjm 0/1 CrashLoopBackOff 4 3m8s
pod/velero-68f47744f5-1b5df 1/1 Running 0 3m8s
```

- Run the following command:

```
kubectl edit daemonset restic -n velero
```

3. Change hostPath from `/var/lib/kubelet/pods` to `/var/vcap/data/kubelet/pods`:

```
- hostPath:
 path: /var/vcap/data/kubelet/pods
```

4. Save the file.
5. To verify the three-pod Restic DaemonSet:

```
kubectl get pod -n velero
```

For example:

```
kubectl get pod -n velero

NAME READY STATUS RESTARTS AGE
restic-5jln8 1/1 Running 0 73s
restic-bpvtq 1/1 Running 0 73s
restic-vg8j7 1/1 Running 0 73s
velero-68f47744f5-1b5df 1/1 Running 0 10m
```

For information about the need to modify the Restic DaemonSet spec, see [Restic Integration in the Velero documentation](#).

## Adjust Velero Memory Limits If Necessary

If your Velero backup returns `status=InProgress` for many hours, increase the limits and requests memory settings.

To increase limits and requests memory settings:

1. Run the following command:

```
kubectl edit deployment/velero -n velero
```

2. Change the limits and request memory settings from the default of `256Mi` and `128Mi` to `512Mi` and `256Mi`:

```
ports:
- containerPort: 8085
 name: metrics
 protocol: TCP
resources:
limits:
cpu: "1"
memory: 512Mi
requests:
cpu: 500m
memory: 256Mi
terminationMessagePath: /dev/termination-log
terminationMessagePolicy: File
```

# Install Velero and Restic in an Air-Gapped Environment

If you are working in an air-gapped environment, you can install Velero and Restic using an internal registry. For more information, see [Air-gapped deployments](#) in the Velero documentation.

## Prerequisites

- A private container registry is installed and configured. The instructions use Harbor.
- Docker is installed on the workstation or TKGI jump host.
- kubectl context has been set and the MinIO `credentials-minio` file exists. For more information, see [Set Up the kubectl Context](#) above.

## Procedure

1. Open the VMware Velero downloads page for your version of TKGI linked to from the *Product Snapshot of the Release Notes*.
2. Download the Velero CLI and Velero with Restic Docker images for your version of TKGI:
  - `velero-v1.9.5+vmware.1.gz`
  - `velero-plugin-for-aws-v1.5.3_vmware.1.tar.gz`
  - `velero-restic-restore-helper-v1.9.5+vmware.1.tar.gz`



**Note:** You must use the container images signed by VMware to be eligible for support from VMware.

3. Push the Docker images into the internal registry. Adjust the variables as needed for your registry instance and preferences.

```
docker login harbor.example.com
docker load -i velero-plugin-for-aws-v1.5.3_vmware.1.tar.gz
docker tag vmware.io/velero-plugin-for-aws:v1.5.3_vmware.1 harbor.example.com/vmware-tanzu/velero-plugin-for-aws:v1.5.3_vmware.1
docker load -i velero-restic-restore-helper-v1.9.5+vmware.1.tar.gz
docker tag projects.registry.vmware.com/tkg/velero/velero-restic-restore-helper:v1.9.5_vmware.1 harbor.example.com/vmware-tanzu/velero-restic-restore-helper:v1.9.5_vmware.1
docker load -i velero-v1.9.5+vmware.1.tar.gz
docker tag projects.registry.vmware.com/tkg/velero/velero:v1.9.5_vmware.1 harbor.example.com/vmware-tanzu/velero:v1.9.5_vmware.1
docker push harbor.example.com/vmware-tanzu/velero-plugin-for-aws:v1.5.3_vmware.1
docker push harbor.example.com/vmware-tanzu/velero-restic-restore-helper:v1.9.5_vmware.1
docker push harbor.example.com/vmware-tanzu/velero:v1.9.5_vmware.1
```

4. Install Velero with Restic:

```
velero install --image harbor.example.com/vmware-tanzu/harbor.example.com/vmware-tanzu/velero:v1.9.5_vmware.1 \
--plugins harbor.example.com/vmware-tanzu/velero-plugin-for-aws:v1.5.3_vmware.1 \
\
```

```
--provider aws --bucket tkgi-velero --secret-file ./credentials-minio \
--use-volume-snapshots=false \
--backup-location-config region=minio,s3ForcePathStyle="true",s3Url=http://20.2.224.27:9000,publicUrl=http://20.20.224.27:9000 --use-restic
```

For example:

```
$ velero install -image harbor.example.com/vmware-tanzu/harbor.example.com/vmware-tanzu/velero:v1.9.5_vmware.1 -plugins harbor.example.com/vmware-tanzu/velero-plugin-for-aws:v1.5.3_vmware.1 -provider aws -bucket tkgi-velero -secret-file ./credentials-minio -use-volume-snapshots=false -backup-location-config region=minio,s3ForcePathStyle="true",s3Url=http://20.20.224.27:9000,publicUrl=http://20.20.224.27:9000 -use-restic
Velero is installed! Use 'kubectl logs deployment/velero -n velero' to view the status.
```

For more information about installing Velero and Restic, see [On-Premises Environments](#) and [Restic Integration](#) in the Velero documentation.

## 5. After installing, configure the Restic post-installation settings:

- ◊ Customize the Restic helper container and make it the [init container](#) for the pod during the restore process. You can do this by creating a configmap and applying it in the Velero namespace, for example `kubectl apply -f restic-cm.yaml -n velero`. Download the example configmap `restic-cm.yaml` provided for this purpose.
- ◊ Modify the host path by editing the Restic daemonset manifest. Replace `/var/lib/kubelet/pods` with `/var/vcap/data/kubelet/pods`. Verify that the Restic pods are running. For more information, see [Modify the Host Path](#) above.
- ◊ (Optional) Increase your Restic timeout: You can increase the Restic timeout for backups 1 TB or larger by editing the Velero deployment manifest and adding `'- --restic-timeout=900m'` to `spec.template.spec.containers`.
- ◊ (Optional) Adjust your Restic Pod CPU and memory reserves: Depending on your requirements, you can adjust the CPU and memory reserves and limits for your Velero and Restic Pods. For more information, see [Adjust Velero Memory Limits \(if necessary\)](#) above.

### **restic pod**

```
resources:
limits:
cpu: "1"
memory: 1Gi
requests:
cpu: 500m
memory: 512Mi
```

### **velero pod**

```
resources:
limits:
cpu: "1"
memory: 256Mi
requests:
```

```
cpu: 500m
memory: 128Mi
```

## Installing Velero vSphere Plugin

This topic describes how to install Velero for backing up and restoring Tanzu Kubernetes Grid Integrated Edition (TKGI)-provisioned Kubernetes workloads on vSphere.

### Prerequisites

Ensure the following before installing Velero for backing up and restoring TKGI on vSphere:

- Your clusters use the automatically installed vSphere CSI Driver. For more information, see [Deploying and Managing Cloud Native Storage \(CNS\) on vSphere](#).
- **Allow Privileged** is enabled in the plan for the cluster being backed up. For more information, see [Plans in Installing Tanzu Kubernetes Grid Integrated Edition on vSphere](#).
- You have read: [Tanzu Kubernetes Workload Backup and Restore Requirements](#) in *Backing Up and Restoring Tanzu Kubernetes Workloads Using Velero with Restic*.
- You have a Linux VM with sufficient storage to store several workload backups. You will install MinIO on this VM. For more information, see [Quick start evaluation install with Minio](#) in the Velero documentation.
- You have a TKGI Client VM (Linux) where CLI tools are installed, such as the TKGI CLI, kubectl, and others. You will install the Velero CLI on this client VM. If you do not have such a VM, you can install the Velero CLI locally, but you must adjust the following installation steps accordingly.
- The Kubernetes environment has internet access and can be reached by the client VM. If the environment does not have internet access, refer to [Install Velero in an Air-Gapped Environment](#) below.

### Deploy an Object Store

The Velero backup procedure requires an object store as the backup destination for workload backups. Deploy and configure a MinIO Server on a Linux Ubuntu VM as the Velero backend object store. For more information, see [Deploy an Object Store](#).

### Install the Velero CLI on Your Workstation

To install the Velero CLI on your workstation:

1. [Download the Velero CLI Binary](#)
2. [Install the Velero CLI](#)

### Download the Velero CLI Binary

To download the Velero CLI Binary:

1. Download the supported version of the signed Velero binary for your version of TKGI from

the TKGI product downloads page at myVMware. For more information about the currently supported Velero versions, see the *Product Snapshot* section of the [Release Notes](#).



**Note:** You must use the Velero binary signed by VMware to be eligible for support from VMware.

## Install the Velero CLI

To install the Velero CLI on the TKGI client or on your local machine:

1. Open a command line and change directory to the Velero CLI download.
2. Unzip the download file:

```
gunzip velero-linux-v1.9.5+vmware.1.gz
```

3. Grant execute permissions to the Velero CLI:

```
chmod +x velero-linux-v1.9.5+vmware.1
```

4. Make the Velero CLI globally available by moving it to the system path:

```
cp velero-linux-v1.9.5+vmware.1 /usr/local/bin/velero
```

5. Verify the installation:

```
velero version
```

For example:

```
$ velero version
```

```
Client:
```

```
Version: v1.9.5
```

## Install Velero on the Target Kubernetes Cluster

To install the Velero pod on each Kubernetes cluster whose workloads you want to backup, complete the following:

1. [Prerequisites](#)
2. [Set Up the kubectl Context](#)
3. [Install Velero](#)
4. [Create a Velero vSphere Credential Secret](#)
5. [Create the Velero vSphere Plugin Configuration File](#)
6. [Install Velero vSphere Plugin](#)
7. [Back up the VCP Volumes Migrated to vSphere CSI Driver](#)
8. [Adjust Velero Memory Limits If Necessary](#)

## Prerequisites

The following steps require that:

- You have installed MinIO as your backup object store. For more information, see [Deploy an Object Store](#) above.
- Your Kubernetes cluster has internet access.

## Set Up the kubectl Context

The Velero CLI context will automatically follow the kubectl context. Before running Velero CLI commands to install Velero on the target cluster, set the kubectl context:

1. Retrieve the name of the MinIO bucket. For example, `tkgi-velero`.
2. Get the AccessKey and SecretKey for the MinIO bucket. For example, AccessKey: `0XXN08JCCGV41QZBV0RQ` and SecretKey: `c1Z1bf8Ljkvkmq7fHucrKCkxV39BRbcycGeXQDfx`.
3. Verify `kubectl` works against the cluster. If needed, use `tkgi get-credentials`.
4. Set the context for the target Kubernetes cluster so that the Velero CLI knows which cluster to work:

```
tkgi get-credentials CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of the cluster.

For example:

```
$ tkgi get-credentials cluster-1

Fetching credentials for cluster cluster-1.
Password: *****
Context set for cluster cluster-1.

You can now switch between clusters by using:
$kubectl config use-context <cluster-name>
```

You can also run `kubectl config use-context CLUSTER-NAME` to set context.

5. To create a secrets file, create a file named `credentials-minio`. Update the file with the MinIO server access credentials that you collected above:

```
[default]
aws_access_key_id = ACCESS-KEY
aws_secret_access_key = SECRET-KEY
```

Where:

- `ACCESS-KEY` is the AccessKey that you collected above.
- `SECRET-KEY` is the SecretKey that you collected above.

For example:

```
[default]
aws_access_key_id = 0XXNO8JCCGV41QZBV0RQ
aws_secret_access_key = clZ1bf8Ljkvkmq7fHucrKCkxV39BRbcycGeXQDFx
```

- Save the file.

## Install Velero

- Install Velero on the target Kubernetes cluster:

```
velero install \
--image projects.registry.vmware.com/tkg/velero/velero:v1.9.5_vmware.1
--provider aws --bucket tkgi-velero \
--secret-file ./credentials-minio \
--plugins "projects.registry.vmware.com/tkg/velero/velero-plugin-for-aws:v1.5.3
_vmware.1" \
--backup-location-config region=minio,s3ForcePathStyle="true",s3Url=http://IP-A
DDRESS:PORT,publicUrl=http://IP-ADDRESS:PORT \
--snapshot-location-config region=minio
```

Where:

- IP-ADDRESS** is the IP address that is used to connect to the MinIO server.
- PORT** is the number of the port that is used to connect to the MinIO server.

For example:

```
$ velero install -image projects.registry.vmware.com/tkg/velero/velero:v1.9.5_vmware.1 -provider aws -bucket tkgi-velero -secret-file ./credentials-minio -plugins "projects.registry.vmware.com/tkg/velero/velero-plugin-for-aws:v1.5.3_vmware.1" -backup-location-config region=minio,s3ForcePathStyle="true",s3Url=http://20.20.233.44:9000,publicUrl=http://20.20.233.44:9000 -snapshot-location-config region=minio
CustomResourceDefinition/backups.velero.io: created
...
Waiting for resources to be ready in cluster...
...
Velero is installed! Use 'kubectl logs deployment/velero -n velero' to view the status.
```



**Note:** You must include the `-snapshot-location-config` region configuration parameter.

- Verify the installation of Velero:

```
kubectl logs deployment/velero -n velero
```

- Verify the `velero` namespace:

```
kubectl get ns
```

For example:

```
$ kubectl get ns

NAME STATUS AGE
default Active 13d
kube-node-lease Active 13d
kube-public Active 13d
kube-system Active 13d
pks-system Active 13d
velero Active 2m38s
```

## Create a Velero vSphere Credential Secret

1. Create the `csi-vsphere.conf` file with the following details:

```
[Global]
cluster-id = "CLUSTER-NAME"
[VirtualCenter "IP-ADDRESS"]
user = "USERNAME"
password = "PASSWORD"
port = "443"
```

Where:

- ◊ `CLUSTER-NAME` is the name of your cluster.
- ◊ `IP-ADDRESS` is the IP address of the vCenter Server.
- ◊ `USERNAME` is the username that you want to use.
- ◊ `PASSWORD` is the password that you want to use.

2. Create the secret:

```
kubectl -n NAMESPACE create secret generic velero-vsphere-config-secret --from-file=csi-vsphere.conf
```

Where `NAMESPACE` is the Velero namespace.

## Create the Velero vSphere Plugin Configuration File

1. Create a ConfigMap YAML file. For example `configmap.yaml`.
2. Modify the ConfigMap file with the following:

```
apiVersion: v1
kind: ConfigMap
metadata:
 name: velero-vsphere-plugin-config
data:
 cluster_flavor: "VANILLA"
 vsphere_secret_name: "SECRET-NAME"
 vsphere_secret_namespace: "SECRET-NAMESPACE" #optional, default is velero
```

Where:

- ◊ `SECRET-NAME` is the name you applied to your Velero secret.

- ◊ `SECRET-NAMESPACE` is the secret namespace. For example `velero`.
3. Save the ConfigMap file.
  4. Apply the ConfigMap:

```
kubectl apply -f CONFIGMAP-FILE -n SECRET-NAMESPACE
```

Where:

- ◊ `CONFIGMAP-FILE` is the name of your ConfigMap file. For example `configmap.yaml`.
- ◊ `SECRET-NAMESPACE` is the secret namespace. For example `velero`.

## Install Velero vSphere Plugin

1. Install the Velero plugin for vSphere:

```
velero plugin add projects.registry.vmware.com/tkg/velero/velero-plugin-for-vsphere:v1.4.2_vmware.1
```

2. Configure the Velero snapshot location:

```
velero snapshot-location create vsl-vsphere --provider velero.io/vsphere
```

3. Verify the `velero` pod:

```
kubectl get all -n velero
```

For example:

| NAME                                 | READY | STATUS  | RESTARTS | AG |
|--------------------------------------|-------|---------|----------|----|
| E<br>pod/velero-8dc7498d9-9v7x4<br>S | 1/1   | Running | 0        | 30 |

4. Verify the snapshot plugin:

```
velero plugin get
```

Confirm the `vsphere volumeSnapshotter` plugin is included in the returned list.

For example:

| NAME                            | KIND             |
|---------------------------------|------------------|
| velero.io/crd-remap-version     | BackupItemAction |
| velero.io/pod                   | BackupItemAction |
| velero.io/pv                    | BackupItemAction |
| velero.io/service-account       | BackupItemAction |
| velero.io/vsphere-pvc-backupper | BackupItemAction |
| velero.io/vsphere-pvc-deleter   | DeleteItemAction |

|                                    |                   |
|------------------------------------|-------------------|
| velero.io/aws                      | ObjectStore       |
| velero.io/add-pv-from-pvc          | RestoreItemAction |
| velero.io/add-pvc-from-pod         | RestoreItemAction |
| velero.io/change-pvc-node-selector | RestoreItemAction |
| velero.io/change-storage-class     | RestoreItemAction |
| velero.io/cluster-role-bindings    | RestoreItemAction |
| velero.io/crd-preserve-fields      | RestoreItemAction |
| velero.io/init-restore-hook        | RestoreItemAction |
| velero.io/job                      | RestoreItemAction |
| velero.io/pod                      | RestoreItemAction |
| velero.io/role-bindings            | RestoreItemAction |
| velero.io/service                  | RestoreItemAction |
| velero.io/service-account          | RestoreItemAction |
| velero.io/vsphere-pvc-restorer     | RestoreItemAction |
| velero.io/aws                      | VolumeSnapshotter |
| velero.io/vsphere                  | VolumeSnapshotter |

## Back up the VCP Volumes Migrated to vSphere CSI Driver

Follow this step if you want to back up the VCP volumes that were migrated to vSphere CSI Driver.

1. Create the `velero-vsphere-plugin-feature-states.yaml` ConfigMap file.
2. Modify the ConfigMap file with the following:

```
apiVersion: v1
data:
 csi-migrated-volume-support: "true"
 decouple-vsphere-csi-driver: "true"
 local-mode: "false"
kind: ConfigMap
metadata:
 name: velero-vsphere-plugin-feature-states
```

3. Save the ConfigMap file.
4. Apply the ConfigMap:

```
kubectl apply -f velero-vsphere-plugin-feature-states.yaml -n velero
```

## Adjust Velero Memory Limits If Necessary

If your Velero backup returns `status=InProgress` for many hours, increase the limits and requests memory settings. To do this:

1. Run the following command:

```
kubectl edit deployment/velero -n velero
```

2. Change the limits and request memory settings from the default of `256Mi` and `128Mi` to `512Mi` and `256Mi`:

```
ports:
```

```

- containerPort: 8085
 name: metrics
 protocol: TCP
resources:
 limits:
 cpu: "1"
 memory: 512Mi
 requests:
 cpu: 500m
 memory: 256Mi
terminationMessagePath: /dev/termination-log
terminationMessagePolicy: File

```

## Install Velero in an Air-Gapped Environment

If you are working in an air-gapped environment, you can install Velero using an internal registry. For more information, see [Air-gapped deployments](#) in the Velero documentation.

### Prerequisites

Ensure the following before installing Velero in an air-gapped environment:

- A private container registry is installed and configured. The procedure below uses a VMware Harbor Container Registry.
- Docker is installed on the workstation or TKGI jump host.
- kubectl context has been set and the MinIO `credentials-minio` file exists. For more information, see [Set Up the kubectl Context](#) above.

### Procedure

1. Open the VMware Velero downloads page for your version of TKGI linked to from the *Product Snapshot* of the [Release Notes](#).
2. Download the Velero CLI and Velero Plugin for vSphere images for your version of TKGI:

```

backup-driver-v1.4.2_vmware.1.tar.gz
data-manager-for-plugin-v1.4.2_vmware.1.tar.gz
velero-plugin-for-vsphere-v1.4.2_vmware.1.tar.gz

```



**Note:** You must use the container images signed by VMware to be eligible for support from VMware.

3. Push the Docker images into the internal registry. Adjust the variables as needed for your registry instance and preferences.

```

docker login harbor.example.com
docker load -i backup-driver-v1.4.2_vmware.1.tar.gz
docker tag vmware.io/backup-driver:v1.4.2_vmware.1 harbor.example.com/vmware-t
anzu/backup-driver:v1.4.2_vmware.1
docker load -i velero-plugin-for-vsphere-v1.4.2_vmware.1.tar.gz
docker tag vmware.io/velero-plugin-for-vsphere:v1.4.2_vmware.1 harbor.example.
com/vmware-tanzu/velero-plugin-for-vsphere:v1.4.2_vmware.1

```

```

docker load -i data-manager-for-plugin-v1.4.2_vmware.1.tar.gz
docker tag vmware.io/data-manager-for-plugin:v1.4.2_vmware.1 harbor.example.com/vmware-tanzu/data-manager-for-plugin:v1.4.2_vmware.1
docker push harbor.example.com/vmware-tanzu/backup-driver:v1.4.2_vmware.1
docker push harbor.example.com/vmware-tanzu/velero-plugin-for-vsphere:v1.4.2_vmware.1
docker push harbor.example.com/vmware-tanzu/data-manager-for-plugin:v1.4.2_vmware.1

```

#### 4. Install Velero:

```

velero install --image harbor.example.com/vmware-tanzu/velero:v1.9.5_vmware.1 \
--plugins harbor.example.com/vmware-tanzu/velero-plugin-for-aws:v1.5.3_vmware.1 \
\
--provider aws --bucket tkgi-velero --secret-file ./credentials-minio \
--backup-location-config region=minio,s3ForcePathStyle="true",s3Url=http://IP-ADDRESS:PORT,publicUrl=http://IP-ADDRESS:PORT --snapshot-location-config region=minio

```

Where:

- ◊ **IP-ADDRESS** is the IP address that is used to connect to the MinIO server.
- ◊ **PORT** is the number of the port that is used to connect to the MinIO server.

For example:

```

$ velero install -image harbor.example.com/vmware-tanzu/harbor.example.com/vmware-tanzu/velero:v1.9.5_vmware.1 -plugins harbor.example.com/vmware-tanzu/velero-plugin-for-aws:v1.5.3_vmware.1 -provider aws -bucket tkgi-velero -secret-file ./credentials-minio -backup-location-config region=minio,s3ForcePathStyle="true",s3Url=http://20.20.224.27:9000,publicUrl=http://20.20.224.27:9000 -snapshot-location-config region=minio
Velero is installed! Use 'kubectl logs deployment/velero -n velero' to view the status.

```

For more information about installing Velero, see [On-Premises Environments](#) in the Velero documentation.

5. Complete the steps in [Create the Velero vSphere Plugin Configuration File](#) above. You must create the Velero vSphere plugin configuration file before installing the Velero plugin for vSphere.
6. Install the Velero plugin for vSphere:

```

velero plugin add harbor.example.com/vmware-tanzu/velero-plugin-for-vsphere:v1.4.2_vmware.1

```

## Backup and Restore Stateless App with Namespace

This topic describes how to use Velero to backup and restore a stateless application using the namespace feature with Velero.

### Overview

The application we are going to use to test backup/restore with Velero is the standard Guestbook app (stateless app).

## Prerequisites

[Install and configure](#) Minio, Velero, and Restic.

Download the [Guestbook app YAML](#) files to a local known directory:

- redis-leader-deployment.yaml
- redis-leader-service.yaml
- redis-follower-deployment.yaml
- redis-follower-service.yaml
- frontend-deployment.yaml
- frontend-service.yaml

## Deploy Guestbook App

Create Guestbook namespace:

```
kubectl create ns guestbook
namespace/guestbook created
```

Deploy the Guestbook app:

```
kubectl apply -f . -n guestbook
deployment.apps/frontend created
service/frontend created
deployment.apps/redis-leader created
service/redis-leader created
deployment.apps/redis-follower created
service/redis-follower created
root@PKS-client-VM-WA:/DATA/K8s-Apps
```

Verify:

| kubectl get pod -n guestbook    |       |         |          |     |  |
|---------------------------------|-------|---------|----------|-----|--|
| NAME                            | READY | STATUS  | RESTARTS | AGE |  |
| frontend-6cb7f8bd65-kqbf6       | 1/1   | Running | 0        | 38s |  |
| frontend-6cb7f8bd65-s7mk9       | 1/1   | Running | 0        | 38s |  |
| frontend-6cb7f8bd65-vtz4k       | 1/1   | Running | 0        | 38s |  |
| redis-leader-7b8487bf68-2mhcd   | 1/1   | Running | 0        | 38s |  |
| redis-follower-5bdcfd74c7-4hhqs | 1/1   | Running | 0        | 37s |  |
| redis-follower-5bdcfd74c7-h18hf | 1/1   | Running | 0        | 37s |  |

| kubectl get svc -n guestbook |              |                |              |              |     |
|------------------------------|--------------|----------------|--------------|--------------|-----|
| NAME                         | TYPE         | CLUSTER-IP     | EXTERNAL-IP  | PORT(S)      | AGE |
| frontend                     | LoadBalancer | 10.100.200.3   | 10.199.41.10 | 80:32498/TCP | 47s |
| redis-leader                 | ClusterIP    | 10.100.200.192 | <none>       | 6379/TCP     | 47s |
| redis-follower               | ClusterIP    | 10.100.200.218 | <none>       | 6379/TCP     | 46s |

Access the Guestbook app at <http://10.199.41.10/>.



## Backup the Guestbook App using Namespace

This example shows how to backup and restore the Guestbook app using the `--include namespace` tag.

```
velero backup create guestbook-backup --include-namespaces guestbook

Backup request "guestbook-backup" submitted successfully.
Run `velero backup describe guestbook-backup` or `velero backup logs guestbook-backup`
for more details.
```

Verify:

```
velero backup describe guestbook-backup
```

Expect result:

```
Name: guestbook-backup
Namespace: velero
Labels: velero.io/storage-location=default
Annotations: velero.io/source-cluster-k8s-gitversion=v1.17.8+vmware.1
 velero.io/source-cluster-k8s-major-version=1
 velero.io/source-cluster-k8s-minor-version=17

Phase: Completed

Errors: 0
Warnings: 0

Namespaces:
 Included: guestbook
 Excluded: <none>

Resources:
 Included: *
```

```

Excluded: <none>
Cluster-scoped: auto

Label selector: <none>

Storage Location: default

Velero-Native Snapshot PVs: auto

TTL: 720h0m0s

Hooks: <none>

Backup Format Version: 1

Started: 2020-07-21 14:45:51 -0700 PDT
Completed: 2020-07-21 14:45:52 -0700 PDT

Expiration: 2020-08-20 14:45:51 -0700 PDT

Total items to be backed up: 60
Items backed up: 60

Velero-Native Snapshots: <none included>
```

Verify the backup that was created.

```

velero backup get

NAME STATUS ERRORS WARNINGS CREATED EXP
IRES STORAGE LOCATION SELECTOR
guestbook-backup Completed 0 0 2020-07-21 14:45:51 -0700 PDT 29d
```

Check the tkgi-velero bucket on the Minio server.

| Name                                                | Size      | Last Modified        | Actions |
|-----------------------------------------------------|-----------|----------------------|---------|
| guestbook-backup-podvolumebackups.json.gz           | 29 bytes  | Jul 21, 2020 2:45 PM | ...     |
| guestbook-backup-csi-volumesnapshotcontents.json.gz | 29 bytes  | Jul 21, 2020 2:45 PM | ...     |
| guestbook-backup-csi-volumesnapshots.json.gz        | 29 bytes  | Jul 21, 2020 2:45 PM | ...     |
| guestbook-backup-resource-list.json.gz              | 600 bytes | Jul 21, 2020 2:45 PM | ...     |
| guestbook-backup-volumesnapshots.json.gz            | 29 bytes  | Jul 21, 2020 2:45 PM | ...     |
| guestbook-backup.tar.gz                             | 13.67 KB  | Jul 21, 2020 2:45 PM | ...     |
| velero-backup.json                                  | 1.08 KB   | Jul 21, 2020 2:45 PM | ...     |
| guestbook-backup-logr.gz                            | 3.84 KB   | Jul 21, 2020 2:45 PM | ...     |

Velero writes some metadata in Kubernetes CRD.

```
kubectl get crd
```

Expected result:

| NAME                              | CREATED AT           |
|-----------------------------------|----------------------|
| backups.velero.io                 | 2020-07-21T21:04:39Z |
| backupstoragelocations.velero.io  | 2020-07-21T21:04:39Z |
| clusterlogsinks.pksapi.io         | 2020-07-07T22:55:25Z |
| clustermetricsinks.pksapi.io      | 2020-07-07T22:55:26Z |
| deletebackuprequests.velero.io    | 2020-07-21T21:04:39Z |
| downloadrequests.velero.io        | 2020-07-21T21:04:39Z |
| loadbalancers.vmware.com          | 2020-07-07T22:39:04Z |
| logssinks.pksapi.io               | 2020-07-07T22:55:25Z |
| metricsinks.pksapi.io             | 2020-07-07T22:55:26Z |
| nsxerrors.nsx.vmware.com          | 2020-07-07T22:39:04Z |
| nsxlbmonitors.vmware.com          | 2020-07-07T22:39:05Z |
| nsxlocks.nsx.vmware.com           | 2020-07-07T22:39:04Z |
| podvolumebackups.velero.io        | 2020-07-21T21:04:39Z |
| podvolumerestores.velero.io       | 2020-07-21T21:04:39Z |
| resticrepositories.velero.io      | 2020-07-21T21:04:39Z |
| restores.velero.io                | 2020-07-21T21:04:39Z |
| schedules.velero.io               | 2020-07-21T21:04:40Z |
| serverstatusrequests.velero.io    | 2020-07-21T21:04:40Z |
| volumesnapshotlocations.velero.io | 2020-07-21T21:04:40Z |

The Velero CRDs let you run certain commands, such as the following:

```
kubectl get backups.velero.io -n velero
```

| NAME             | AGE   |
|------------------|-------|
| guestbook-backup | 4m58s |

```
kubectl describe backups.velero.io guestbook-backup -n velero
```

```
Name: guestbook-backup
Namespace: velero
Labels: velero.io/storage-location=default
Annotations: velero.io/source-cluster-k8s-gitversion: v1.17.8+vmware.1
 velero.io/source-cluster-k8s-major-version: 1
 velero.io/source-cluster-k8s-minor-version: 17
API Version: velero.io/v1
Kind: Backup
Metadata:
 Creation Timestamp: 2020-07-21T21:45:51Z
 Generation: 5
 Resource Version: 3355027
 Self Link: /apis/velero.io/v1/namespaces/velero/backups/guestbook-backup
 UID: 9d71a2c4-6fdd-42a2-963d-e85a91926dac
Spec:
 Hooks:
 Included Namespaces:
 guestbook
 Storage Location: default
 Ttl: 720h0m0s
Status:
 Completion Timestamp: 2020-07-21T21:45:52Z
 Expiration: 2020-08-20T21:45:51Z
 Format Version: 1.1.0
 Phase: Completed
 Progress:
 Items Backed Up: 60
```

|                  |                      |
|------------------|----------------------|
| Total Items:     | 60                   |
| Start Timestamp: | 2020-07-21T21:45:51Z |
| Version:         | 1                    |
| Events:          | <none>               |

## Restore the Guestbook App

To test the restoration of the Guestbook app, delete it.

Delete the namespace:

```
kubectl delete ns guestbook
namespace "guestbook" deleted
```

Restore the app:

```
velero restore create --from-backup guestbook-backup

Restore request "guestbook-backup-20200721145620" submitted successfully.
Run `velero restore describe guestbook-backup-20200721145620` or `velero restore logs
guestbook-backup-20200721145620` for more details.
```

Verify that the app is restored:

```
velero restore describe guestbook-backup-20200721145620
Name: guestbook-backup-20200721145620
Namespace: velero
Labels: <none>
Annotations: <none>

Phase: Completed

Backup: guestbook-backup

Namespaces:
 Included: all namespaces found in the backup
 Excluded: <none>

Resources:
 Included: *
 Excluded: nodes, events, events.events.k8s.io, backups.velero.io, restores.velero.io, resticrepositories.velero.io
 Cluster-scoped: auto

Namespace mappings: <none>

Label selector: <none>

Restore PVs: auto
```

Run the following commands to verify:

| velero restore get              |                  |           |        |          |   |
|---------------------------------|------------------|-----------|--------|----------|---|
| NAME                            | BACKUP           | STATUS    | ERRORS | WARNINGS | C |
| CREATED                         | SELECTOR         |           |        |          |   |
| guestbook-backup-20200721145620 | guestbook-backup | Completed | 0      | 0        | 2 |

020-07-21 14:56:20 -0700 PDT &lt;none&gt;

```
kubectl get ns
```

| NAME            | STATUS | AGE |
|-----------------|--------|-----|
| default         | Active | 13d |
| guestbook       | Active | 19s |
| kube-node-lease | Active | 13d |
| kube-public     | Active | 13d |
| kube-system     | Active | 13d |
| pks-system      | Active | 13d |
| velero          | Active | 50m |

```
kubectl get pod -n guestbook
```

| NAME                            | READY | STATUS  | RESTARTS | AGE |
|---------------------------------|-------|---------|----------|-----|
| frontend-6cb7f8bd65-kqbf6       | 1/1   | Running | 0        | 29s |
| frontend-6cb7f8bd65-s7mk9       | 1/1   | Running | 0        | 29s |
| frontend-6cb7f8bd65-vtz4k       | 1/1   | Running | 0        | 29s |
| redis-leader-7b8487bf68-2mhcd   | 1/1   | Running | 0        | 29s |
| redis-follower-5bdcfd74c7-4hhqs | 1/1   | Running | 0        | 29s |
| redis-follower-5bdcfd74c7-h18hf | 1/1   | Running | 0        | 29s |

```
kubectl get svc -n guestbook
```

| NAME           | TYPE         | CLUSTER-IP     | EXTERNAL-IP  | PORT(S)      | AGE |
|----------------|--------------|----------------|--------------|--------------|-----|
| frontend       | LoadBalancer | 10.100.200.127 | 10.199.41.14 | 80:32673/TCP | 34s |
| redis-leader   | ClusterIP    | 10.100.200.78  | <none>       | 6379/TCP     | 34s |
| redis-follower | ClusterIP    | 10.100.200.250 | <none>       | 6379/TCP     | 34s |

Access the Guestbook app at <http://10.199.41.14/>.



## Guestbook

Messages

Submit

## Conclusions

Note the following:

- The namespace ‘guestbook’ is automatically re-created.

- The IP address for the Kubernetes service load balancer is changed from 10.199.41.10 to .14.

## Backup and Restore Stateless App with Label

This topic describes how to use Velero to backup and restore a stateless application using the label selector feature with Velero.

### Overview

To test Velero backup and restore, use the [stateless Guestbook application](#). This example demonstrates Velero backup and restore with [label selectors](#), so the stateless Guestbook app is deployed using labels.

### Prerequisites

[Install and configure](#) Minio, Velero, and Restic.

Download the [Guestbook app YAML files](#) to a local known directory:

- redis-leader-deployment.yaml
- redis-leader-service.yaml
- redis-follower-deployment.yaml
- redis-follower-service.yaml
- frontend-deployment.yaml
- frontend-service.yaml

### Deploy Guestbook App

Create Guestbook namespace:

```
kubectl create ns guestbook
namespace/guestbook created
```

Deploy the Guestbook app:

```
kubectl apply -f . -n guestbook

deployment.apps/frontend created
service/frontend created
deployment.apps/redis-leader created
service/redis-leader created
deployment.apps/redis-follower created
service/redis-follower created
root@PKS-client-VM-WA:/DATA/K8s-Apps
```

Verify:

```
kubectl get all -n guestbook --show-labels
```

| NAME | READY | STATUS | RESTARTS | AGE | LABELS |
|------|-------|--------|----------|-----|--------|
|------|-------|--------|----------|-----|--------|

| pod/frontend-6cb7f8bd65-571mk                | 1/1          | Running       | 0            | 18m          | app=guestbook,pod-template-hash=6cb7f8bd65,tier=frontend          |
|----------------------------------------------|--------------|---------------|--------------|--------------|-------------------------------------------------------------------|
| pod/frontend-6cb7f8bd65-92j8j                | 1/1          | Running       | 0            | 18m          | app=guestbook,pod-template-hash=6cb7f8bd65,tier=frontend          |
| pod/frontend-6cb7f8bd65-mvrqg                | 1/1          | Running       | 0            | 18m          | app=guestbook,pod-template-hash=6cb7f8bd65,tier=frontend          |
| pod/redis-leader-7b8487bf68-gmkgp            | 1/1          | Running       | 0            | 18m          | app=redis,pod-template-hash=7b8487bf68,role=leader,tier=backend   |
| pod/redis-follower-5bdcfd74c7-7jjg6          | 1/1          | Running       | 0            | 18m          | app=redis,pod-template-hash=5bdcfd74c7,role=follower,tier=backend |
| pod/redis-follower-5bdcfd74c7-mbdcr          | 1/1          | Running       | 0            | 18m          | app=redis,pod-template-hash=5bdcfd74c7,role=follower,tier=backend |
| <hr/>                                        |              |               |              |              |                                                                   |
| NAME                                         | TYPE         | CLUSTER-IP    | EXTERNAL-IP  | PORT(S)      | AGE                                                               |
| service/frontend                             | LoadBalancer | 10.100.200.64 | 10.199.41.15 | 80:30698/TCP | 18m                                                               |
| service/app=guestbook,tier=frontend          |              |               |              |              |                                                                   |
| service/redis-leader                         | ClusterIP    | 10.100.200.98 | <none>       | 6379/TCP     | 18m                                                               |
| service/app=redis,role=leader,tier=backend   |              |               |              |              |                                                                   |
| service/redis-follower                       | ClusterIP    | 10.100.200.47 | <none>       | 6379/TCP     | 18m                                                               |
| service/app=redis,role=follower,tier=backend |              |               |              |              |                                                                   |
| <hr/>                                        |              |               |              |              |                                                                   |
| NAME                                         | READY        | UP-TO-DATE    | AVAILABLE    | AGE          | LABELS                                                            |
| deployment.apps/frontend                     | 3/3          | 3             | 3            | 18m          | <none>                                                            |
| deployment.apps/redis-leader                 | 1/1          | 1             | 1            | 18m          | <none>                                                            |
| deployment.apps/redis-follower               | 2/2          | 2             | 2            | 18m          | <none>                                                            |
| <hr/>                                        |              |               |              |              |                                                                   |
| NAME                                         | DESIRED      | CURRENT       | READY        | AGE          | LABELS                                                            |
| replicaset.apps/frontend-6cb7f8bd65          | 3            | 3             | 3            | 18m          | app=guestbook,pod-template-hash=6cb7f8bd65,tier=frontend          |
| replicaset.apps/redis-leader-7b8487bf68      | 1            | 1             | 1            | 18m          | app=redis,pod-template-hash=7b8487bf68,role=leader,tier=backend   |
| replicaset.apps/redis-follower-5bdcfd74c7    | 2            | 2             | 2            | 18m          | app=redis,pod-template-hash=5bdcfd74c7,role=follower,tier=backend |

Access the Guestbook app at <http://10.199.41.15/>.



## Guestbook

Messages

Submit

hello there

## Apply a Common Label to all App Objects

By default, there is no common label that encompasses all the Kubernetes objects in the namespace ‘guestbook’ for the application. Here you apply a common label (application=guestbook) to these objects.

```
kubectl label -n guestbook pod --all application=guestbook
```

```
kubectl label -n guestbook service --all application=guestbook
```

```
kubectl label -n guestbook deployment --all application=guestbook
```

```
kubectl label -n guestbook replicaset --all application=guestbook
```

Verify: you should see all pods in the application with the same `guestbook` label:

```
kubectl get all -n guestbook --show-labels
```

## Backup the Guestbook App Using Label

When backing up an application using Velero with the `label` option, only one label can be specified in the `--selector` field. This means the label must be present on all the Kubernetes objects that belong to the application.

```
velero backup create guestbook-backup-label --selector application=guestbook

Backup request "guestbook-backup-label" submitted successfully.
Run `velero backup describe guestbook-backup-label` or `velero backup logs guestbook-backup-label` for more details.
```

Verify:

```
velero backup describe guestbook-backup-label --details
```

Expected result:

```
Name: guestbook-backup-label
Namespace: velero
Labels: velero.io/storage-location=default
Annotations: velero.io/source-cluster-k8s-gitversion=v1.17.8+vmware.1
 velero.io/source-cluster-k8s-major-version=1
 velero.io/source-cluster-k8s-minor-version=17

Phase: Completed

Errors: 0
Warnings: 0

Namespaces:
 Included: *
 Excluded: <none>
```

```

Resources:
Included: *
Excluded: <none>
Cluster-scoped: auto

Label selector: application=guestbook

Storage Location: default

Velero-Native Snapshot PVs: auto

TTL: 720h0m0s

Hooks: <none>

Backup Format Version: 1

Started: 2020-07-22 14:46:41 -0700 PDT
Completed: 2020-07-22 14:46:43 -0700 PDT

Expiration: 2020-08-21 14:46:41 -0700 PDT

Total items to be backed up: 18
Items backed up: 18

Resource List:
apps/v1/Deployment:
- guestbook/frontend
- guestbook/redis-leader
- guestbook/redis-follower
apps/v1/ReplicaSet:
- guestbook/frontend-6cb7f8bd65
- guestbook/redis-leader-7b8487bf68
- guestbook/redis-follower-5bdcfd74c7
v1/Endpoints:
- guestbook/frontend
- guestbook/redis-leader
- guestbook/redis-follower
v1/Pod:
- guestbook/frontend-6cb7f8bd65-571mk
- guestbook/frontend-6cb7f8bd65-92j8j
- guestbook/frontend-6cb7f8bd65-mvrqg
- guestbook/redis-leader-7b8487bf68-gmkgp
- guestbook/redis-follower-5bdcfd74c7-7jjg6
- guestbook/redis-follower-5bdcfd74c7-mbdcr
v1/Service:
- guestbook/frontend
- guestbook/redis-leader
- guestbook/redis-follower

Velero-Native Snapshots: <none included>
```

Verify the backup that was created.

| velero backup get |                  |          |          |                               |     | EXP |
|-------------------|------------------|----------|----------|-------------------------------|-----|-----|
| NAME              | STATUS           | ERRORS   | WARNINGS | CREATED                       |     | EXP |
| IRES              | STORAGE LOCATION | SELECTOR |          |                               |     |     |
| guestbook-backup  | Completed        | 0        | 0        | 2020-07-21 14:45:51 -0700 PDT | 29d |     |

|         |        |
|---------|--------|
| default | <none> |
|---------|--------|

Also, check the tkgi-velero bucket on the Minio server.

Velero writes some metadata in Kubernetes CRD.

```
kubectl get crd
```

The Velero CRDs let you run certain commands, such as the following:

```
kubectl get backups.velero.io -n velero

NAME AGE
guestbook-backup 24h
guestbook-backup-label 2m16s
```

```
kubectl describe backups.velero.io guestbook-backup-label -n velero

Name: guestbook-backup-label
Namespace: velero
Labels: velero.io/storage-location=default
Annotations: velero.io/source-cluster-k8s-gitversion: v1.17.8+vmware.1
 velero.io/source-cluster-k8s-major-version: 1
 velero.io/source-cluster-k8s-minor-version: 17
API Version: velero.io/v1
Kind: Backup
Metadata:
 Creation Timestamp: 2020-07-22T21:46:41Z
 Generation: 5
 Resource Version: 3597347
 Self Link: /apis/velero.io/v1/namespaces/velero/backups/guestbook-backup-label
 UID: 0a88adde-1c1a-4988-828f-886aa14fd17
Spec:
 Hooks:
 Included Namespaces:
 *
 Label Selector:
 Match Labels:
 Application: guestbook
 Storage Location: default
 Ttl: 720h0m0s
Status:
 Completion Timestamp: 2020-07-22T21:46:43Z
 Expiration: 2020-08-21T21:46:41Z
 Format Version: 1.1.0
 Phase: Completed
 Progress:
 Items Backed Up: 18
 Total Items: 18
 Start Timestamp: 2020-07-22T21:46:41Z
 Version: 1
Events: <none>
```

## Restore the Guestbook App Using Label

To test the restoration of the Guestbook app, delete the namespace:

```
kubectl delete ns guestbook
namespace "guestbook" deleted
```

Restore the Guestbook app:

```
velero restore create --from-backup guestbook-backup-label

Restore request "guestbook-backup-label-20200722145118" submitted successfully.
Run `velero restore describe guestbook-backup-label-20200722145118` or `velero restore logs guestbook-backup-label-20200722145118` for more details.
```

Verify that the app is restored:

```
velero restore describe guestbook-backup-label-20200722145118

Name: guestbook-backup-label-20200722145118
Namespace: velero
Labels: <none>
Annotations: <none>

Phase: Completed

Backup: guestbook-backup-label

Namespaces:
 Included: all namespaces found in the backup
 Excluded: <none>

Resources:
 Included: *
 Excluded: nodes, events, events.events.k8s.io, backups.velero.io, restores.velero.io, resticrepositories.velero.io
 Cluster-scoped: auto

Namespace mappings: <none>

Label selector: <none>

Restore PVs: auto
```

Run the following commands to verify:

| velero restore get |                               |                        |             |
|--------------------|-------------------------------|------------------------|-------------|
| NAME               | BACKUP                        | STATUS                 | ERRORS      |
| WARNINGS           | CREATED                       | SELECTOR               |             |
| 0                  | 2020-07-21 14:56:20 -0700 PDT | <none>                 | Completed 0 |
| 0                  | 2020-07-22 14:51:18 -0700 PDT | guestbook-backup-label | Completed 0 |

| kubectl get ns |        |     |
|----------------|--------|-----|
| NAME           | STATUS | AGE |
| default        | Active | 13d |

```
guestbook Active 19s
kube-node-lease Active 13d
kube-public Active 13d
kube-system Active 13d
pks-system Active 13d
velero Active 50m
```

```
kubectl get all -n guestbook --show-labels
```

Access the Guestbook app at <http://10.199.41.14/>.

The screenshot shows a web browser window with the URL [10.199.41.14](http://10.199.41.14) in the address bar. The page content is a simple form titled "Guestbook". It has a single input field labeled "Messages" and a blue "Submit" button below it.

## Conclusions

Note the following:

- The namespace ‘guestbook’ was automatically re-created
- The IP address for the Kubernetes service load balancer is changed from 10.199.41.10 to .14

## Backup and Restore Stateful App Using Namespace

This topic describes how to use Velero to backup and restore a stateful application using the namespace feature with Velero.

### Overview

This example demonstrates Velero backup and restore for a stateful application using namespace.

To test Velero backup and restore for stateful applications, use the [Guestbook application](#) with a persistent volume.

When restoring the stateful application using Velero, the Storage Class that was used by the PVC in the application must be present on the Kubernetes cluster. If the PVC is using the default storage class, then the default storage class must also be present prior to initiating the restore operation with Velero.

## Prerequisites

Install and configure Minio, Velero, and Restic.

Download the [Guestbook app YAML files](#) to a local known directory:

- redis-leader-deployment.yaml
- redis-leader-service.yaml
- redis-follower-deployment.yaml
- redis-follower-service.yaml
- frontend-deployment.yaml
- frontend-service.yaml

## Deploy Guestbook App

Create the default storage class named `redis-sc.yaml`:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
 name: thin-disk
 annotations:
 storageclass.kubernetes.io/is-default-class: "true"
provisioner: kubernetes.io/vsphere-volume
parameters:
 diskformat: thin
```

Apply the YAML file:

```
kubectl apply -f redis-sc.yaml
storageclass.storage.k8s.io/thin-disk created
```

Verify the storage class:

| kubectl get sc       |                              |               |                   |  |
|----------------------|------------------------------|---------------|-------------------|--|
| NAME                 | PROVISIONER                  | RECLAIMPOLICY | VOLUMEBINDINGMODE |  |
| ALLOWVOLUMEEXPANSION | AGE                          |               |                   |  |
| thin-disk (default)  | kubernetes.io/vsphere-volume | Delete        | Immediate         |  |
| false                | 4s                           |               |                   |  |

Create Guestbook namespace:

```
kubectl create ns guestbook-pv
namespace/guestbook-pv created
```

Deploy the Guestbook app:

```
kubectl apply -f . -n guestbook-pv

storageclass.storage.k8s.io/thin-disk unchanged
persistentvolumeclaim/redis-leader-claim created
persistentvolumeclaim/redis-follower-claim created
```

```
service/redis-leader created
deployment.apps/redis-leader created
service/redis-follower created
deployment.apps/redis-follower created
service/frontend created
deployment.apps/frontend created
```

Verify:

```
kubectl get all -n guestbook-pv
```

| NAME                            | READY | STATUS  | RESTARTS | AGE |
|---------------------------------|-------|---------|----------|-----|
| frontend-6cb7f8bd65-kqbf6       | 1/1   | Running | 0        | 38s |
| frontend-6cb7f8bd65-s7mk9       | 1/1   | Running | 0        | 38s |
| frontend-6cb7f8bd65-vtz4k       | 1/1   | Running | 0        | 38s |
| redis-leader-7b8487bf68-2mhcd   | 1/1   | Running | 0        | 38s |
| redis-follower-5bdcfd74c7-4hhqs | 1/1   | Running | 0        | 37s |
| redis-follower-5bdcfd74c7-h18hf | 1/1   | Running | 0        | 37s |

```
kubectl get pvc,pv -n guestbook-pv
```

| NAME                                                      | CAPACITY       | ACCESS MODES | STORAGECLASS                      | STATUS   | VOLUME                             | AGE   |
|-----------------------------------------------------------|----------------|--------------|-----------------------------------|----------|------------------------------------|-------|
| persistentvolumeclaim/redis-leader-claim                  | 2Gi            | RWO          | thin-disk                         | Bound    | pvc-e0f3b455-e66e-45e6-8f77-4ace5a | 2m31s |
| persistentvolumeclaim/redis-follower-claim                | 2Gi            | RWO          | thin-disk                         | Bound    | pvc-672ab412-1167-4110-87cc-51c2bf | 2m31s |
| <hr/>                                                     |                |              |                                   |          |                                    |       |
| NAME                                                      | RECLAIM POLICY | STATUS       | CLAIM                             | CAPACITY | ACCESS MODES                       | AGE   |
| persistentvolume/pvc-672ab412-1167-4110-87cc-51c2bf       | Delete         | Bound        | guestbook-pv/redis-follower-claim | 2Gi      | RWO                                | 2m30s |
| persistentvolume/pvc-e0f3b455-e66e-45e6-8f77-4ace5af0ce6a | Delete         | Bound        | guestbook-pv/redis-leader-claim   | 2Gi      | RWO                                | 2m30s |

Access the Guestbook app at <http://10.199.41.14/>.

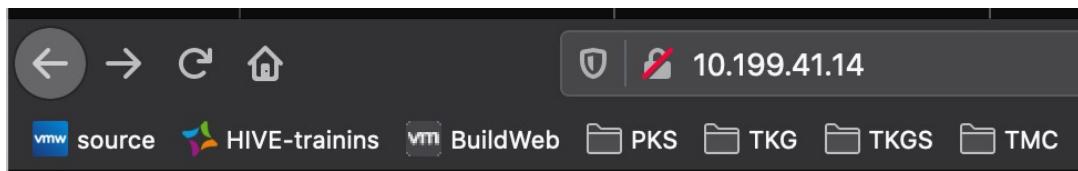


## Guestbook

Messages

Submit

Add many messages so they can be stored in the PV.



## Guestbook

### Messages

**Submit**

message 1  
message 2  
message 3  
message 4  
message 5  
message 6  
message 7  
message 8  
message 9  
message 10

## Backup the Guestbook App Using Namespace

This example shows how to backup and restore the Guestbook app using the `--include namespace` tag.

Because this app is stateful, you need to add annotations for the stateful pods with the volume name.

Get the volume names:

```
kubectl get pod -n guestbook-pv
```

| NAME                            | READY | STATUS  | RESTARTS | AGE   |
|---------------------------------|-------|---------|----------|-------|
| frontend-6cb7f8bd65-9cpdc       | 1/1   | Running | 0        | 6m27s |
| frontend-6cb7f8bd65-h2pn        | 1/1   | Running | 0        | 6m27s |
| frontend-6cb7f8bd65-kwlpr       | 1/1   | Running | 0        | 6m27s |
| frontend-6cb7f8bd65-snwl4       | 1/1   | Running | 0        | 6m27s |
| redis-leader-64fb8775bf-kbs6s   | 1/1   | Running | 0        | 6m28s |
| redis-follower-779b6d8f79-5dphr | 1/1   | Running | 0        | 6m28s |

The volume name is ‘redis-leader-data’ for pod redis-leader-64fb8775bf-kbs6s.

The volume name is ‘redis-follower-data’ for pod redis-follower-779b6d8f79-5dphr.

Annotate the pods:

```
kubectl -n guestbook-pv annotate pod redis-leader-64fb8775bf-kbs6s backup.velero.io/backup-volumes=redis-leader-data
pod/redis-leader-64fb8775bf-kbs6s annotated

kubectl -n guestbook-pv annotate pod redis-follower-779b6d8f79-5dphr backup.velero.io/backup-volumes=redis-follower-data
pod/redis-follower-779b6d8f79-5dphr annotated
```

Verify:

```
kubectl -n guestbook-pv describe pod redis-leader-64fb8775bf-kbs6s | grep Annotations
Annotations: backup.velero.io/backup-volumes: redis-leader-data

kubectl -n guestbook-pv describe pod redis-follower-779b6d8f79-5dphr | grep Annotations
Annotations: backup.velero.io/backup-volumes: redis-follower-data
velero backup create guestbook-backup --include-namespaces guestbook
```

Perform the Velero backup:

```
velero backup create guestbook-pv-backup --include-namespaces guestbook-pv
Backup request "guestbook-pv-backup" submitted successfully.
Run `velero backup describe guestbook-pv-backup` or `velero backup logs guestbook-pv-backup` for more details.
```

Verify the backup that was created.

| velero backup get   |         |           |         |          |        |          |                               |          |
|---------------------|---------|-----------|---------|----------|--------|----------|-------------------------------|----------|
| NAME                | EXPIRES | STATUS    | STORAGE | LOCATION | ERRORS | WARNINGS | CREATED                       | SELECTOR |
| guestbook-pv-backup | 29d     | Completed | default |          | 0      | 0        | 2020-07-23 16:13:46 -0700 PDT | <none>   |

Verify backup details:

```
velero backup describe guestbook-pv-backup --details
```

The Velero CRDs let you run certain commands, such as the following:

```
kubectl get backups.velero.io -n velero
NAME AGE
guestbook-backup 4m58s
```

```
kubectl describe backups.velero.io guestbook-pv-backup -n velero
```

## Restore the Guestbook App

To test the restoration of the Guestbook app, delete it.

Delete the namespace:

```
kubectl delete ns guestbook-pv
namespace "guestbook-pv" deleted
```

Verify namespace deletion:

```
kubectl get ns
```

```
kubectl get pvc,pv --all-namespaces
```

Make sure the storage class used by the application is still present:

```
kubectl get sc

NAME PROVISIONER RECLAIMPOLICY VOLUMEBINDINGMODE
thin-disk (default) kubernetes.io/vsphere-volume Delete Immediate
 false 26m
```

Restore the app:

```
velero restore create --from-backup guestbook-pv-backup

Restore request "guestbook-pv-backup-20200723161841" submitted successfully.
Run `velero restore describe guestbook-pv-backup-20200723161841` or `velero restore logs guestbook-pv-backup-20200723161841` for more details.
```

Verify that the app is restored:

```
velero restore describe guestbook-pv-backup-20200723161841

Name: guestbook-pv-backup-20200723161841
Namespace: velero
Labels: <none>
Annotations: <none>

Phase: Completed
Backup: guestbook-pv-backup

Namespaces:
 Included: all namespaces found in the backup
 Excluded: <none>

Resources:
 Included: *
 Excluded: nodes, events, events.events.k8s.io, backups.velero.io, restores.velero.io, resticrepositories.velero.io
 Cluster-scoped: auto

Namespace mappings: <none>

Label selector: <none>

Restore PVs: auto

Restic Restores (specify --details for more information):
```

```
Completed: 2
```

Run the following commands to verify:

```
velero restore get

NAME BACKUP STATUS ERRORS
WARNINGS CREATED SELECTOR
guestbook-pv-backup-20200723161841 guestbook-pv-backup Completed 0
0 2020-07-23 16:18:41 -0700 PDT <none>
```

```
kubectl get ns

NAME STATUS AGE
default Active 16d
guestbook-pv Active 76s
kube-node-lease Active 16d
kube-public Active 16d
kube-system Active 16d
pks-system Active 16d
velero Active 2d2h
```

```
kubectl get all -n guestbook-pv

NAME READY STATUS RESTARTS AGE
pod/frontend-6cb7f8bd65-9cpdc 1/1 Running 0 2m4s
pod/frontend-6cb7f8bd65-h2pnb 1/1 Running 0 2m4s
pod/frontend-6cb7f8bd65-kwlpr 1/1 Running 0 2m4s
pod/frontend-6cb7f8bd65-snwl4 1/1 Running 0 2m4s
pod/redis-leader-64fb8775bf-kbs6s 1/1 Running 0 2m4s
pod/redis-follower-779b6d8f79-5dphr 1/1 Running 0 2m4s

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
service/frontend LoadBalancer 10.100.200.144 10.199.41.16 80:30038/TCP 2m4s
service/redis-leader ClusterIP 10.100.200.156 <none> 6379/TCP 2m4s
service/redis-follower ClusterIP 10.100.200.231 <none> 6379/TCP 2m4s

NAME READY UP-TO-DATE AVAILABLE AGE
deployment.apps/frontend 4/4 4 4 2m4s
deployment.apps/redis-leader 1/1 1 1 2m4s
deployment.apps/redis-follower 1/1 1 1 2m4s

NAME DESIRED CURRENT READY AGE
replicaset.apps/frontend-6cb7f8bd65 4 4 4 2m4s
replicaset.apps/redis-leader-64fb8775bf 1 1 1 2m4s
replicaset.apps/redis-follower-779b6d8f79 1 1 1 2m4s
```

```
kubectl get pvc,pv -n guestbook-pv
```

| NAME                                     | STATUS       | VOLUME                             |     |
|------------------------------------------|--------------|------------------------------------|-----|
| CAPACITY                                 | ACCESS MODES | STORAGECLASS                       | AGE |
| persistentvolumeclaim/redis-leader-claim | Bound        | pvc-a2f6e6d4-42db-4fb8-a198-5379a2 |     |

|                                                           |        |                                   |                                          |              |
|-----------------------------------------------------------|--------|-----------------------------------|------------------------------------------|--------------|
| 552509                                                    | 2Gi    | RWO                               | thin-disk                                | 2m40s        |
| persistentvolumeclaim/redis-follower-claim                |        | Bound                             | pvc-55591938-921f-452a-b418-2cc680c0560b |              |
| <hr/>                                                     |        |                                   |                                          |              |
| NAME                                                      |        |                                   | CAPACITY                                 | ACCESS MODES |
| RECLAIM POLICY                                            | STATUS | CLAIM                             | STORAGECLASS                             | REASON AG    |
| E                                                         |        |                                   |                                          |              |
| persistentvolume/pvc-55591938-921f-452a-b418-2cc680c0560b | 2Gi    | RWO                               |                                          |              |
| Delete                                                    | Bound  | guestbook-pv/redis-follower-claim | thin-disk                                |              |
| 2m40s                                                     |        |                                   |                                          |              |
| persistentvolume/pvc-a2f6e6d4-42db-4fb8-a198-5379a2552509 | 2Gi    | RWO                               |                                          |              |
| Delete                                                    | Bound  | guestbook-pv/redis-leader-claim   | thin-disk                                | 2m           |
| 40s                                                       |        |                                   |                                          |              |

Access the Guestbook app at <http://10.199.41.16/>.



## Guestbook

Messages

Submit

message 1  
message 2  
message 3  
message 4  
message 5  
message 6  
message 7  
message 8  
message 9  
message 10

## Backup and Restore Stateful App with Label

This topic describes how to use Velero to backup and restore a stateless application using the label selector feature with Velero.

## Overview

To test Velero backup and restore, use the [stateless Guestbook application](#). This example

demonstrates Velero backup and restore with [label selectors](#), so the stateless Guestbook app is deployed using labels.

## Prerequisites

[Install and configure Minio, Velero, and Restic.](#)

Download the [Guestbook app YAML files](#) to a local known directory:

- redis-leader-deployment.yaml
- redis-leader-service.yaml
- redis-follower-deployment.yaml
- redis-follower-service.yaml
- frontend-deployment.yaml
- frontend-service.yaml

## Deploy Guestbook App

Create Guestbook namespace:

```
kubectl create ns guestbook
namespace/guestbook created
```

Deploy the Guestbook app:

```
kubectl apply -f . -n guestbook

deployment.apps/frontend created
service/frontend created
deployment.apps/redis-leader created
service/redis-leader created
deployment.apps/redis-follower created
service/redis-follower created
root@PKS-client-VM-WA:/DATA/K8s-Apps
```

Verify:

```
kubectl get all -n guestbook --show-labels

NAME READY STATUS RESTARTS AGE LABELS
pod/frontend-6cb7f8bd65-571mk 1/1 Running 0 18m app=guestbook,pod-template-hash=6cb7f8bd65,tier=frontend
pod/frontend-6cb7f8bd65-92j8j 1/1 Running 0 18m app=guestbook,pod-template-hash=6cb7f8bd65,tier=frontend
pod/frontend-6cb7f8bd65-mvrqg 1/1 Running 0 18m app=guestbook,pod-template-hash=6cb7f8bd65,tier=frontend
pod/redis-leader-7b8487bf68-gmkgp 1/1 Running 0 18m app=redis,pod-template-hash=7b8487bf68,role=leader,tier=backend
pod/redis-follower-5bdcfd74c7-7jjg6 1/1 Running 0 18m app=redis,pod-template-hash=5bdcfd74c7,role=follower,tier=backend
pod/redis-follower-5bdcfd74c7-mbdcr 1/1 Running 0 18m app=redis,pod-template-hash=5bdcfd74c7,role=follower,tier=backend
```

| NAME                                      | TYPE         | CLUSTER-IP    | EXTERNAL-IP  | PORT(S)      | AGE                                                               |
|-------------------------------------------|--------------|---------------|--------------|--------------|-------------------------------------------------------------------|
| E LABELS                                  |              |               |              |              |                                                                   |
| service/frontend                          | LoadBalancer | 10.100.200.64 | 10.199.41.15 | 80:30698/TCP | 18m                                                               |
| m app=guestbook,tier=frontend             |              |               |              |              |                                                                   |
| service/redis-leader                      | ClusterIP    | 10.100.200.98 | <none>       | 6379/TCP     | 18m                                                               |
| m app=redis,role=leader,tier=backend      |              |               |              |              |                                                                   |
| service/redis-follower                    | ClusterIP    | 10.100.200.47 | <none>       | 6379/TCP     | 18m                                                               |
| 18m app=redis,role=follower,tier=backend  |              |               |              |              |                                                                   |
| NAME                                      | READY        | UP-TO-DATE    | AVAILABLE    | AGE          | LABELS                                                            |
| deployment.apps/frontend                  | 3/3          | 3             | 3            | 18m          | <none>                                                            |
| deployment.apps/redis-leader              | 1/1          | 1             | 1            | 18m          | <none>                                                            |
| deployment.apps/redis-follower            | 2/2          | 2             | 2            | 18m          | <none>                                                            |
| NAME                                      | DESIRED      | CURRENT       | READY        | AGE          | LABELS                                                            |
| replicaset.apps/frontend-6cb7f8bd65       | 3            | 3             | 3            | 18m          | app=guestbook,pod-template-hash=6cb7f8bd65,tier=frontend          |
| replicaset.apps/redis-leader-7b8487bf68   | 1            | 1             | 1            | 18m          | app=redis,pod-template-hash=7b8487bf68,role=leader,tier=backend   |
| replicaset.apps/redis-follower-5bdcfd74c7 | 2            | 2             | 2            | 18m          | app=redis,pod-template-hash=5bdcfd74c7,role=follower,tier=backend |

Access the Guestbook app at <http://10.199.41.15/>.



## Guestbook

Messages

Submit

hello there

## Apply a Common Label to all App Objects

By default, there is no common label that encompasses all the Kubernetes objects in the namespace ‘guestbook’ for the application. Here you apply a common label (application=guestbook) to these objects.

```
kubectl label -n guestbook pod --all application=guestbook
```

```
kubectl label -n guestbook service --all application=guestbook
```

```
kubectl label -n guestbook deployment --all application=guestbook
```

```
kubectl label -n guestbook replicaset --all application=guestbook
```

Verify: you should see all pods in the application with the same `guestbook` label:

```
kubectl get all -n guestbook --show-labels
```

## Backup the Guestbook App Using Label

When backing up an application using Velero with the `label` option, only one label can be specified in the `--selector` field. This means the label must be present on all the Kubernetes objects that belong to the application.

```
velero backup create guestbook-backup-label --selector application=guestbook

Backup request "guestbook-backup-label" submitted successfully.
Run `velero backup describe guestbook-backup-label` or `velero backup logs guestbook-backup-label` for more details.
```

Verify:

```
velero backup describe guestbook-backup-label --details
```

Expected result:

```
Name: guestbook-backup-label
Namespace: velero
Labels: velero.io/storage-location=default
Annotations: velero.io/source-cluster-k8s-gitversion=v1.17.8+vmware.1
 velero.io/source-cluster-k8s-major-version=1
 velero.io/source-cluster-k8s-minor-version=17

Phase: Completed

Errors: 0
Warnings: 0

Namespaces:
 Included: *
 Excluded: <none>

Resources:
 Included: *
 Excluded: <none>
 Cluster-scoped: auto

Label selector: application=guestbook

Storage Location: default

Velero-Native Snapshot PVs: auto

TTL: 720h0m0s
```

```

Hooks: <none>

Backup Format Version: 1

Started: 2020-07-22 14:46:41 -0700 PDT
Completed: 2020-07-22 14:46:43 -0700 PDT

Expiration: 2020-08-21 14:46:41 -0700 PDT

Total items to be backed up: 18
Items backed up: 18

Resource List:
 apps/v1/Deployment:
 - guestbook/frontend
 - guestbook/redis-leader
 - guestbook/redis-follower
 apps/v1/ReplicaSet:
 - guestbook/frontend-6cb7f8bd65
 - guestbook/redis-leader-7b8487bf68
 - guestbook/redis-follower-5bdcfd74c7
 v1/Endpoints:
 - guestbook/frontend
 - guestbook/redis-leader
 - guestbook/redis-follower
 v1/Pod:
 - guestbook/frontend-6cb7f8bd65-57lmk
 - guestbook/frontend-6cb7f8bd65-92j8j
 - guestbook/frontend-6cb7f8bd65-mvrqg
 - guestbook/redis-leader-7b8487bf68-gmkgp
 - guestbook/redis-follower-5bdcfd74c7-7jjg6
 - guestbook/redis-follower-5bdcfd74c7-mbdcr
 v1/Service:
 - guestbook/frontend
 - guestbook/redis-leader
 - guestbook/redis-follower

```

Velero-Native Snapshots: <none included>

Verify the backup that was created.

| velero backup get |                  |          |          |                               |     |
|-------------------|------------------|----------|----------|-------------------------------|-----|
| NAME              | STATUS           | ERRORS   | WARNINGS | CREATED                       | EXP |
| IRES              | STORAGE LOCATION | SELECTOR |          |                               |     |
| guestbook-backup  | Completed        | 0        | 0        | 2020-07-21 14:45:51 -0700 PDT | 29d |
|                   | default          | <none>   |          |                               |     |

Also, check the tkgi-velero bucket on the Minio server.

Velero writes some metadata in Kubernetes CRD.

```
kubectl get crd
```

The Velero CRDs let you run certain commands, such as the following:

```
kubectl get backups.velero.io -n velero
```

| NAME                   | AGE   |
|------------------------|-------|
| guestbook-backup       | 24h   |
| guestbook-backup-label | 2m16s |

```
kubectl describe backups.velero.io guestbook-backup-label -n velero

Name: guestbook-backup-label
Namespace: velero
Labels: velero.io/storage-location=default
Annotations: velero.io/source-cluster-k8s-gitversion: v1.17.8+vmware.1
 velero.io/source-cluster-k8s-major-version: 1
 velero.io/source-cluster-k8s-minor-version: 17
API Version: velero.io/v1
Kind: Backup
Metadata:
 Creation Timestamp: 2020-07-22T21:46:41Z
 Generation: 5
 Resource Version: 3597347
 Self Link: /apis/velero.io/v1/namespaces/velero/backups/guestbook-backup-l
abel
 UID: 0a88adde-1c1a-4988-828f-886aa14fd17
Spec:
 Hooks:
 Included Namespaces:
 *
 Label Selector:
 Match Labels:
 Application: guestbook
 Storage Location: default
 Ttl: 720h0m0s
Status:
 Completion Timestamp: 2020-07-22T21:46:43Z
 Expiration: 2020-08-21T21:46:41Z
 Format Version: 1.1.0
 Phase: Completed
 Progress:
 Items Backed Up: 18
 Total Items: 18
 Start Timestamp: 2020-07-22T21:46:41Z
 Version: 1
Events: <none>
```

## Restore the Guestbook App Using Label

To test the restoration of the Guestbook app, delete the namespace:

```
kubectl delete ns guestbook
namespace "guestbook" deleted
```

Restore the Guestbook app:

```
velero restore create --from-backup guestbook-backup-label

Restore request "guestbook-backup-label-20200722145118" submitted successfully.
Run `velero restore describe guestbook-backup-label-20200722145118` or `velero restore
logs guestbook-backup-label-20200722145118` for more details.
```

Verify that the app is restored:

```
velero restore describe guestbook-backup-label-20200722145118

Name: guestbook-backup-label-20200722145118
Namespace: velero
Labels: <none>
Annotations: <none>

Phase: Completed

Backup: guestbook-backup-label

Namespaces:
 Included: all namespaces found in the backup
 Excluded: <none>

Resources:
 Included: *
 Excluded: nodes, events, events.events.k8s.io, backups.velero.io, restores.velero.io, resticrepositories.velero.io
 Cluster-scoped: auto

Namespace mappings: <none>

Label selector: <none>

Restore PVs: auto
```

Run the following commands to verify:

```
velero restore get

NAME BACKUP STATUS ERRORS
WARNINGS CREATED SELECTOR
guestbook-backup-20200721145620 guestbook-backup Completed 0
0 2020-07-21 14:56:20 -0700 PDT <none>
guestbook-backup-label-20200722145118 guestbook-backup-label Completed 0
0 2020-07-22 14:51:18 -0700 PDT <none>
```

```
kubectl get ns

NAME STATUS AGE
default Active 13d
guestbook Active 19s
kube-node-lease Active 13d
kube-public Active 13d
kube-system Active 13d
pks-system Active 13d
velero Active 50m
```

```
kubectl get all -n guestbook --show-labels
```

Access the Guestbook app at <http://10.199.41.14/>.



## Guestbook

Messages

Submit

## Conclusions

Note the following:

- The namespace ‘guestbook’ is automatically re-created.
- The IP address for the Kubernetes service load balancer is changed from 10.199.41.10 to .14.

## Backup and Restore Stateful App with Namespace (CSI Edition)

This topic describes how to use Velero to backup and restore a stateful application with namespace using the Container Storage Interface (CSI).

### Overview

This topic describes how to use Velero to backup and restore a stateful application with namespace using the Container Storage Interface (CSI) instead of the proprietary VMware Cloud Provider (vCP).

To test Velero backup and restore for stateful applications, use the [Guestbook application](#) with a persistent volume.

When restoring the stateful application using Velero, the Storage Class that was used by the PVC in the application must be present on the Kubernetes cluster. If the PVC is using the default storage class, then the default storage class must also be present prior to initiating the restore operation with Velero.

### Prerequisites

[Install and configure](#) Minio, Velero, and Restic.

Download the [Guestbook app YAML](#) files to a local known directory:

- redis-leader-deployment.yaml
- redis-leader-service.yaml

- redis-follower-deployment.yaml
- redis-follower-service.yaml
- frontend-deployment.yaml
- frontend-service.yaml

## Create CSI Storage Class

Create the following storage class named `guestbook-storageclass.yaml`. Note that the generic CSI provider is used ([csi.vsphere.vmware.com](https://csi.vsphere.vmware.com)), as opposed to the proprietary vCP provisioner ([kubernetes.io/vsphere-volume](https://kubernetes.io/vsphere-volume)).

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
 name: guestbook-sc-csi
 annotations:
 storageclass.kubernetes.io/is-default-class: "true"
provisioner: csi.vsphere.vmware.com
parameters:
 datastoreurl: "ds:///vmfs/volumes/vsan:52d8eb4842dbf493-41523be9cd4ff7b7/"
```

Apply the YAML file:

```
kubectl apply -f 0-guestbook-storageclass.yaml

storageclass.storage.k8s.io/guestbook-sc-csi created
```

Verify:

```
kubectl get sc

NAME PROVISIONER RECLAIMPOLICY VOLUMEBINDINGMOD
E ALLOWVOLUMEEXPANSION AGE
guestbook-sc-csi (default) csi.vsphere.vmware.com Delete Immediate
 false 4s
```

## Deploy Guestbook App

Create Guestbook namespace:

```
kubectl create ns guestbook-csi

namespace/guestbook-csi created
```

Deploy the Guestbook app:

```
kubectl apply -f . -n guestbook-csi

storageclass.storage.k8s.io/guestbook-sc-csi unchanged
persistentvolumeclaim/redis-leader-claim created
persistentvolumeclaim/redis-follower-claim created
```

```
service/redis-leader created
deployment.apps/redis-leader created
service/redis-follower created
deployment.apps/redis-follower created
service/frontend created
deployment.apps/frontend created
```

Verify:

```
kubectl get all -n guestbook-csi
```

```
kubectl get pvc,pv -n guestbook-csi
```

Access the Guestbook app at <http://10.199.41.14/>.

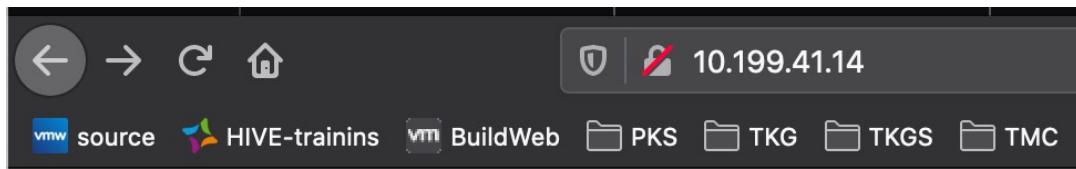


## Guestbook

Messages

Submit

Add many messages so they can be stored in the PV.



# Guestbook

Messages

**Submit**

message 1  
message 2  
message 3  
message 4  
message 5  
message 6  
message 7  
message 8  
message 9  
message 10

## Backup the Guestbook App Using Namespace

Because this app is stateful, you need to add annotations for the stateful pods with the volume name.

Get the volume names:

```
kubectl get pod -n guestbook-csi
```

The volume name is `redis-leader-data` for pod `redis-leader-64fb8775bf-wg5tk`

The volume name is `redis-follower-data` for pod `redis-follower-779b6d8f79-mqcrx`

Annotate the pods:

```
kubectl -n guestbook-csi annotate pod redis-leader-64fb8775bf-wg5tk backup.velero.io/b
ackup-volumes=redis-leader-data
pod/redis-leader-64fb8775bf-wg5tk annotated

kubectl -n guestbook-csi annotate pod redis-follower-779b6d8f79-mqcrx backup.velero.io/
backup-volumes=redis-follower-data
pod/redis-follower-779b6d8f79-mqcrx annotated
```

Verify:

```
kubectl -n guestbook-csi describe pod redis-leader-64fb8775bf-wg5tk | grep Annotations
Annotations: backup.velero.io/backup-volumes: redis-leader-data

kubectl -n guestbook-csi describe pod redis-follower-779b6d8f79-mqcrx | grep Annotations
Annotations: backup.velero.io/backup-volumes: redis-follower-data
```

Perform the Velero backup:

```
velero backup create guestbook-csi-backup --include-namespaces guestbook-csi

Backup request "guestbook-csi-backup" submitted successfully.
Run `velero backup describe guestbook-csi-backup` or `velero backup logs guestbook-csi-backup` for more details.
```

Verify the backup that was created.

```
velero backup get

NAME STATUS ERRORS WARNINGS CREATED
EXPIRES STORAGE LOCATION SELECTOR
guestbook-csi-backup Completed 0 0 2020-07-30 13:36:13 -0700 PDT
29d default <none>
```

Verify backup details:

```
velero backup describe guestbook-csi-backup --details
```

Use Velero CRD commands to further verify:

```
kubectl get backups.velero.io -n velero

NAME AGE
guestbook-csi-backup 4m58s
```

```
kubectl describe backups.velero.io guestbook-csi-backup -n velero
```

## Restore the Guestbook App

To test the restoration of the Guestbook app, delete it.

Delete the namespace:

```
kubectl delete ns guestbook-csi

namespace "guestbook-csi" deleted
```

Verify namespace deletion:

```
kubectl get ns
```

```
kubectl get pvc,pv --all-namespaces
```

Make sure the storage class used by the application is still present:

```
kubectl get sc

NAME PROVISIONER RECLAIMPOLICY VOLUMEBINDINGMOD
E ALLOWVOLUMEEXPANSION AGE
guestbook-sc-csi (default) csi.vsphere.vmware.com Delete Immediate
false 11m
```

Restore the app:

```
velero restore create --from-backup guestbook-csi-backup

Restore request "guestbook-csi-backup-20200730134254" submitted successfully.
Run `velero restore describe guestbook-csi-backup-20200730134254` or `velero restore logs guestbook-csi-backup-20200730134254` for more details.
```

Verify that the app is restored:

```
velero restore describe guestbook-csi-backup-20200730134254
```

Run the following commands to verify:

```
velero restore get

NAME BACKUP STATUS ERRORS WARN
INGS CREATED SELECTOR
guestbook-csi-backup-20200730134254 guestbook-csi-backup Completed 0 0
2020-07-30 13:42:54 -0700 PDT <none>
```

```
kubectl get ns

NAME STATUS AGE
default Active 22d
guestbook-csi Active 71s
kube-node-lease Active 22d
kube-public Active 22d
kube-system Active 22d
pks-system Active 22d
velero Active 8d
```

```
kubectl get all -n guestbook-csi
```

```
kubectl get pvc,pv -n guestbook-csi
```

Access the Guestbook app at <http://10.199.41.10/>.

## Guestbook

Messages

**Submit**

m1  
m2  
m3

## Conclusions

The Velero restore might not show the saved messages. This is not a Velero issue. The guestbook app is using some kind of RAM cache for the messages so if those messages are not flushed into disk, the data won't be available in the PV.

Key takeaways from the restore operation for this scenario:

- Pod annotation is still required for Velero backup of PV
- The namespace ‘guestbook-csi’ was automatically re-created
- The Kubernetes service LB IP has changed (from 10.199.41.14 to .16)

## Backup and Restore StatefulSet App with Namespace

This topic describes how to use Velero to backup and restore a statefulset application with namespace.

## Overview

This example demonstrates Velero backup and restore for a statefulset application with namespace. The CassandraDB app is used for demonstrating backup and restore with Velero.

When restoring a stateful application using Velero, the Storage Class that was used by the PVC in the application must be present on the Kubernetes cluster. If the PVC was using the default storage class, then the default storage class must also be present prior to initiating the restore operation with Velero.

## Prerequisites

[Install and configure](#) Minio, Velero, and Restic.

The application we are going to use is the CassandraDB statefulset app. Download the [CassandraDB YAML files](#) to a local known directory:

- cassandra-statefulset.yaml
- cassandra-storageclass.yaml
- headless-cassandra-service.yaml

## Create the Storage Class

Modify the storage class **cassandra-storageclass.yaml**:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
 name: demo-sts-sc
provisioner: kubernetes.io/vsphere-volume
parameters:
 diskformat: thin
```

Apply the storage class YAML file:

```
kubectl apply -f cassandra-storageclass.yaml

storageclass.storage.k8s.io/demo-sts-sc created
```

Verify the storage class:

```
kubectl get sc
NAME PROVISIONER RECLAIMPOLICY VOLUMEBINDINGMODE ALLOW
VOLUMEEXPANSION AGE
demo-sts-sc kubernetes.io/vsphere-volume Delete Immediate false
 3s
```

## Deploy CassandraDB App

Create the namespace:

```
kubectl create ns cassandra

namespace/cassandra created
```

Create the service:

```
kubectl apply -f headless-cassandra-service.yaml -n cassandra

service/cassandra created
```

Deploy CassandraDB app:

```
kubectl apply -f cassandra-statefulset.yaml -n cassandra

statefulset.apps/cassandra created
```

Verify:

```
kubectl get all -n cassandra

NAME READY STATUS RESTARTS AGE

```

|                            |           |            |             |         |
|----------------------------|-----------|------------|-------------|---------|
| pod/cassandra-0            | 1/1       | Running    | 0           | 5m16s   |
| pod/cassandra-1            | 1/1       | Running    | 0           | 4m25s   |
| pod/cassandra-2            | 1/1       | Running    | 0           | 2m52s   |
| NAME                       | TYPE      | CLUSTER-IP | EXTERNAL-IP | PORT(S) |
| service/cassandra          | ClusterIP | None       | <none>      | <none>  |
| NAME                       | READY     | AGE        |             |         |
| statefulset.apps/cassandra | 3/3       | 5m16s      |             |         |

|                                                                                                                                         |
|-----------------------------------------------------------------------------------------------------------------------------------------|
| kubectl get pvc,pv -n cassandra                                                                                                         |
| NAME                                                                                                                                    |
| CAPACITY ACCESS MODES STORAGECLASS AGE                                                                                                  |
| persistentvolumeclaim/cassandra-data-cassandra-0 Bound pvc-394770b3-64ba-4c35-af0c-f76a52695b25 100Gi RWO demo-sts-sc 5m46s             |
| persistentvolumeclaim/cassandra-data-cassandra-1 Bound pvc-5e0e4935-472e-46de-bba2-91d06d301f3b 100Gi RWO demo-sts-sc 4m55s             |
| persistentvolumeclaim/cassandra-data-cassandra-2 Bound pvc-cc4d55c4-b794-4a9f-871f-365ec0c3ad50 100Gi RWO demo-sts-sc 3m22s             |
| NAME CAPACITY ACCESS MODES STORAGECLASS REASON                                                                                          |
| RECLAIM POLICY STATUS CLAIM AGE                                                                                                         |
| persistentvolume/pvc-394770b3-64ba-4c35-af0c-f76a52695b25 100Gi RWO demo-sts-sc Delete Bound cassandra/cassandra-data-cassandra-0 5m46s |
| persistentvolume/pvc-5e0e4935-472e-46de-bba2-91d06d301f3b 100Gi RWO demo-sts-sc Delete Bound cassandra/cassandra-data-cassandra-1 4m54s |
| persistentvolume/pvc-cc4d55c4-b794-4a9f-871f-365ec0c3ad50 100Gi RWO demo-sts-sc Delete Bound cassandra/cassandra-data-cassandra-2 3m22s |

Make sure CassandraDB instances are fully participating in the cluster:

|                                                                                  |
|----------------------------------------------------------------------------------|
| kubectl exec -it cassandra-0 -n cassandra -- nodetool status                     |
| Datacenter: Demo-DataCenter                                                      |
| =====                                                                            |
| Status=Up/Down                                                                   |
| / State=Normal/Leaving/Joining/Moving                                            |
| -- Address Load Tokens Owns (effective) Host ID                                  |
| Rack                                                                             |
| UN 172.16.1.2 104.42 KiB 32 69.1% 6670d501-ce1a-41a6-b0ab-e5654be90d05 Demo-Rack |
| UN 172.16.1.3 75.87 KiB 32 60.8% d4c65f54-6ba0-4caa-8b0c-9a2b945cb05c Demo-Rack  |
| UN 172.16.1.4 81.09 KiB 32 70.1% fb0cca97-eb35-4e69-ad87-09ab18f739b2 Demo-Rack  |

## Create and Populate a Database and Table in Cassandra

Create the DB:

|                                                    |
|----------------------------------------------------|
| kubectl exec -it cassandra-0 -n cassandra -- cqlsh |
|----------------------------------------------------|

```
Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
```

Create and populate table:

```
cqlsh> CREATE KEYSPACE demodb WITH REPLICATION = { 'class' : 'SimpleStrategy', 'replication_factor' : 3 };
cqlsh> use demodb;
cqlsh:demodb> CREATE TABLE emp(emp_id int PRIMARY KEY, emp_name text, emp_city text, emp_sal varint,emp_phone varint);
cqlsh:demodb> INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal) VALUES (100, 'Tom', 'Cork', 999, 1000000);
cqlsh:demodb> INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal) VALUES (101, 'Andrew', 'NY', 1000, 1000000);
cqlsh:demodb> INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal) VALUES (102, 'Lara', 'Paris', 1001, 1000000);
cqlsh:demodb> select * from emp;
```

Verify:

| emp_id | emp_city | emp_name | emp_phone | emp_sal |
|--------|----------|----------|-----------|---------|
| 100    | Cork     | Tom      | 999       | 1000000 |
| 102    | Paris    | Lara     | 1001      | 1000000 |
| 101    | NY       | Andrew   | 1000      | 1000000 |

(3 rows)

Verify that the other Cassandra DB instances have the same information:

- Cassandra DB instance 1

```
kubectl exec -it cassandra-1 -n cassandra -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

- Cassandra DB instance 2

```
kubectl exec -it cassandra-2 -n cassandra -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
```

```
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

## Add Annotations

Add annotations for the stateful pods with the volume name `cassandra-data`.

```
kubectl get pod -n cassandra

NAME READY STATUS RESTARTS AGE
cassandra-0 1/1 Running 0 19m
cassandra-1 1/1 Running 0 19m
cassandra-2 1/1 Running 0 17m
```

The pods `cassandra-0`, `cassandra-1` and `cassandra-2` must be annotated with `cassandra-data`.

```
kubectl -n cassandra annotate pod/cassandra-0 backup.velero.io/backup-volumes=cassandra-data
pod/cassandra-0 annotated

kubectl -n cassandra annotate pod/cassandra-1 backup.velero.io/backup-volumes=cassandra-data
pod/cassandra-1 annotated

kubectl -n cassandra annotate pod/cassandra-2 backup.velero.io/backup-volumes=cassandra-data
pod/cassandra-2 annotated
```

Verify the annotations:

```
kubectl -n cassandra describe pod/cassandra-0 | grep Annotations
Annotations: backup.velero.io/backup-volumes: cassandra-data

kubectl -n cassandra describe pod/cassandra-1 | grep Annotations
Annotations: backup.velero.io/backup-volumes: cassandra-data

kubectl -n cassandra describe pod/cassandra-2 | grep Annotations
Annotations: backup.velero.io/backup-volumes: cassandra-data
```

## Backup the CassandraDB App using Namespace

Perform the Velero backup:

```
velero backup create cassandra-backup --include-namespaces cassandra

Backup request "cassandra-backup" submitted successfully.
Run `velero backup describe cassandra-backup` or `velero backup logs cassandra-backup` for more details.
```

### Verify the backup:

```
velero backup create cassandra-backup --include-namespaces cassandra

Backup request "cassandra-backup" submitted successfully.
Run `velero backup describe cassandra-backup` or `velero backup logs cassandra-backup` for more details.
```

```
velero backup describe cassandra-backup --details
Name: cassandra-backup
Namespace: velero
Labels: velero.io/storage-location=default
Annotations: velero.io/source-cluster-k8s-gitversion=v1.17.8+vmware.1
 velero.io/source-cluster-k8s-major-version=1
 velero.io/source-cluster-k8s-minor-version=17

Phase: Completed
...
Velero-Native Snapshots: <none included>

Restic Backups:
Completed:
 cassandra/cassandra-0: cassandra-data
 cassandra/cassandra-1: cassandra-data
 cassandra/cassandra-2: cassandra-data
```

### Get the backup:

```
velero backup get

NAME STATUS ERRORS WARNINGS CREATED EXP
IRES STORAGE LOCATION SELECTOR
cassandra-backup Completed 0 0 2020-07-29 08:26:18 -0700 PDT 29d
 default <none>
```

### Use the Velero CRD commands to verify:

```
kubectl get crd
```

```
kubectl get backups.velero.io -n velero
NAME AGE
cassandra-backup 94s
```

```
kubectl describe backups.velero.io cassandra-backup -n velero
```

## Restore the CassandraDB App

Restore the CassandraDB app from the Velero backup. Note the following about the restore operation:

- Pod annotation is still required for Velero backup of PV

- The namespace ‘cassandra’ was automatically re-created

Delete the namespace:

```
kubectl delete ns cassandra
namespace "cassandra" deleted
```

Verify that Cassandra resources are deleted:

```
kubectl get ns

NAME STATUS AGE
default Active 21d
kube-node-lease Active 21d
kube-public Active 21d
kube-system Active 21d
pks-system Active 21d
velero Active 7d18h
```

```
kubectl get pvc,pv --all-namespaces
No resources found
```

Verify that the storage class used by the application is still present:

```
kubectl get sc

NAME PROVISIONER RECLAIMPOLICY VOLUMEBINDINGMODE ALLOW
VOLUMEEXPANSION AGE
demo-sts-sc kubernetes.io/vsphere-volume Delete Immediate false
 36m
```

Restore the CassandraDB app from the backup:

```
velero restore create --from-backup cassandra-backup

Restore request "cassandra-backup-20200729083742" submitted successfully.
Run `velero restore describe cassandra-backup-20200729083742` or `velero restore logs cassandra-backup-20200729083742` for more details.
```

Verify that the app is restored:

```
velero restore describe cassandra-backup-20200729083742
Name: cassandra-backup-20200729083742
Namespace: velero
Labels: <none>
Annotations: <none>

Phase: Completed

Backup: cassandra-backup

Namespaces:
Included: all namespaces found in the backup
Excluded: <none>

Resources:
```

```

Included: *
Excluded: nodes, events, events.events.k8s.io, backups.velero.io, restores.velero.io, resticrepositories.velero.io
Cluster-scoped: auto

Namespace mappings: <none>

Label selector: <none>

Restore PVs: auto

Restic Restores (specify --details for more information):
Completed: 3

```

```
velero restore get
```

| NAME                            | BACKUP<br>CREATED            | SELECTOR                | STATUS    | ERRORS | WARNINGS | C |
|---------------------------------|------------------------------|-------------------------|-----------|--------|----------|---|
| cassandra-backup-20200729083742 | 020-07-29 08:37:42 -0700 PDT | cassandra-backup <none> | Completed | 0      | 0        | 2 |

```
kubectl get ns
```

| NAME            | STATUS | AGE   |
|-----------------|--------|-------|
| cassandra       | Active | 80s   |
| default         | Active | 21d   |
| kube-node-lease | Active | 21d   |
| kube-public     | Active | 21d   |
| kube-system     | Active | 21d   |
| pks-system      | Active | 21d   |
| velero          | Active | 7d18h |

```
kubectl get all -n cassandra
```

| NAME            | READY | STATUS  | RESTARTS | AGE  |
|-----------------|-------|---------|----------|------|
| pod/cassandra-0 | 1/1   | Running | 0        | 2m6s |
| pod/cassandra-1 | 1/1   | Running | 2        | 2m6s |
| pod/cassandra-2 | 1/1   | Running | 2        | 2m6s |

| NAME              | TYPE      | CLUSTER-IP | EXTERNAL-IP | PORT(S) | AGE  |
|-------------------|-----------|------------|-------------|---------|------|
| service/cassandra | ClusterIP | None       | <none>      | <none>  | 2m6s |

| NAME                       | READY | AGE  |
|----------------------------|-------|------|
| statefulset.apps/cassandra | 3/3   | 2m6s |

Verify the persistent volume:

```
kubectl get pvc,pv -n cassandra
```

Check the content of each Cassandra DB instance:

- Cassandra DB instance 0

```
kubectl exec -it cassandra-0 -n cassandra -- cqlsh
```

```
Connected to Demo-Cluster at 127.0.0.1:9042.
```

```
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

 emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

- Cassandra DB instance 1

```
kubectl exec -it cassandra-1 -n cassandra -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

 emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

- Cassandra DB instance 2

```
kubectl exec -it cassandra-2 -n cassandra -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

 emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

## Backup and Restore StatefulSet App with Label

This topic describes how to use Velero to backup and restore a statefulset application with label.

### Overview

This example demonstrates Velero backup and restore for a statefulset application with label

selector. The CassandraDB app is used for demonstrating stateful backup and restore with Velero.

When restoring a stateful application using Velero, the Storage Class that was used by the PVC in the application must be present on the Kubernetes cluster. If the PVC was using the default storage class, then the default storage class must also be present prior to initiating the restore operation with Velero.

## Prerequisites

[Install and configure](#) Minio, Velero, and Restic.

The application we are going to use is the CassandraDB statefulset app. Download the [CassandraDB YAML files](#) to a local known directory:

- `cassandra-statefulset.yaml`
- `cassandra-storageclass.yaml`
- `headless-cassandra-service.yaml`

## Create the Storage Class

Modify the storage class `cassandra-storageclass.yaml`:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
 name: demo-sts-sc
provisioner: kubernetes.io/vsphere-volume
parameters:
 diskformat: thin
```

Apply the storage class YAML file:

```
kubectl apply -f cassandra-storageclass.yaml

storageclass.storage.k8s.io/demo-sts-sc created
```

Verify the storage class:

| NAME        | PROVISIONER                  | RECLAIMPOLICY | VOLUMEBINDINGMODE | ALLOW |
|-------------|------------------------------|---------------|-------------------|-------|
| demo-sts-sc | kubernetes.io/vsphere-volume | Delete        | Immediate         | false |
|             | 3s                           |               |                   |       |

## Deploy CassandraDB App

By default all the objects related to CassandraDB app have the same and common label:

`app=cassandra`. Unlike the Guestbook app, you do not need to create a new label that is common to all Kubernetes objects for this app.

Create the namespace:

```
kubectl create ns cassandra
namespace/cassandra created
```

Create the service:

```
kubectl apply -f headless-cassandra-service.yaml -n cassandra
service/cassandra created
```

Deploy CassandraDB app:

```
kubectl apply -f cassandra-statefulset.yaml -n cassandra
statefulset.apps/cassandra created
```

Verify the CassandraDB app:

```
kubectl get all -n cassandra --show-labels
NAME READY STATUS RESTARTS AGE LABELS
pod/cassandra-0 1/1 Running 0 2m24s app=cassandra,controller-revision-hash=cassandra-559df6f5c,statefulset.kubernetes.io/pod-name=cassandra-0
pod/cassandra-1 1/1 Running 0 103s app=cassandra,controller-revision-hash=cassandra-559df6f5c,statefulset.kubernetes.io/pod-name=cassandra-1
pod/cassandra-2 0/1 Running 0 29s app=cassandra,controller-revision-hash=cassandra-559df6f5c,statefulset.kubernetes.io/pod-name=cassandra-2

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE LABELS
service/cassandra ClusterIP None <none> <none> 2m28s app=cassandra

NAME READY AGE LABELS
statefulset.apps/cassandra 2/3 2m24s app=cassandra
```

Verify PVC and PV:

```
kubectl get pvc,pv -n cassandra
```

Make sure CassandraDB instances are fully participating in the cluster:

```
kubectl exec -it cassandra-0 -n cassandra -- nodetool status

Datacenter: Demo-DataCenter
=====
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
-- Address Load Tokens Owns (effective) Host ID
 Rack
UN 172.16.1.2 104.41 KiB 32 68.0% 06df61ac-135c-406c-a1f1-d99
37de38b9d Demo-Rack
UN 172.16.1.3 94.96 KiB 32 75.7% a49a5c78-85be-411f-a0b9-e28f
f35fd918 Demo-Rack
UN 172.16.1.4 81.1 KiB 32 56.4% 85d124ee-0217-49e8-b56c-e137
36038f02 Demo-Rack
```

## Create and Populate a Database and Table in Cassandra

Create the DB:

```
kubectl exec -it cassandra-0 -n cassandra -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
```

Create and populate table:

```
cqlsh> CREATE KEYSPACE demodb WITH REPLICATION = { 'class' : 'SimpleStrategy', 'replication_factor' : 3 };
cqlsh> use demodb;
cqlsh:demodb> CREATE TABLE emp(emp_id int PRIMARY KEY, emp_name text, emp_city text, emp_sal varint,emp_phone varint);
cqlsh:demodb> INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal) VALUES
(100, 'Tom', 'Cork', 999, 1000000);
cqlsh:demodb> INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal) VALUES
(101, 'Andrew', 'NY', 1000, 1000000);
cqlsh:demodb> INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal) VALUES
(102, 'Lara', 'Paris', 1001, 1000000);
cqlsh:demodb> select * from emp;
```

Verify:

| emp_id | emp_city | emp_name | emp_phone | emp_sal |
|--------|----------|----------|-----------|---------|
| 100    | Cork     | Tom      | 999       | 1000000 |
| 102    | Paris    | Lara     | 1001      | 1000000 |
| 101    | NY       | Andrew   | 1000      | 1000000 |

(3 rows)

Verify that the other Cassandra DB instances have the same information:

- Cassandra DB instance 1

```
kubectl exec -it cassandra-1 -n cassandra -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

- Cassandra DB instance 2

```
kubectl exec -it cassandra-2 -n cassandra -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

## Add Annotations

Add annotations for the stateful pods with the volume name `cassandra-data`.

```
kubectl get pod -n cassandra

NAME READY STATUS RESTARTS AGE
cassandra-0 1/1 Running 0 19m
cassandra-1 1/1 Running 0 19m
cassandra-2 1/1 Running 0 17m
```

The pods `cassandra-0`, `cassandra-1` and `cassandra-2` must be annotated with `cassandra-data`.

```
kubectl -n cassandra annotate pod/cassandra-0 backup.velero.io/backup-volumes=cassandra-data
pod/cassandra-0 annotated

kubectl -n cassandra annotate pod/cassandra-1 backup.velero.io/backup-volumes=cassandra-data
pod/cassandra-1 annotated

kubectl -n cassandra annotate pod/cassandra-2 backup.velero.io/backup-volumes=cassandra-data
pod/cassandra-2 annotated
```

Verify the annotations:

```
kubectl -n cassandra describe pod/cassandra-0 | grep Annotations
Annotations: backup.velero.io/backup-volumes: cassandra-data

kubectl -n cassandra describe pod/cassandra-1 | grep Annotations
Annotations: backup.velero.io/backup-volumes: cassandra-data

kubectl -n cassandra describe pod/cassandra-2 | grep Annotations
Annotations: backup.velero.io/backup-volumes: cassandra-data
```

## Backup the CassandraDB App Using Label

Perform the Velero backup:

```
velero backup create cassandra-label-backup --selector app=cassandra
Backup request "cassandra-label-backup" submitted successfully.
```

Run `velero backup describe cassandra-label-backup` or `velero backup logs cassandra-label-backup` for more details.

Verify the backup:

```
velero backup get

NAME STATUS ERRORS WARNINGS CREATED
EXPIRES STORAGE LOCATION SELECTOR
cassandra-label-backup Completed 0 0 2020-07-29 09:48:49 -0700 PDT
29d default app=cassandra
```

View backup details:

```
velero backup describe cassandra-label-backup --details

Name: cassandra-label-backup
Namespace: velero
Labels: velero.io/storage-location=default
Annotations: velero.io/source-cluster-k8s-gitversion=v1.17.8+vmware.1
 velero.io/source-cluster-k8s-major-version=1
 velero.io/source-cluster-k8s-minor-version=17

Phase: Completed

...
Velero-Native Snapshots: <none included>

Restic Backups:
Completed:
 cassandra/cassandra-0: cassandra-data
 cassandra/cassandra-1: cassandra-data
 cassandra/cassandra-2: cassandra-data
```

Use Velero CRD commands to further verify the backup:

```
kubectl get crd
```

```
kubectl get backups.velero.io -n velero
```

| NAME                   | AGE |
|------------------------|-----|
| cassandra-label-backup | 96s |

```
kubectl describe backups.velero.io cassandra-label-backup -n velero
```

## Restore the CassandraDB App

Restore the CassandraDB app from the Velero backup. Note the following about the restore operation:

- Pod annotation is required to restore a Velero backup of a persistent volume (PV)
- The namespace ‘cassandra’ is automatically recreated during the restoration

Delete the namespace:

```
kubectl delete ns cassandra
namespace "cassandra" deleted
```

Confirm that Cassandra resources are deleted:

```
kubectl get ns

NAME STATUS AGE
default Active 21d
kube-node-lease Active 21d
kube-public Active 21d
kube-system Active 21d
pks-system Active 21d
velero Active 7d18h
```

```
kubectl get pvc,pv --all-namespaces

No resources found
```

Verify that the storage class used by the application is still present:

```
kubectl get sc

NAME PROVISIONER RECLAIMPOLICY VOLUMEBINDINGMODE ALLOW
VOLUMEEXPANSION AGE
demo-sts-sc kubernetes.io/vsphere-volume Delete Immediate false
36m
```

Restore the CassandraDB app from the Velero backup:

```
velero restore create --from-backup cassandra-label-backup

Restore request "cassandra-label-backup-20200729095415" submitted successfully.
Run `velero restore describe cassandra-label-backup-20200729095415` or `velero restore logs cassandra-label-backup-20200729095415` for more details.
```

Verify that the app is restored:

```
velero restore get

NAME BACKUP STATUS ERRORS
WARNINGS CREATED SELECTOR
cassandra-label-backup-20200729095415 cassandra-label-backup Completed 0
0 2020-07-29 09:54:15 -0700 PDT <none>
```

```
velero restore describe cassandra-label-backup-20200729095415

Name: cassandra-label-backup-20200729095415
Namespace: velero
```

```

Labels: <none>
Annotations: <none>

Phase: Completed

Backup: cassandra-label-backup

Namespaces:
 Included: all namespaces found in the backup
 Excluded: <none>

Resources:
 Included: *
 Excluded: nodes, events, events.events.k8s.io, backups.velero.io, restores.velero.io, resticrepositories.velero.io
 Cluster-scoped: auto

Namespace mappings: <none>

Label selector: <none>

Restore PVs: auto

Restic Restores (specify --details for more information):
 Completed: 3

```

```

kubectl get ns

NAME STATUS AGE
cassandra Active 79s
default Active 21d
kube-node-lease Active 21d
kube-public Active 21d
kube-system Active 21d
pks-system Active 21d
velero Active 7d19h

```

```
kubectl get all -n cassandra --show-labels
```

Verify the persistent volume:

```
kubectl get pvc,pv -n cassandra --show-labels
```

Check the content of each Cassandra DB instance:

- Cassandra DB instance 0

```

kubectl exec -it cassandra-0 -n cassandra -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+-----+

```

|     |       |        |      |         |
|-----|-------|--------|------|---------|
| 100 | Cork  | Tom    | 999  | 1000000 |
| 102 | Paris | Lara   | 1001 | 1000000 |
| 101 | NY    | Andrew | 1000 | 1000000 |

- Cassandra DB instance 1

```
kubectl exec -it cassandra-1 -n cassandra -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

- Cassandra DB instance 2

```
kubectl exec -it cassandra-2 -n cassandra -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

## Backup and Restore StatefulSet App with Namespace

This topic describes how to use Velero to back up and restore a StatefulSet application with namespace.

### Overview

This example demonstrates Velero backup and restore for a StatefulSet application with namespace. The CassandraDB app is used for demonstrating backup and restore with Velero.

When restoring a stateful application using Velero, the storage class that was used by the PersistentVolumeClaim (PVC) in the application must be present on the Kubernetes cluster. If the PVC was using the default storage class, then the default storage class must also be present prior to initiating the restore operation with Velero.

## Prerequisites

[Install and configure](#) Minio, Velero, and Restic.

The application we are going to use is the CassandraDB StatefulSet app. Download the [CassandraDB YAML files](#) to a local known directory:

- `cassandra-statefulset.yaml`
- `cassandra-storageclass.yaml`
- `headless-cassandra-service.yaml`

## Create the Storage Class

Modify the storage class **cassandra-storageclass.yaml** to use the generic Container Storage Interface (CSI) provisioner:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
 name: cassandra-sc-csi
 annotations:
 storageclass.kubernetes.io/is-default-class: "true"
provisioner: csi.vsphere.vmware.com
parameters:
 datastoreurl: "ds:///vmfs/volumes/vsan:52d8eb4842dbf493-41523be9cd4ff7b7/"
```

Apply the storage class YAML file:

```
kubectl apply -f 0-cassandra-storageclass.yaml
storageclass.storage.k8s.io/cassandra-sc-csi created
```

Verify the storage class:

```
kubectl get sc
NAME PROVISIONER RECLAIMPOLICY VOLUMEBINDINGMOD
cassandra-sc-csi (default) csi.vsphere.vmware.com Delete Immediate
false 98m
```

## Deploy CassandraDB App

Create the namespace:

```
kubectl create ns cassandra-csi
namespace/cassandra-csi created
```

Create the service:

```
kubectl apply -f 1-headless-cassandra-service.yaml -n cassandra
```

```
service/cassandra created
```

Deploy the CassandraDB app:

```
kubectl apply -f 2-cassandra-statefulset.yaml -n cassandra
statefulset.apps/cassandra created
```

Verify:

```
kubectl get all -n cassandra-csi

NAME READY STATUS RESTARTS AGE
pod/cassandra-0 1/1 Running 0 101m
pod/cassandra-1 1/1 Running 0 100m
pod/cassandra-2 1/1 Running 0 99m

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
service/cassandra ClusterIP None <none> <none> 101m

NAME READY AGE
statefulset.apps/cassandra 3/3 101m
```

```
kubectl get pvc,pv -n cassandra-csi
```

Make sure CassandraDB instances are fully participating in the cluster:

```
kubectl exec -it cassandra-0 -n cassandra-csi -- nodetool status

Datacenter: Demo-DataCenter
=====
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
-- Address Load Tokens Owns (effective) Host ID
 Rack
UN 172.16.1.2 134.43 KiB 32 55.6% f4ae812c-fd83-4268-a225-890
df1ca84d6 Demo-Rack
UN 172.16.1.3 132.68 KiB 32 73.0% deb9625e-89f5-45d0-9d0b-c8e
e3dcb610a Demo-Rack
UN 172.16.1.4 109.58 KiB 32 71.4% 4599bc2e-6f7b-4093-9397-531
84a83a92c Demo-Rack
```

## Create and Populate a Database and Table in Cassandra

Create the DB:

```
kubectl exec -it cassandra-0 -n cassandra-csi -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
```

Create and populate the table:

```
cqlsh> CREATE KEYSPACE demodb WITH REPLICATION = { 'class' : 'SimpleStrategy', 'replication_factor' : 3 };
cqlsh> use demodb;
cqlsh:demodb> CREATE TABLE emp(emp_id int PRIMARY KEY, emp_name text, emp_city text, emp_sal varint,emp_phone varint);
cqlsh:demodb> INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal) VALUES (100, 'Tom', 'Cork', 999, 1000000);
cqlsh:demodb> INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal) VALUES (101, 'Andrew', 'NY', 1000, 1000000);
cqlsh:demodb> INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal) VALUES (102, 'Lara', 'Paris', 1001, 1000000);
```

Verify:

```
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

Verify that the other Cassandra DB instances have the same information:

```
kubectl exec -it cassandra-1 -n cassandra-csi -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

```
kubectl exec -it cassandra-2 -n cassandra-csi -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

## Add Annotations

Add annotations for the stateful pods with the volume name `cassandra-data`:

```
kubectl get pod -n cassandra-csi

NAME READY STATUS RESTARTS AGE
cassandra-0 1/1 Running 0 105m
cassandra-1 1/1 Running 0 105m
cassandra-2 1/1 Running 0 103m
```

The pods `cassandra-0`, `cassandra-1` and `cassandra-2` must be annotated with `cassandra-data`.

```
kubectl -n cassandra-csi annotate pod/cassandra-0 backup.velero.io/backup-volumes=cassandra-data
pod/cassandra-0 annotated

kubectl -n cassandra-csi annotate pod/cassandra-1 backup.velero.io/backup-volumes=cassandra-data
pod/cassandra-1 annotated

kubectl -n cassandra-csi annotate pod/cassandra-2 backup.velero.io/backup-volumes=cassandra-data
pod/cassandra-2 annotated
```

Verify the annotations:

```
kubectl -n cassandra-csi describe pod/cassandra-0 | grep Annotations
Annotations: backup.velero.io/backup-volumes: cassandra-data

kubectl -n cassandra-csi describe pod/cassandra-1 | grep Annotations
Annotations: backup.velero.io/backup-volumes: cassandra-data

kubectl -n cassandra-csi describe pod/cassandra-2 | grep Annotations
Annotations: backup.velero.io/backup-volumes: cassandra-data
```

## Back Up the CassandraDB App Using Namespace

Perform the Velero backup:

```
velero backup create cassandra-csi-backup --include-namespaces cassandra-csi

Backup request "cassandra-csi-backup" submitted successfully.
Run `velero backup describe cassandra-csi-backup` or `velero backup logs cassandra-csi-backup` for more details.
```

Verify the backup:

```
velero backup get

NAME STATUS ERRORS WARNINGS CREATED
EXPIRES STORAGE LOCATION SELECTOR
cassandra-csi-backup Completed 0 0 2020-07-30 12:02:43 -0700 PDT
29d default <none>
```

Get backup details:

```
velero backup describe cassandra-csi-backup --details
```

Use the Velero CRD commands to verify:

```
kubectl get crd
```

```
kubectl get backups.velero.io -n velero
```

```
kubectl describe backups.velero.io cassandra-backup -n velero
```

## Restore the CassandraDB App

Restore the CassandraDB app from the Velero backup. Note the following about the restore operation:

- Pod annotation is still required for Velero backup of PV
- The namespace ‘cassandra’ was automatically re-created

Delete the namespace:

```
kubectl delete ns cassandra-csi
namespace "cassandra-csi" deleted
```

Verify that Cassandra resources are deleted:

```
kubectl get ns

NAME STATUS AGE
default Active 22d
kube-node-lease Active 22d
kube-public Active 22d
kube-system Active 22d
pks-system Active 22d
velero Active 8d
```

```
kubectl get pvc,pv --all-namespaces
```

```
No resources found
```

Verify that the storage class used by the application is still present:

```
kubectl get sc

NAME PROVISIONER RECLAIMPOLICY VOLUMEBINDINGMOD
E ALLOWVOLUMEEXPANSION AGE
cassandra-sc-csi (default) csi.vsphere.vmware.com Delete Immediate
false 175m
```

Restore the CassandraDB app from the backup:

```
velero restore create --from-backup cassandra-csi-backup

Restore request "cassandra-csi-backup-20200730130959" submitted successfully.
Run `velero restore describe cassandra-csi-backup-20200730130959` or `velero restore logs cassandra-csi-backup-20200730130959` for more details.
```

Verify that the app is restored:

```
velero restore get

NAME BACKUP STATUS ERRORS WARNINGS
INGS CREATED SELECTOR
cassandra-csi-backup-20200730130959 cassandra-csi-backup Completed 0 0
2020-07-30 13:09:59 -0700 PDT <none>
```

Check restoration details:

```
velero restore describe cassandra-csi-backup-20200730130959
```

Verify the namespace:

```
kubectl get ns

NAME STATUS AGE
cassandra-csi Active 75s
default Active 22d
kube-node-lease Active 22d
kube-public Active 22d
kube-system Active 22d
pks-system Active 22d
velero Active 8d
```

Verify the pods:

```
kubectl get all -n cassandra-csi

NAME READY STATUS RESTARTS AGE
pod/cassandra-0 1/1 Running 0 105s
pod/cassandra-1 0/1 Running 3 105s
pod/cassandra-2 0/1 Running 3 105s

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
service/cassandra ClusterIP None <none> <none> 105s

NAME READY AGE
statefulset.apps/cassandra 1/3 105s
```

Verify the persistent volume:

```
kubectl get pvc,pv -n cassandra-csi
```

Check the content of each Cassandra DB instance:

```
kubectl exec -it cassandra-0 -n cassandra-csi -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
```

```
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

```
kubectl exec -it cassandra-1 -n cassandra-csi -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

```
kubectl exec -it cassandra-2 -n cassandra-csi -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

## Conclusions

Key takeaways from the restore operation:

- Pod annotation is still required for Velero backup of PV
- The namespace ‘cassandra-csi’ was automatically re-created
- Restic works fine with CSI

## Backup and Restore Stateful App with Static IP for Load

## Balancer Service

This topic describes how to use Velero to backup and restore a stateful application with a load balancer service with a static IP address.

### Overview

This topic describes how to use Velero to backup and restore a Kubernetes stateful application with a service of type load balancer that uses a static IP address.

The application used to demonstrate Velero backup and restore with fixed IP is the Wordpress stateful app. By design Wordpress stores in its data structure the original IP address that was used during its initial launch. To successfully restore from backup a Wordpress app, the service of type load balancer must be deployed with a fixed IP address. This ensures the same IP will be used when a Velero restore is performed.

### Prerequisites

[Install and configure](#) Minio, Velero, and Restic.

Download the [Wordpress app YAML files](#) to a local known directory:

- mysql-deployment.yaml
- wordpress-deployment.yaml

### Configure Wordpress YAML Files

Edit the **wordpress-deployment.yaml** to include the static IP for the load balancer (`loadBalancerIP: IP`). In this example, the IP address used is taken from the NSX-T floating IP pool that is resourced for TKGI.

```
apiVersion: v1
kind: Service
metadata:
 name: wordpress
 labels:
 app: wordpress
spec:
 ports:
 - port: 80
 selector:
 app: wordpress
 tier: frontend
 type: LoadBalancer
 loadBalancerIP: 10.199.41.110
```

Create the `kustomization.yaml` file` with a value for YOUR\_PASSWORD:

```
secretGenerator:
- name: mysql-pass
 literals:
 - password=YOUR_PASSWORD
resources:
```

```
- mysql-deployment.yaml
- wordpress-deployment.yaml
```

Create the default storage class YAML `default-sc.yaml`:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
 name: default-sc
 annotations:
 storageclass.kubernetes.io/is-default-class: "true"
provisioner: kubernetes.io/vsphere-volume
parameters:
 diskformat: thin
```

Apply the storage class YAML file:

```
kubectl apply -f 0-default-sc.yaml
storageclass.storage.k8s.io/default-sc created
```

Verify the storage class:

```
kubectl get sc
NAME PROVISIONER RECLAIMPOLICY VOLUMEBINDINGMOD
E ALLOWVOLUMEEXPANSION AGE
default-sc (default) kubernetes.io/vsphere-volume Delete Immediate
false 3m38s
```

## Deploy Wordpress App

Create Wordpress namespace:

```
kubectl create ns wordpress-lbstaticip
namespace/wordpress-lbstaticip created
```

Deploy the Wordpress app:

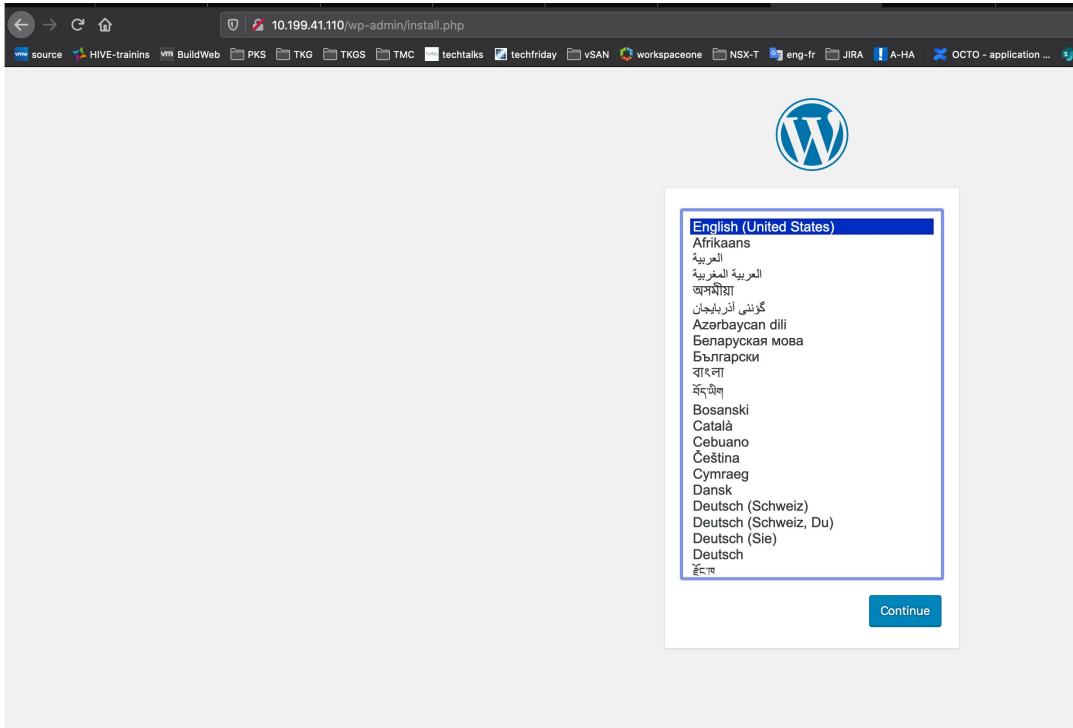
```
kubectl apply -k . -n wordpress-lbstaticip
secret/mysql-pass-c57bb4t7mf created
service/wordpress-mysql created
service/wordpress created
deployment.apps/wordpress-mysql created
deployment.apps/wordpress created
persistentvolumeclaim/mysql-pv-claim created
persistentvolumeclaim/wp-pv-claim created
```

Verify Wordpress app deployment:

```
kubectl get all -n wordpress-lbstaticip
```

```
kubectl get pvc,pv -n wordpress-lbstaticip
```

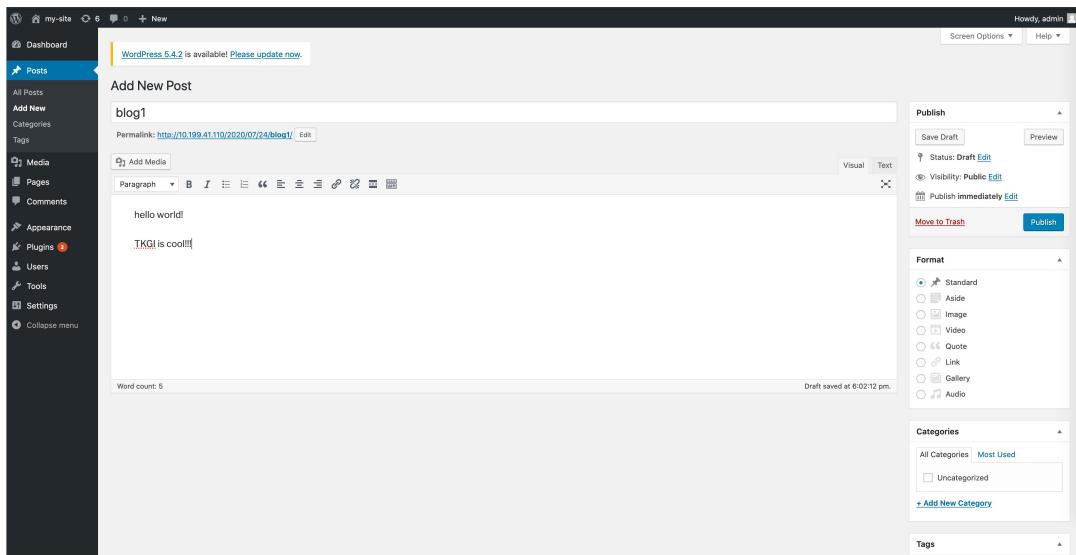
Access the Wordpress app at <http://10.199.41.110/>. Use the static IP you set for the load balancer.



Create a user and log in.

The screenshot shows the WordPress dashboard for the site 'my-site'. The left sidebar includes links for Home, Updates, Posts, Media, Pages, Comments, Appearance, Plugins, Users, Tools, and Settings. The main area displays a 'Welcome to WordPress!' message, 'Get Started' options like 'Customize Your Site' or 'Change your theme completely', and sections for 'At a Glance' (1 Post, 1 Comment), 'Activity' (Recent Published post 'Hello world!'), 'Recent Comments' (one comment from a WordPress Commenter), and 'Quick Draft' (a draft titled 'What's on your mind?'). The top right shows the user 'Howdy, admin' and 'Screen Options' and 'Help' buttons.

Create blog post.



Publish the blog.

## Backup the Wordpress App Using Namespace

Because the Wordpress app is stateful, add annotations for the stateful pods with the volume name.

From `wordpress-deployment.yaml`, the volume name is `wordpress-persistent-storage`.

From `mysql-deployment.yaml` the volume name is `mysql-persistent-storage`.

Get the pod names:

```
kubectl get pod -n wordpress-lbstaticip
```

Annotate the pods:

```
kubectl -n wordpress-lbstaticip annotate pod/wordpress-675699f695-f5zdl backup.velero.io/backup-volumes=wordpress-persistent-storage
pod/wordpress-675699f695-f5zdl annotated
```

```
kubectl -n wordpress-lbstaticip annotate pod/wordpress-mysql-69dcc4fc49-zpjrd backup.v
```

```
elero.io/backup-volumes=mysql-persistent-storage
pod/wordpress-mysql-69dcc4fc49-zpjrd annotated
```

Verify the annotations:

```
kubectl -n wordpress-lbstaticip describe pod wordpress-675699f695-f5zdl | grep Annotations
Annotations: backup.velero.io/backup-volumes: wordpress-persistent-storage

kubectl -n wordpress-lbstaticip describe pod wordpress-mysql-69dcc4fc49-zpjrd | grep Annotations
Annotations: backup.velero.io/backup-volumes: mysql-persistent-storage
```

Perform the Velero backup:

```
velero backup create wordpress-lbstaticip-backup --include-namespaces wordpress-lbstaticip

Backup request "wordpress-lbstaticip-backup" submitted successfully.
Run `velero backup describe wordpress-lbstaticip-backup` or `velero backup logs wordpress-lbstaticip-backup` for more details.
```

Verify the backup that was created.

| NAME                        | STATUS    | ERRORS           | WARNINGS | CREATED                       |
|-----------------------------|-----------|------------------|----------|-------------------------------|
|                             | EXPIRES   | STORAGE LOCATION | SELECTOR |                               |
| wordpress-lbstaticip-backup | Completed | 0                | 0        | 2020-07-24 11:35:46 -0700 PDT |
|                             | 29d       | default          | <none>   |                               |

Verify backup details:

```
velero backup describe wordpress-lbstaticip-backup --details
```

Use Velero CRD commands to further verify the backup:

```
kubectl get crd
```

```
kubectl get backups.velero.io -n velero
```

```
kubectl describe backups.velero.io wordpress-lbstaticip-backup -n velero
```

## Restore the Wordpress App

To test the restoration of the Wordpress app, delete it.

Delete the namespace:

```
kubectl delete ns wordpress-lbstaticip

namespace "wordpress-lbstaticip" deleted
```

Verify namespace deletion:

```
kubectl get ns
```

```
kubectl get pvc,pv --all-namespaces
```

Make sure the storage class used by the application is still present:

```
kubectl get sc
```

Restore the Wordpress app:

```
velero restore create --from-backup wordpress-lbstaticip-backup

Restore request "wordpress-lbstaticip-backup-20200724114046" submitted successfully.
Run `velero restore describe wordpress-lbstaticip-backup-20200724114046` or `velero re
store logs wordpress-lbstaticip-backup-20200724114046` for more details.
```

Verify that the Wordpress app is restored:

| velero restore get                         |        |          | BACKUP              | STATUS                      |           |
|--------------------------------------------|--------|----------|---------------------|-----------------------------|-----------|
| NAME                                       | ERRORS | WARNINGS | CREATED             | SELECTOR                    |           |
| wordpress-lbstaticip-backup-20200724114046 | 0      | 0        | 2020-07-24 11:40:46 | wordpress-lbstaticip-backup | Completed |

Verify restoration details:

```
velero restore describe wordpress-lbstaticip-backup-20200724114046

Name: wordpress-lbstaticip-backup-20200724114046
Namespace: velero
Labels: <none>
Annotations: <none>

Phase: Completed

Backup: wordpress-lbstaticip-backup

Namespaces:
 Included: all namespaces found in the backup
 Excluded: <none>

Resources:
 Included: *
 Excluded: nodes, events, events.events.k8s.io, backups.velero.io, restores.v
 elero.io, resticrepositories.velero.io
 Cluster-scoped: auto

Namespace mappings: <none>

Label selector: <none>

Restore PVs: auto

Restic Restores (specify --details for more information):
 Completed: 2
```

Verify the Wordpress namespace:

```
kubectl get ns
NAME STATUS AGE
default Active 16d
kube-node-lease Active 16d
kube-public Active 16d
kube-system Active 16d
pks-system Active 16d
velero Active 2d21h
wordpress-lbstaticip Active 68s
```

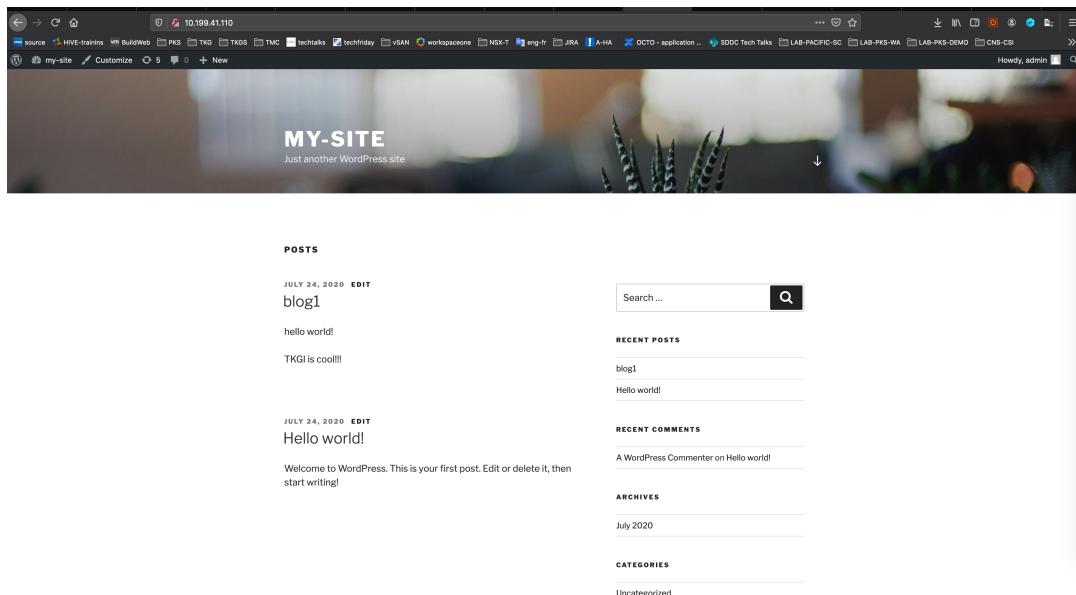
Check for all Wordpress objects in the namespace:

```
kubectl get all -n wordpress-lbstaticip
```

Verify persistent volume for Wordpress:

```
kubectl get pvc,pv -n wordpress-lbstaticip
```

Access the Wordpress blog at <http://10.199.41.110/>. Use the static IP you set for the load balancer.



## Conclusions

Key takeaways from the Velero backup and restore operation for this type of application:

- Pod annotation is still required for Velero backup of PV
- The namespace ‘wordpress-lbstaticip’ was automatically re-created
- The K8s SVC LB IP has been preserved as expected

## Backup and Restore Stateful App with Static IP for Ingress

This topic describes how to use Velero to backup and restore a stateful application with ingress and a static IP address.

## Overview

This topic describes how to use Velero to backup and restore a stateful application with ingress and a static IP address.

The application we are going to use to demonstrate this scenario is the Cafe stateless app. Kubernetes ingress provides a layer 7 load balancer. In this case the IP address must be static.

To demonstrate backing up and restoring a stateful application:

1. [Create a Network Profile](#)
2. [Deploy the Coffee-Tea App](#)
3. [Back Up the Coffee-Tea App Using Namespace](#)
4. [Restore the Coffee-Tea App](#)
5. [Review Conclusions](#)

## Prerequisites

Before starting your Velero demonstraion, you need to:

- Have a TKGI Kubernetes cluster with static IP set from a floating IP pool.
- Minio, Velero, and Restic have been installed. For more information, see [Installing Velero and Restic](#).
- Download the Coffee-Tea app YAML files to a local known directory:
  - `coffee-rc.yml`
  - `tea-rc.yml`
  - `coffee-svc.yml`
  - `tea-svc.yml`
  - `cafe-ingress-http.yml`
- If testing locally, ensure the following entry is present in the `/etc/hosts` of the computer accessing the Coffee-Tea app:

```
/etc/hosts
10.199.41.111 cafe.example.com
```

## Create a Network Profile

To create and apply a network profile for DNS lookup of the Kubernetes API server and the fixed IP address:

1. Create a network profile using the following template:

```
{
 "name": "dns-lookup-api-ingress",
 "description": "Network Profile for DNS Lookup - API and INGRESS",
```

```

"parameters": {
 "fip_pool_ids": [
 "970e09f1-6f28-4457-b069-5c40d145f4e3"
],
 "dns_lookup_mode": "API_INGRESS",
 "ingress_prefix": "INGRESS-SUBDOMAIN"
}
}

```

Where `INGRESS-SUBDOMAIN` is the ingress subdomain prefix.

Because DNS mode is set to `API_INGRESS`, TKGI creates the cluster with `ingress_prefix.hostname` as the Kubernetes control plane FQDN. TKGI confirms that the ingress subdomain can be resolved as a subdomain prefix on the host before creating new clusters.

2. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
3. Apply the network profile to your Kubernetes cluster using `tkgi update-cluster`. For more information, see [Assign a Network Profile to an Existing Cluster](#) in *Using Network Profiles*.

## Deploy the Coffee-Tea App

To deploy the example Coffee-Tea App:

1. To create the Namespace for the application:

```
kubectl create ns tea-coffee
```

For example:

```
kubectl create ns tea-coffee

namespace/tea-coffee created
```

2. To deploy the Tea-Coffee app:

```
kubectl apply -f . -n tea-coffee
```

For example:

```
kubectl apply -f . -n tea-coffee

ingress.extensions/cafe-ingress created

replicationcontroller/coffee-rc created

service/coffee-svc created
```

```
replicationcontroller/tea-rc created
service/tea-svc created
```

3. To verify the example app deployment:

```
kubectl get all -n tea-coffee
```

For example:

```
kubectl get all -n tea-coffee

NAME READY STATUS RESTARTS AGE
pod/coffee-rc-8lrwn 1/1 Running 0 7m19s
pod/coffee-rc-kn65r 1/1 Running 0 7m19s
pod/tea-rc-fhhnz 1/1 Running 0 7m19s
pod/tea-rc-t59cs 1/1 Running 0 7m19s

NAME DESIRED CURRENT READY AGE
replicationcontroller/coffee-rc 2 2 2 7m19s
replicationcontroller/tea-rc 2 2 2 7m19s

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S)
) AGE
service/coffee-svc ClusterIP 10.100.200.223 80/TCP 7m
19s
service/tea-svc ClusterIP 10.100.200.229 80/TCP 7m
19s
```

4. To review the sample app's ingress configuration:

```
kubectl get ingress -n tea-coffee
```

For example:

```
kubectl get ingress -n tea-coffee
```

| NAME | HOSTS | ADDRESS | PORTS | AGE |
|------|-------|---------|-------|-----|
|------|-------|---------|-------|-----|

```
cafe-ingress cafe.example.com 10.199.41.111 80 8s
```

5. To review the sample app's ingress configuration:

```
kubectl describe ingress cafe-ingress -n tea-coffee
```

For example:

```
kubectl describe ingress cafe-ingress -n tea-coffee

Name: cafe-ingress
Namespace: tea-coffee
Address: 10.199.41.111
Default backend: default-http-backend:80 ()
Rules:
Host Path Backends

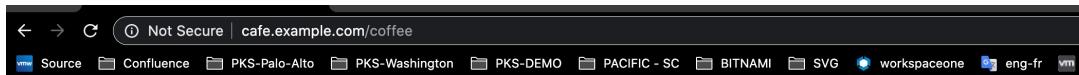
cafe.example.com
/tea tea-svc:80 (172.16.19.4:80,172.16.19.5:80)
/coffee coffee-svc:80 (172.16.19.2:80,172.16.19.3:80)

Annotations: kubectl.kubernetes.io/last-applied-configuration: {"apiVersion":"networking.k8s.io/v1","kind":"Ingress","metadata":{"annotations":{},"name":"cafe-ingress","namespace":"tea-coffee"},"spec":{"rules":[{"host":"cafe.example.com","http":{"paths":[{"backend":{"service":{"name":"tea-svc","port":{"number":80}}},"path":"/tea","pathType":"Prefix"}, {"backend":{"service":{"name":"coffee-svc","port":{"number":80}}},"path":"/coffee","pathType":"Prefix"}]}]}}
```

Events:

6. To access the Coffee-Tea app, connect to the Coffee-Tea app at <http://cafe.example.com/coffee> and <http://cafe.example.com/tea>.

For example:



**NGINX**

Server address: 172.16.19.3:80  
 Server name: coffee-rc-kn65r  
 Date: 27/Jul/2020:14:59:21 +0000  
 URI: /coffee

Auto Refresh

Request ID: 8fb9e31f9216ea885ed66fbe6493b593  
© NGINX, Inc. 2018



**NGINX**

Server address: 172.16.19.4:80  
 Server name: tea-rc-t59cs  
 Date: 27/Jul/2020:14:59:45 +0000  
 URI: /tea

Auto Refresh

Request ID: 358500ff72ec87bb1542118c10f2c55f  
© NGINX, Inc. 2018

## Back Up the Coffee-Tea App Using Namespace

To back up the Coffee-Tea App using the sample app's `tea-coffee-backup` namespace:

1. Use the Velero `backup` command:

```
velero backup create tea-coffee-backup --include-namespaces tea-coffee
```

For example:

```
velero backup create tea-coffee-backup -include-namespaces tea-coffee

Backup request "tea-coffee-backup" submitted successfully.

Run velero backup describe tea-coffee-backup or velero backup logs tea-coffee-backup for more details.
```

2. Verify the backup:

```
velero backup get
```

For example:

| NAME              | STATUS    | ERRORS  | WARNINGS | CREATED                          |
|-------------------|-----------|---------|----------|----------------------------------|
|                   | EXPIRES   | STORAGE | LOCATION | SELECTOR                         |
| tea-coffee-backup | Completed | 0       | 0        | 2020-07-27<br>09:16:02 -0700 PDT |
|                   | 29d       | default |          |                                  |

3. Verify the backup by reviewing backup details:

```
velero backup describe tea-coffee-backup
```

4. To verify the backup further:

1. Use the Velero CRD command:

```
kubectl get crd
```

2. Review the status of the backup:

```
kubectl get backups.velero.io -n velero
```

For example:

| NAME              | AGE |
|-------------------|-----|
| tea-coffee-backup | 97s |

3. Review the details of the backup:

```
kubectl describe backups.velero.io tea-coffee-backup -n velero
```

## Restore the Coffee-Tea App

To restore the Coffee-Tea app from the backup using Velero:

1. To clear the original Coffee-Tea app from your cluster:

1. Delete the Coffee-Tea app namespace:

```
kubectl delete ns tea-coffee
```

For example:

```
kubectl delete ns tea-coffee
```

```
namespace "tea-coffee" deleted
```

2. Verify that the Coffee-Tea app has been removed:

```
kubectl get ns
```

2. To restore the Coffee-Tea app from backup using Velero:

```
velero restore create --from-backup tea-coffee-backup
```

For example:

```
velero restore create --from-backup tea-coffee-backup
```

```
Restore request "tea-coffee-backup-20200727092014" submitted successfully.
```

```
Run velero restore describe tea-coffee-backup-20200727092014 or velero restore logs tea-coffee-backup-20200727092014 for more details.
```

3. To verify the Coffee-Tea app has been restored:

1. Review the Velero restoral history:

```
velero restore get
```

For example:

```
velero restore get
```

| NAME   | BACKUP   | STATUS  |          |
|--------|----------|---------|----------|
| ERRORS | WARNINGS | CREATED | SELECTOR |

|                                  |                   |                               |
|----------------------------------|-------------------|-------------------------------|
| tea-coffee-backup-20200727092014 | tea-coffee-backup | Completed                     |
| 0                                | 0                 | 2020-07-27 09:20:14 -0700 PDT |

2. To review the Velero restoration:

```
velero restore describe tea-coffee-backup-20200727092014
```

For example:

```
velero restore describe tea-coffee-backup-20200727092014
```

Name: tea-coffee-backup-20200727092014

Namespace: velero

Labels:

Annotations:

Phase: Completed

Backup: tea-coffee-backup

Namespaces:

Included: all namespaces found in the backup

Excluded:

Resources:

Included: \*

Excluded: nodes, events, events.events.k8s.io, backups.velero.io, restores.velero.io, resticrepositories.velero.io

Cluster-scoped: auto

Namespace mappings:

Label selector:

Restore PVs: auto

3. Confirm that the Coffee-Tea app's `tea-coffee` namespace has been restored:

```
kubectl get ns
```

For example:

| kubectl get ns  |        |       |
|-----------------|--------|-------|
| NAME            | STATUS | AGE   |
| default         | Active | 138m  |
| kube-node-lease | Active | 138m  |
| kube-public     | Active | 138m  |
| kube-system     | Active | 138m  |
| pks-system      | Active | 121m  |
| tea-coffee      | Active | 56s   |
| velero          | Active | 9m24s |

4. Verify that all app objects have been restored:

```
kubectl get all -n tea-coffee
```

For example:

| kubectl get all -n tea-coffee |       |         |          |     |  |
|-------------------------------|-------|---------|----------|-----|--|
| NAME                          | READY | STATUS  | RESTARTS | AGE |  |
| pod/coffee-rc-8lrwn           | 1/1   | Running | 0        | 89s |  |
| pod/coffee-rc-kn65r           | 1/1   | Running | 0        | 89s |  |
| pod/tea-rc-fhhnz              | 1/1   | Running | 0        | 89s |  |
| pod/tea-rc-t59cs              | 1/1   | Running | 0        | 89s |  |

| NAME                              | DESIRED | CURRENT | READY | AG |
|-----------------------------------|---------|---------|-------|----|
| replicationcontroller/coffee-rc-s | 2       | 2       | 2     | 89 |
| replicationcontroller/tea-rc      | 2       | 2       | 2     | 89 |

| S | NAME<br>PORT (S)          | TYPE<br>AGE | CLUSTER-IP     | EXTERNAL-IP |
|---|---------------------------|-------------|----------------|-------------|
|   | service/coffee-svc<br>89s | ClusterIP   | 10.100.200.197 | 80/TCP      |
|   | service/tea-svc<br>89s    | ClusterIP   | 10.100.200.17  | 80/TCP      |

5. Review the Coffee-Tea app ingress:

```
kubectl get ingress -n tea-coffee
```

For example:

| kubectl get ingress -n tea-coffee |                  |               |       |      |
|-----------------------------------|------------------|---------------|-------|------|
| NAME                              | HOSTS            | ADDRESS       | PORTS | AGE  |
| cafe-ingress                      | cafe.example.com | 10.199.41.111 | 80    | 112s |

6. Review Coffee-Tea app ingress details:

```
kubectl describe ingress cafe-ingress -n tea-coffee
```

For example:

```
kubectl describe ingress cafe-ingress -n tea-coffee
```

Name: cafe-ingress

Namespace: tea-coffee

Address: 10.199.41.111

Default backend: default-http-backend:80 ()

Rules:

| Host | Path | Backends |
|------|------|----------|
|------|------|----------|

---

cafe.example.com

```
/tea tea-svc:80 (172.16.19.2:80,172.16.19.3:80)

/coffee coffee-svc:80 (172.16.19.4:80,172.16.19.5:80)
```

Annotations:

```
kubectl.kubernetes.io/last-applied-configuration: {"apiVersion": "networking.k8s.io/v1", "kind": "Ingress", "metadata": {"annotations": {}, "name": "cafe-ingress", "namespace": "tea-coffee"}, "spec": {"rules": [{"host": "cafe.example.com", "http": {"paths": [{"backend": {"service": {"name": "tea-svc", "port": {"number": 80}}, "path": "/tea", "pathType": "Prefix"}, {"backend": {"service": {"name": "coffee-svc", "port": {"number": 80}}, "path": "/coffee", "pathType": "Prefix"}]}]}]}}
```

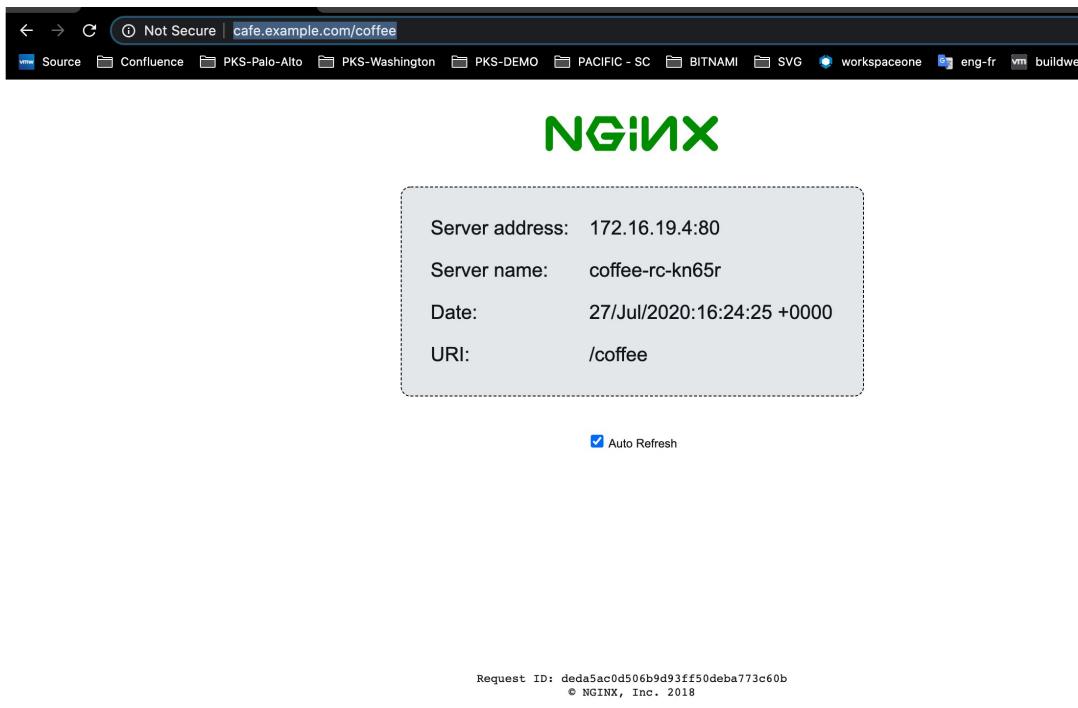
```
ncp/internal_ip_for_policy: 100.64.208.63
```

Events:

- To access the restored Coffee-Tea app, connect to the Coffee-Tea app at <http://cafe.example.com/coffee> and <http://cafe.example.com/tea>.

For example:





## Conclusions

Key takeaways from the Velero backup and restore operation for this type of application:

- The namespace ‘tea-coffee’ is automatically recreated by Velero
- The Kubernetes ingress IP is preserved (10.199.41.111)

## Backup and Restore Stateful App with Static IP for Load Balancer Service (CSI Edition)

This topic describes how to use Velero to backup and restore a stateful application with a load balancer service with a static IP address.

## Overview

This topic describes how to use Velero to backup and restore a Kubernetes stateful application with a service of type load balancer that uses a static IP address.

The application used to demonstrate Velero backup and restore with fixed IP is the Wordpress stateful app. By design Wordpress stores in its data structure the original IP address that was used during its initial launch. To successfully restore from backup a Wordpress app, the service of type load balancer must be deployed with a fixed IP address. This ensures the same IP will be used when a Velero restore is performed.

## Prerequisites

Install and configure Minio, Velero, and Restic.

Download the Wordpress app YAML files to a local known directory:

- mysql-deployment.yaml
- wordpress-deployment.yaml

## Configure Wordpress YAML Files

Edit the **wordpress-deployment.yaml** to include the static IP for the load balancer (`loadBalancerIP: IP`). In this example, the IP address used is taken from the NSX-T floating IP pool that is resourced for TKGI.

```
apiVersion: v1
kind: Service
metadata:
 name: wordpress
 labels:
 app: wordpress
spec:
 ports:
 - port: 80
 selector:
 app: wordpress
 tier: frontend
 type: LoadBalancer
 loadBalancerIP: 10.199.41.110
```

Create the `kustomization.yaml` file` with a value for YOUR\_PASSWORD:

```
secretGenerator:
- name: mysql-pass
 literals:
 - password=YOUR_PASSWORD
resources:
- mysql-deployment.yaml
- wordpress-deployment.yaml
```

Create the default storage class YAML `default-sc.yaml`:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
 name: default-sc
 annotations:
 storageclass.kubernetes.io/is-default-class: "true"
provisioner: kubernetes.io/vsphere-volume
parameters:
 diskformat: thin
```

Apply the storage class YAML file:

```
kubectl apply -f 0-default-sc.yaml
storageclass.storage.k8s.io/default-sc created
```

Verify the storage class:

```
kubectl get sc
```

| NAME                                                                                     | PROVISIONER                           | RECLAIMPOLICY | VOLUMEBINDINGMOD |
|------------------------------------------------------------------------------------------|---------------------------------------|---------------|------------------|
| E ALLOWVOLUMEEXPANSION AGE<br>default-sc (default) kubernetes.io/vsphere-volume<br>false | kubernetes.io/vsphere-volume<br>3m38s | Delete        | Immediate        |

## Deploy Wordpress App

Create Wordpress namespace:

```
kubectl create ns wordpress-lbstaticip
```

```
namespace/wordpress-lbstaticip created
```

Deploy the Wordpress app:

```
kubectl apply -k . -n wordpress-lbstaticip
```

```
secret/mysql-pass-c57bb4t7mf created
```

```
service/wordpress-mysql created
```

```
service/wordpress created
```

```
deployment.apps/wordpress-mysql created
```

```
deployment.apps/wordpress created
```

```
persistentvolumeclaim/mysql-pv-claim created
```

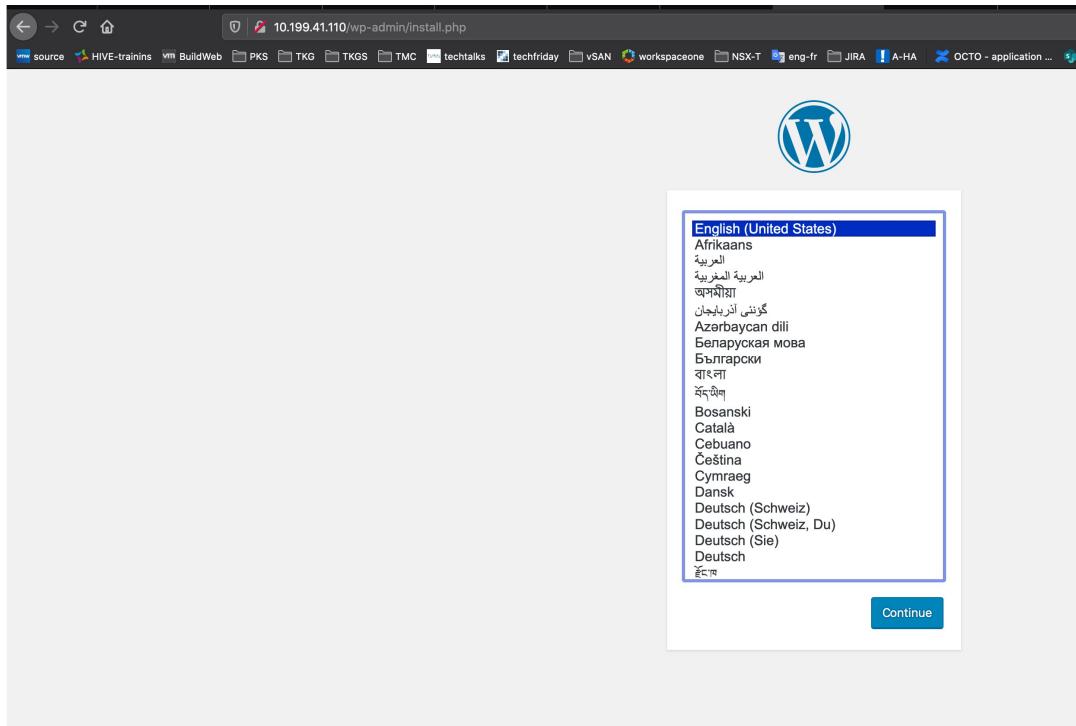
```
persistentvolumeclaim/wp-pv-claim created
```

Verify Wordpress app deployment:

```
kubectl get all -n wordpress-lbstaticip
```

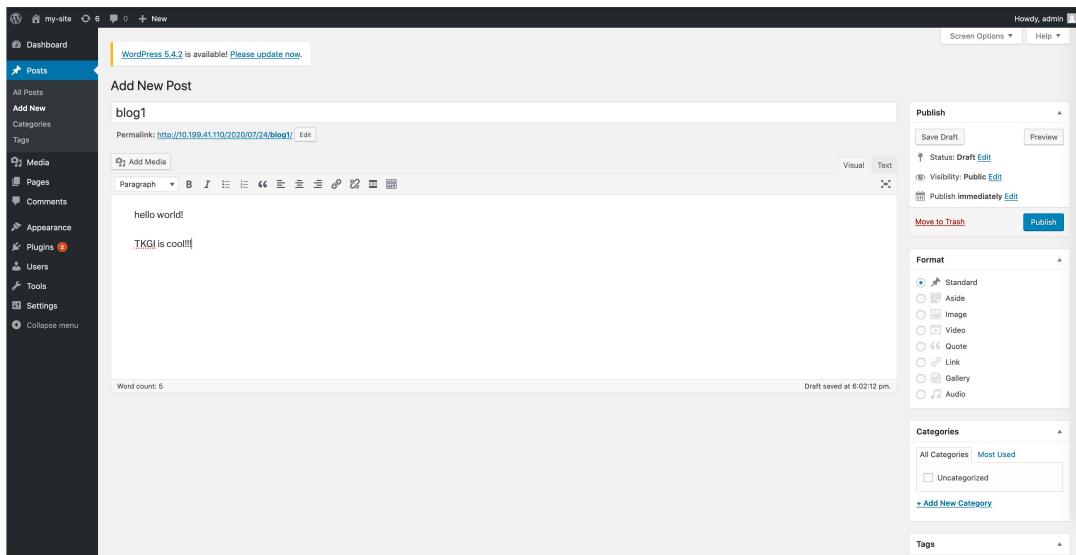
```
kubectl get pvc,pv -n wordpress-lbstaticip
```

Access the Wordpress app at <http://10.199.41.110/>. Use the static IP you set for the load balancer.



Create a user and log in.

Create blog post.



Publish the blog.

## Backup the Wordpress App Using Namespace

Because the Wordpress app is stateful, add annotations for the stateful pods with the volume name.

From `wordpress-deployment.yaml`, the volume name is `wordpress-persistent-storage`.

From `mysql-deployment.yaml` the volume name is `mysql-persistent-storage`.

Get the pod names:

```
kubectl get pod -n wordpress-lbstaticip
```

Annotate the pods:

```
kubectl -n wordpress-lbstaticip annotate pod/wordpress-675699f695-f5zdl backup.velero.io/backup-volumes=wordpress-persistent-storage
pod/wordpress-675699f695-f5zdl annotated
```

```
kubectl -n wordpress-lbstaticip annotate pod/wordpress-mysql-69dcc4fc49-zpjrd backup.v
```

```
elero.io/backup-volumes=mysql-persistent-storage
pod/wordpress-mysql-69dcc4fc49-zpjrd annotated
```

Verify the annotations:

```
kubectl -n wordpress-lbstaticip describe pod wordpress-675699f695-f5zdl | grep Annotations
Annotations: backup.velero.io/backup-volumes: wordpress-persistent-storage

kubectl -n wordpress-lbstaticip describe pod wordpress-mysql-69dcc4fc49-zpjrd | grep Annotations
Annotations: backup.velero.io/backup-volumes: mysql-persistent-storage
```

Perform the Velero backup:

```
velero backup create wordpress-lbstaticip-backup --include-namespaces wordpress-lbstaticip

Backup request "wordpress-lbstaticip-backup" submitted successfully.
Run `velero backup describe wordpress-lbstaticip-backup` or `velero backup logs wordpress-lbstaticip-backup` for more details.
```

Verify the backup that was created.

| velero backup get           |           |          |          |                               |  |
|-----------------------------|-----------|----------|----------|-------------------------------|--|
| NAME                        | STATUS    | ERRORS   | WARNINGS | CREATED                       |  |
| EXPIRES                     | STORAGE   | LOCATION | SELECTOR |                               |  |
| wordpress-lbstaticip-backup | Completed | 0        | 0        | 2020-07-24 11:35:46 -0700 PDT |  |
| 0 PDT                       | 29d       | default  | <none>   |                               |  |

Verify backup details:

```
velero backup describe wordpress-lbstaticip-backup --details
```

Use Velero CRD commands to further verify the backup:

```
kubectl get crd
```

```
kubectl get backups.velero.io -n velero
```

```
kubectl describe backups.velero.io wordpress-lbstaticip-backup -n velero
```

## Restore the Wordpress App

To test the restoration of the Wordpress app, delete it.

Delete the namespace:

```
kubectl delete ns wordpress-lbstaticip
namespace "wordpress-lbstaticip" deleted
```

Verify namespace deletion:

```
kubectl get ns
```

```
kubectl get pvc,pv --all-namespaces
```

Make sure the storage class used by the application is still present:

```
kubectl get sc
```

Restore the Wordpress app:

```
velero restore create --from-backup wordpress-lbstaticip-backup

Restore request "wordpress-lbstaticip-backup-20200724114046" submitted successfully.
Run `velero restore describe wordpress-lbstaticip-backup-20200724114046` or `velero re
store logs wordpress-lbstaticip-backup-20200724114046` for more details.
```

Verify that the Wordpress app is restored:

| NAME                                       | BACKUP   | STATUS                      |           |
|--------------------------------------------|----------|-----------------------------|-----------|
| ERRORS                                     | WARNINGS | CREATED                     | SELECTOR  |
| wordpress-lbstaticip-backup-20200724114046 |          | wordpress-lbstaticip-backup | Completed |
| 0                                          | 0        | 2020-07-24 11:40:46         |           |

Verify restoration details:

```
velero restore describe wordpress-lbstaticip-backup-20200724114046

Name: wordpress-lbstaticip-backup-20200724114046
Namespace: velero
Labels: <none>
Annotations: <none>

Phase: Completed

Backup: wordpress-lbstaticip-backup

Namespaces:
 Included: all namespaces found in the backup
 Excluded: <none>

Resources:
 Included: *
 Excluded: nodes, events, events.events.k8s.io, backups.velero.io, restores.velero.io, resticrepositories.velero.io
 Cluster-scoped: auto

Namespace mappings: <none>

Label selector: <none>

Restore PVs: auto

Restic Restores (specify --details for more information):
```

```
Completed: 2
```

Verify the Wordpress namespace:

```
kubectl get ns
NAME STATUS AGE
default Active 16d
kube-node-lease Active 16d
kube-public Active 16d
kube-system Active 16d
pks-system Active 16d
velero Active 2d21h
wordpress-lbstaticip Active 68s
```

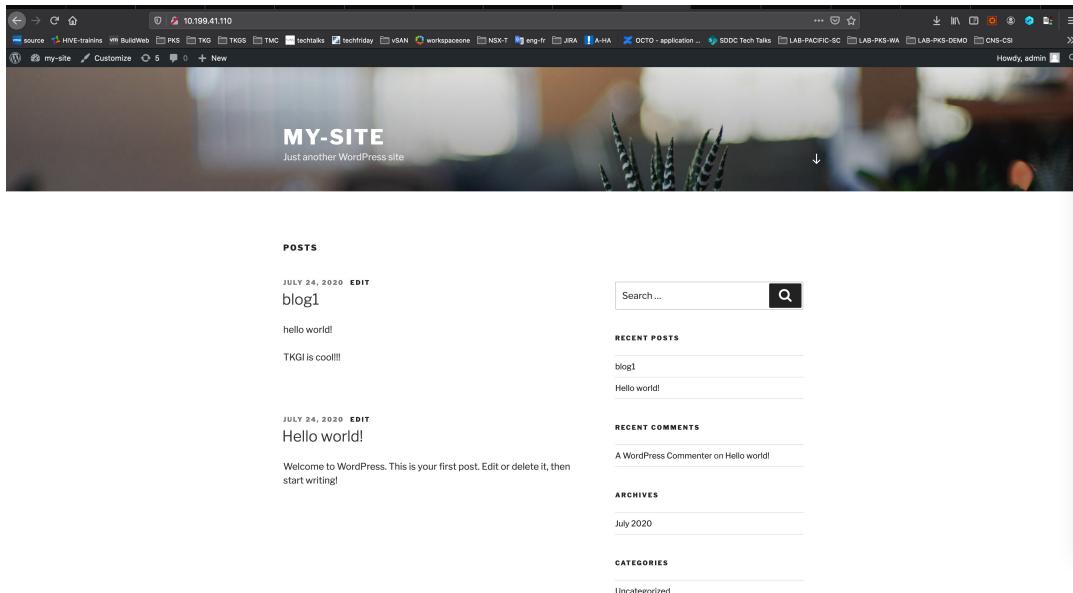
Check for all Wordpress objects in the namespace:

```
kubectl get all -n wordpress-lbstaticip
```

Verify persistent volume for Wordpress:

```
kubectl get pvc,pv -n wordpress-lbstaticip
```

Access the Wordpress blog at <http://10.199.41.110/>. Use the static IP you set for the load balancer.



## Conclusions

Key takeaways from the Velero backup and restore operation for this type of application:

- Pod annotation is still required for Velero backup of PV
- The namespace ‘wordpress-lbstaticip’ was automatically re-created
- The K8s SVC LB IP has been preserved as expected

## Backup and Restore All Cluster Workloads

This topic describes how to use Velero to backup and restore all workloads on a Kubernetes cluster.

## Overview

This example demonstrates how to use Velero to perform a full cluster backup and restore. This example uses the stateless Guestbook app with namespace and the statefulset CassandraDB app with namespace to perform cluster backup and restore.

## Prerequisites

Install and configure Minio, Velero, and Restic.

Download the [Guestbook app YAML files](#) to a local known directory:

- redis-leader-deployment.yaml
- redis-leader-service.yaml
- redis-follower-deployment.yaml
- redis-follower-service.yaml
- frontend-deployment.yaml
- frontend-service.yaml

Download the [CassandraDB YAML files](#) to a local known directory:

- cassandra-statefulset.yaml
- cassandra-storageclass.yaml
- headless-cassandra-service.yaml

## Deploy Stateless Guestbook App

Create Guestbook namespace:

```
kubectl create ns guestbook
```

Deploy the Guestbook app:

```
kubectl apply -f . -n guestbook
```

Verify:

```
kubectl get pod -n guestbook
```

```
kubectl get svc -n guestbook
```

Access the Guestbook app at <http://10.199.41.14/>.

---



## Guestbook

Messages

Submit

## Deploy StatefulSet CassandraDB App

Modify the storage class **cassandra-storageclass.yaml**:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
 name: demo-sts-sc
provisioner: kubernetes.io/vsphere-volume
parameters:
 diskformat: thin
```

Apply the storage class YAML file:

```
kubectl apply -f cassandra-storageclass.yaml
storageclass.storage.k8s.io/demo-sts-sc created
```

Verify the storage class:

```
kubectl get sc
NAME PROVISIONER RECLAIMPOLICY VOLUMEBINDINGMODE ALLOW
VOLUMEEXPANSION AGE
demo-sts-sc kubernetes.io/vsphere-volume Delete Immediate false
3s
```

Create the namespace:

```
kubectl create ns cassandra
```

Create the service:

```
kubectl apply -f headless-cassandra-service.yaml -n cassandra
```

Deploy CassandraDB app:

```
kubectl apply -f cassandra-statefulset.yaml -n cassandra
```

Verify pods and services:

```
kubectl get all -n cassandra

NAME READY STATUS RESTARTS AGE
pod/cassandra-0 1/1 Running 0 5m16s
pod/cassandra-1 1/1 Running 0 4m25s
pod/cassandra-2 1/1 Running 0 2m52s

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
service/cassandra ClusterIP None <none> <none> 5m21s

NAME READY AGE
statefulset.apps/cassandra 3/3 5m16s
```

Verify the persistent volume:

```
kubectl get pvc,pv -n cassandra
```

Make sure CassandraDB instances are fully participating in the cluster:

```
kubectl exec -it cassandra-0 -n cassandra -- nodetool status
```

## Create and Populate Cassandra Database

Create the DB:

```
kubectl exec -it cassandra-0 -n cassandra -- cqlsh
```

Create and populate table:

```
cqlsh> CREATE KEYSPACE demodb WITH REPLICATION = { 'class' : 'SimpleStrategy', 'replication_factor' : 3 };
cqlsh> use demodb;
cqlsh:demodb> CREATE TABLE emp(emp_id int PRIMARY KEY, emp_name text, emp_city text, emp_sal varint,emp_phone varint);
cqlsh:demodb> INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal) VALUES (100, 'Tom', 'Cork', 999, 1000000);
cqlsh:demodb> INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal) VALUES (101, 'Andrew', 'NY', 1000, 1000000);
cqlsh:demodb> INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal) VALUES (102, 'Lara', 'Paris', 1001, 1000000);
cqlsh:demodb> select * from emp;
```

Verify:

```
emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000
(3 rows)
```

Verify that the other Cassandra DB instances have the same information:

```
kubectl exec -it cassandra-1 -n cassandra -- cqlsh
```

```
kubectl exec -it cassandra-2 -n cassandra -- cqlsh
```

## Add Annotations

Add annotations for the CassandraDB statefulset pods with the volume name `cassandra-data`.

```
kubectl get pod -n cassandra

NAME READY STATUS RESTARTS AGE
cassandra-0 1/1 Running 0 19m
cassandra-1 1/1 Running 0 19m
cassandra-2 1/1 Running 0 17m
```

The pods `cassandra-0`, `cassandra-1` and `cassandra-2` must be annotated with `cassandra-data`.

```
kubectl -n cassandra annotate pod/cassandra-0 backup.velero.io/backup-volumes=cassandra-data
pod/cassandra-0 annotated

kubectl -n cassandra annotate pod/cassandra-1 backup.velero.io/backup-volumes=cassandra-data
pod/cassandra-1 annotated

kubectl -n cassandra annotate pod/cassandra-2 backup.velero.io/backup-volumes=cassandra-data
pod/cassandra-2 annotated
```

Verify the annotations:

```
kubectl -n cassandra describe pod/cassandra-0 | grep Annotations
Annotations: backup.velero.io/backup-volumes: cassandra-data

kubectl -n cassandra describe pod/cassandra-1 | grep Annotations
Annotations: backup.velero.io/backup-volumes: cassandra-data

kubectl -n cassandra describe pod/cassandra-2 | grep Annotations
Annotations: backup.velero.io/backup-volumes: cassandra-data
```

## Perform Velero Backup of the Cluster

Perform the Velero backup:

```
velero backup create k8s-cluster-backup

Backup request "k8s-cluster-backup" submitted successfully.
Run `velero backup describe k8s-cluster-backup` or `velero backup logs k8s-cluster-backup` for more details.
```

Verify the backup that was created.

| NAME               | STATUS    | ERRORS   | WARNINGS | CREATED                       | E |
|--------------------|-----------|----------|----------|-------------------------------|---|
| XPIRES             | STORAGE   | LOCATION | SELECTOR |                               |   |
| k8s-cluster-backup | Completed | 0        | 0        | 2020-07-29 13:56:37 -0700 PDT | 2 |
| 9d                 | default   |          | <none>   |                               |   |

Verify backup details:

```
velero backup describe k8s-cluster-backup --details

Name: k8s-cluster-backup
Namespace: velero
Labels: velero.io/storage-location=default
Annotations: velero.io/source-cluster-k8s-gitversion=v1.17.8+vmware.1
 velero.io/source-cluster-k8s-major-version=1
 velero.io/source-cluster-k8s-minor-version=17

Phase: Completed

Errors: 0
Warnings: 0

...
Velero-Native Snapshots: <none included>

Restic Backups:
 Completed:
 cassandra/cassandra-0: cassandra-data
 cassandra/cassandra-1: cassandra-data
 cassandra/cassandra-2: cassandra-data
```

Use Velero CRD commands to verify further:

```
kubectl get crd
```

```
kubectl get backups.velero.io -n velero

NAME AGE
k8s-cluster-backup 2m6s
```

```
kubectl describe backups.velero.io k8s-cluster-backup -n velero
```

## Restore All Cluster Workloads

Deploy a new Kubernetes cluster.

Get the credentials for the cluster.

[Install and configure Velero and Restic on the cluster.](#)

Verify that the Velero namesapce exists on the new cluster:

```
kubectl get ns

NAME STATUS AGE
default Active 2d8h
kube-node-lease Active 2d8h
kube-public Active 2d8h
kube-system Active 2d8h
pks-system Active 2d8h
velero Active 2d6h
```

Restore the all workload to the new cluster:

```
velero restore create --from-backup k8s-cluster-backup

Restore request "k8s-cluster-backup-20200729154634" submitted successfully.
Run `velero restore describe k8s-cluster-backup-20200729154634` or `velero restore log s k8s-cluster-backup-20200729154634` for more details.
```

Verify cluster restoration:

```
velero restore get

NAME BACKUP STATUS ERRORS WARNINGS
CREATED SELECTOR
k8s-cluster-backup-20200729154634 k8s-cluster-backup Completed 0 41
2020-07-29 15:46:34 -0700 PDT <none>
```

`velero restore describe k8s-cluster-backup-20200729154634`

```
Name: k8s-cluster-backup-20200729154634
Namespace: velero
Labels: <none>
Annotations: <none>

Phase: Completed

...
Restic Restores (specify --details for more information):
Completed: 3
```

Verify that the Cassandra and Guestbook namespaces exist:

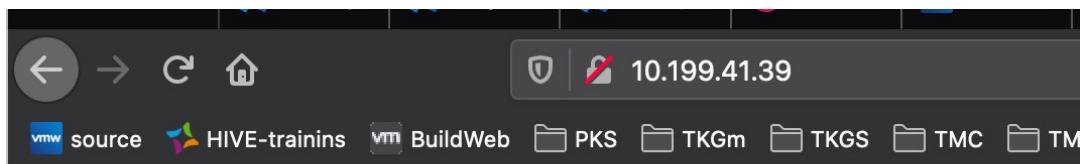
```
kubectl get ns

NAME STATUS AGE
cassandra Active 8m14s
default Active 2d8h
guestbook Active 8m11s
kube-node-lease Active 2d8h
kube-public Active 2d8h
kube-system Active 2d8h
pks-system Active 2d8h
velero Active 2d6h
```

Check the Guestbook app:

```
kubectl get all -n guestbook
```

Access the Guestbook app: <http://10.199.41.39/>. The message log will be empty because it is a stateless app.



## Guestbook

Messages

Submit

Verify that the CassandraDB storage class object exists. The storage class is automatically created during the restore operation because the storage class object is part of the default namespace.

```
kubectl get sc
```

| NAME            | PROVISIONER                  | RECLAIMPOLICY | VOLUMEBINDINGMODE | ALLOW |
|-----------------|------------------------------|---------------|-------------------|-------|
| VOLUMEEXPANSION | AGE                          |               |                   |       |
| demo-sts-sc     | kubernetes.io/vsphere-volume | Delete        | Immediate         | false |
|                 | 5m51s                        |               |                   |       |

Check the CassandraDB pods, service, and statefulset:

```
kubectl get all -n cassandra
```

| NAME            | READY | STATUS  | RESTARTS | AGE   |
|-----------------|-------|---------|----------|-------|
| pod/cassandra-0 | 1/1   | Running | 0        | 6m35s |
| pod/cassandra-1 | 1/1   | Running | 0        | 6m35s |
| pod/cassandra-2 | 1/1   | Running | 0        | 6m35s |

| NAME              | TYPE      | CLUSTER-IP | EXTERNAL-IP | PORT(S) | AGE  |
|-------------------|-----------|------------|-------------|---------|------|
| service/cassandra | ClusterIP | None       | <none>      | <none>  | 6m5s |

| NAME                       | READY | AGE  |
|----------------------------|-------|------|
| statefulset.apps/cassandra | 3/3   | 6m5s |

Check the CassandraDB persistence volume:

```
kubectl get pvc,pv -n cassandra
```

Verify the content of each Cassandra DB instance:

```
kubectl exec -it cassandra-0 -n cassandra -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

```
kubectl exec -it cassandra-1 -n cassandra -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

```
kubectl exec -it cassandra-2 -n cassandra -- cqlsh

Connected to Demo-Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.9 | CQL spec 3.4.2 | Native protocol v4]
Use HELP for help.
cqlsh> use demodb;
cqlsh:demodb> select * from emp;

emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+
 100 | Cork | Tom | 999 | 1000000
 102 | Paris | Lara | 1001 | 1000000
 101 | NY | Andrew | 1000 | 1000000

(3 rows)
```

## Conclusions

Note the following:

- Storage Class ‘demo-sts-sc’ used by CassandraDB app was automatically restored by Velero
- Pod annotation is still required for Velero backup of PV
- The namespace ‘cassandra’ was automatically re-created
- The namespace ‘guestbook’ was automatically re-created

## Backing Up and Restoring Kubernetes Clusters Provisioned by TKGI

This section describes how to back up and restore Kubernetes clusters provisioned by Tanzu Kubernetes Grid Integrated Edition.

### Overview

TKGI uses BOSH Backup and Restore (BBR) to back up and restore Kubernetes clusters provisioned by TKGI. For more information about BBR, see [BOSH Backup and Restore](#).

BBR orchestrates triggering the backup or restore process on the BOSH deployment, and transfers the backup artifacts to and from the BOSH deployment.

In context of Kubernetes clusters provisioned by TKGI, BBR backs up and restores the following components:

- TKGI cluster control plane nodes
- TKGI cluster worker nodes

BBR can also be used to backup and restore the TKGI Management Plane. See [Backup and Restore TKGI Components](#).

In context of TKGI, BBR does not back up and restore:

- Kubernetes workloads, see [Backing up and restoring Kubernetes workloads](#).
- Ops Manager VM (including the BOSH Director and TKGI tiles), see [Backing up and restoring Ops Manager](#).
- NSX-T objects and resources, such as load balancers, see [Backing up and restoring TKGI Infrastructure](#).
- Harbor VM

To use BBR to backup and restore Kubernetes clusters provisioned by TKGI, see the following topics:

- [Install and Configure BOSH Backup and Restore](#)
- [Backup Kubernetes Clusters Provisioned by Tanzu Kubernetes Grid Integrated Edition](#)
- [Restore Kubernetes Clusters Provisioned by](#)

## Testing Considerations

As part of your TKGI backup and restore planning and testing, consider the following test scenario.

- Take backup of a TKGI-provisioned Kubernetes cluster using BBR.

- Delete the cluster control plane and worker node VMs. Delete the disks and references.
- Restore cluster nodes using BBR. Delete old worker nodes using kubectl and restart kubelet.

The cluster nodes should be restored, and the cluster should be operational. If an application was deployed before cluster backup, the VIP of the load balancer for the application changes. To recover without application redeployment, create the service using a static IP. How to do this is described in the [workload backup and restore](#) documentation. If an application was deployed after cluster backup, the application is no longer available and the NSX-T objects created for the application are automatically deleted. In this case you will need to restore the application using Velero and the NSX-T objects using NSX Manager.

## Installing and Configuring BOSH Backup and Restore

This topic describes how to install BOSH Backup and Restore (BBR).

### Overview

To install BBR, first validate that your jumpbox VM is a valid BOSH backup host, then copy the `bbr` executable to the jumpbox and configure BBR.

For more information, see [Install and Configure BOSH Backup and Restore](#) below.

After installing BBR, you can run `bbr` commands to back up and restore your Tanzu Kubernetes Grid Integrated Edition deployment.

For more information about using BOSH Backup and Restore, see:

- To backup and restore Kubernetes clusters provisioned by TKGI, see [Backing Up and Restoring TKGI Clusters](#).
- To backup and restore TKGI Management Plane Components, see [Backing Up and Restoring TKGI Management Plane](#).

### Prerequisites

Using BBR requires the following:

- A jumpbox.

A jumpbox is a separate, hardened server on your network that provides a controlled means of accessing the other VMs on your network. See the [jumpbox-deployment GitHub](#) repository for an example jumpbox deployment.

You must have a jumpbox before you can install BBR to the jumpbox.

- The OpenBSD version of netcat must be installed on the jumpbox host.
- A `bbr` executable file. You must have the correct BBR executable version for your TKGI installation.
  - To determine the correct version of BBR for your deployment, see the [Tanzu Kubernetes Grid Integrated Edition Release Notes](#).

- To download a BBR installation file, see [BOSH Backup and Restore](#) on the VMware Tanzu Network.



**Note:** BBR does not support SSH gateways.

## Install and Configure BOSH Backup and Restore

To install and configure BBR:

1. [Configure Your Jumpbox for BBR](#)
2. [Install BBR on Your Jumpbox](#)
3. [Verify Your BBR Installation](#)
4. [Configure BBR Logging](#)

### Configure Your Jumpbox for BBR

Your jumpbox must meet or exceed minimum BBR requirements. You can use the Ops Manager VM as your jumpbox if it can be configured to meet all of the requirements below.

To configure your jumpbox to meet BBR requirements:

1. Size the jumpbox to have sufficient storage space for your backups.
2. Ensure the jumpbox can communicate with the network containing your Tanzu Kubernetes Grid Integrated Edition deployment.

BBR uses SSH to orchestrate the backup of your Tanzu Kubernetes Grid Integrated Edition instances using port 22 by default.

3. Configure the jumpbox to be in the same network as the deployed VMs.

BBR connects to the deployed VMs at their private IP addresses.

4. Ensure there is minimal network latency between the jumpbox and the source VMs BBR backs up.

### Install BBR on Your Jumpbox

To install the `bbr` executable to your jumpbox:

1. Download the latest compatible [BOSH Backup and Restore release](#) from the VMware Tanzu Network.
2. To add executable permissions to the `bbr` binary file, run the following command:

```
chmod a+x bbr
```

3. To securely copy the `bbr` binary file to your jumpbox, run the following command:

```
scp LOCAL-PATH-TO-BBR/bbr JUMPBOX-USER@JUMPBOX-ADDRESS:
```

Where:

- ◊ `LOCAL-PATH-TO-BBR` is the path to the `bbr` binary you downloaded from VMware Tanzu Network.
- ◊ `JUMPBOX-USER` is the ssh username for connecting to the jumpbox.
- ◊ `JUMPBOX-ADDRESS` is the IP address, or hostname, of the jumpbox.

## Verify Your BBR Installation

To verify that BBR is installed:

1. Run the following command:

```
bbr version
```

Verify the returned BBR version.

## Configure BBR Logging

BBR writes backup and restore logs to the current directory in a file named `bbr-TIMESTAMP.err.log`.

By default BBR writes errors associated with stack traces to the log file.

BBR also reports default information about the backup and restore run:

- The backup and restore scripts that it finds.
- When it starts or finishes a stage, such as `pre-backup scripts` or `backup scripts`.
- When the process is complete.
- When any error occurs.

To troubleshoot a failed BBR run, enable verbose logging. When executed in verbose mode, BBR reports the following additional information:

- Logs about the API requests made to the BOSH server.
- All commands executed on remote instances.
- All commands executed on local environment.
- Standard in and standard out streams for the backup and restore scripts when they are executed.

To enable verbose logging, use the optional `--debug` flag.

## Backing Up Tanzu Kubernetes Grid Integrated Edition

This topic describes how to use BOSH Backup and Restore (BBR) to back up Kubernetes clusters provisioned by VMware Tanzu Kubernetes Grid Integrated Edition.

### Overview

You can use BOSH Backup and Restore (BBR) to backup Kubernetes clusters provisioned by TKGI, including the control plane nodes, etcd database, and worker node VMs.

Kubernetes clusters provisioned by TKGI include custom backup and restore scripts which

encapsulate the correct procedure for backing up and restoring the cluster nodes and etcd database.

BBR orchestrates running the backup and restore scripts and transferring the generated backup artifacts to and from a backup directory. If configured correctly, BBR can use TLS to communicate securely with backup targets.

To view the BBR release notes, see the Cloud Foundry documentation, [BOSH Backup and Restore Release Notes](#).

## Recommendations

VMware recommends:

- Follow the full procedure documented in this topic when creating a backup. This ensures that you always have a consistent backup of Ops Manager and Tanzu Kubernetes Grid Integrated Edition to restore from.
- Back up your Kubernetes clusters frequently, especially before upgrading your Tanzu Kubernetes Grid Integrated Edition deployment.
- For BOSH v270.0 and above (currently in Ops Manager 2.7), prune the BOSH blobstore by running `bosh clean-up --all` prior to running a backup of the BOSH director. This removes all unused resources, including packages compiled against older stemcell versions, which can result in a smaller, faster backup of the BOSH Director. For more information see the [clean-up](#) command.



**Note:** The command `bosh clean-up -all` is a destructive operation and can remove resources that are unused but needed. For example, if an On-Demand Service Broker such as Tanzu Kubernetes Grid Integrated Edition is deployed **and** no service instances have been created, the releases needed to create a service instance will be categorized as unused and removed.

## Prepare to Back Up

Before you use BBR to either back up TKGI or restore TKGI from backup, follow these steps to retrieve deployment information and credentials:

- [Verify your BBR Version](#)
- [Retrieve the BBR SSH Credentials](#)
- [Retrieve the BOSH Director Credentials](#)
- [Retrieve the UAA Client Credentials](#)
- [Retrieve the BOSH Director Address](#)
- [Download the Root CA Certificate](#)
- [Retrieve the BOSH Command Line Credentials](#)
- [Retrieve Your Cluster Deployment Names](#)

## Verify Your BBR Version

Before running BBR, verify that the installed version of BBR is compatible with the version of Ops Manager your TKGI tile is on:

1. To determine the Ops Manager BBR version requirements, see the [Ops Manager Release Notes](#) for the version of Ops Manager you are using.
2. To verify the currently installed BBR version, run the following command:

```
bbr version
```

3. If the installed BBR version does not meet the Ops Manager BBR version requirement, or BBR is not installed, you must upgrade BBR. For more information, see [Installing BOSH Backup and Restore](#).

## Retrieve the BBR SSH Credentials

There are two ways to retrieve BOSH Director credentials:

- [Ops Manager Installation Dashboard](#)
- [Ops Manager API](#)

### Ops Manager Installation Dashboard

To retrieve your BBR SSH Credentials using the Ops Manager Installation Dashboard:

1. Navigate to the Ops Manager Installation Dashboard.
2. Click the BOSH Director tile.
3. Click the **Credentials** tab.
4. Locate **Bbr Ssh Credentials**.
5. Click **Link to Credentials** next to it.
6. Copy the `private_key_pem` field value.

### Ops Manager API

To retrieve your BBR SSH Credentials using the Ops Manager API:

1. Obtain your UAA access token. For more information, see [Access the Ops Manager API](#).
2. Retrieve the **Bbr Ssh Credentials** by running the following command:

```
curl "https://OPS-MAN-FQDN/api/v0/deployed/director/credentials/bbr_ssh_credentials" \
-X GET \
-H "Authorization: Bearer UAA-ACCESS-TOKEN"
```

Where:

- `OPS-MAN-FQDN` is the fully-qualified domain name (FQDN) for your Ops Manager deployment.
- `UAA-ACCESS-TOKEN` is your UAA access token.

3. Copy the value of the `private_key_pem` field.

## Save the BBR SSH Credentials to File

To save the BBR SSH credentials to a private key file:

1. To reformat the copied `private_key_pem` value and save it to a file in the current directory:

```
printf -- "YOUR-PRIVATE-KEY" > PRIVATE-KEY-FILE
```

Where:

- ◊ `YOUR-PRIVATE-KEY` is the text of your private key.
- ◊ `PRIVATE-KEY-FILE` is the path to the private key file you are creating.

For example:

```
$ printf - "--begin rsa private key-- fake key contents --end rsa private key--" > bbr_key.pem
```

## Retrieve the BOSH Director Credentials

There are two ways to retrieve BOSH Director credentials:

- [Ops Manager Installation Dashboard](#)
- [Ops Manager API](#)

### Ops Manager Installation Dashboard

To retrieve your BOSH Director credentials using the Ops Manager Installation Dashboard, perform the following steps:

1. Navigate to the Ops Manager Installation Dashboard.
2. Click the BOSH Director tile.
3. Click the **Credentials** tab.
4. Locate **Director Credentials**.
5. Click **Link to Credentials** next to it.
6. Copy and record the value of the `password` field.

### Ops Manager API

To retrieve your BOSH Director credentials using the Ops Manager API, perform the following steps:

1. Obtain your UAA access token. For more information, see [Access the Ops Manager API](#).
2. Retrieve the **Director Credentials** by running the following command:

```
curl "https://OPS-MAN-FQDN/api/v0/deployed/director/credentials/bbr_ssh_credentials" \
-X GET \
```

```
-H "Authorization: Bearer UAA-ACCESS-TOKEN"
```

Where:

- ◊ **OPS-MAN-FQDN** is the fully-qualified domain name (FQDN) for your Ops Manager deployment.
- ◊ **UAA-ACCESS-TOKEN** is your UAA access token.

3. Copy and record the value of the **password** field.

## Retrieve the UAA Client Credentials

To obtain BOSH credentials for your BBR operations, perform the following steps:

1. From the Ops Manager Installation Dashboard, click the **Tanzu Kubernetes Grid Integrated Edition** tile.
2. Select the **Credentials** tab.
3. Navigate to **Credentials > UAA Client Credentials**.
4. Record the value for **uaa\_client\_secret**.
5. Record the value for **uaa\_client\_name**.



**Note:** You must use BOSH credentials that limit the scope of BBR activity to your cluster deployments.

## Retrieve the BOSH Director Address

You access the BOSH Director using an IP address.

To obtain your BOSH Director's IP address:

1. Open the Ops Manager Installation Dashboard.
2. Select **BOSH Director > Status**.
3. Select the listed Director IP Address.

### Log In To BOSH Director

1. If you are not using the Ops Manager VM as your jumpbox, install the latest **BOSH CLI** on your jumpbox.
2. To log in to BOSH Director, using the IP address that you recorded above, run the following command line:

```
bosh -e BOSH-DIRECTOR-IP \
--ca-cert PATH-TO-BOSH-SERVER-CERTIFICATE log-in
```

Where:

- ◊ **BOSH-DIRECTOR-IP** is the BOSH Director IP address recorded above.
- ◊ **PATH-TO-BOSH-SERVER-CERTIFICATE** is the path to the root Certificate Authority (CA) certificate as outlined in [Download the Root CA Certificate](#).

3. To specify **Email**, specify `director`.
4. To specify **Password**, enter the **Director Credentials** that you obtained in [Retrieve the BOSH Director Credentials](#).

For example:

```
$ bosh -e 10.0.0.3
-ca-cert /var/tempest/workspaces/default/root_ca_certificate log-in Email (): d
irector Password (): **** Successfully authenticated with UAA Su
cceeded
```

## Download the Root CA Certificate

To download the root CA certificate for your Tanzu Kubernetes Grid Integrated Edition deployment, perform the following steps:

1. Open the Ops Manager Installation Dashboard.
2. In the top right corner, click your username.
3. Navigate to **Settings > Advanced**.
4. Click **Download Root CA Cert**.

## Retrieve the BOSH Command Line Credentials

1. Open the Ops Manager Installation Dashboard.
2. Click the **BOSH Director** tile.
3. In the BOSH Director tile, click the **Credentials** tab.
4. Navigate to **Bosh Commandline Credentials**.
5. Click **Link to Credential**.
6. Copy the credential value.

## Retrieve Your Cluster Deployment Names

To locate and record a cluster deployment name, follow the steps below for each cluster:

1. On the command line, run the following command to log in:

```
tkgi login -a TKGI-API -u USERNAME -k
```

Where:

- ◊ `TKGI-API` is the domain name for the TKGI API that you entered in **Ops Manager > Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN)**. For example, `api.tkgi.example.com`.
- ◊ `USERNAME` is your user name.

See [Logging in to Tanzu Kubernetes Grid Integrated Edition](#) for more information

about the `tkgi login` command.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

2. Identify the cluster ID:

```
tkgi cluster CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of your cluster.

3. From the output of this command, record the **UUID** value.
4. Open the Ops Manager Installation Dashboard.
5. Click the **BOSH Director** tile.
6. Select the **Credentials** tab.
7. Navigate to **Bosh Commandline Credentials** and click **Link to Credential**.
8. Copy the credential value.
9. SSH into your jumpbox. For more information about the jumpbox, see [Installing BOSH Backup and Restore](#).
10. To retrieve your cluster deployment name, run the following command:

```
BOSH-CLI-CREDENTIALS deployments | grep UUID
```

Where:

- ◊ `BOSH-CLI-CREDENTIALS` is the full value that you copied from the BOSH Director tile in [Retrieve the BOSH Command Line Credentials](#).
- ◊ `UUID` is the cluster UUID that you recorded in the previous step.

## Back Up Tanzu Kubernetes Grid Integrated Edition

To back up your Tanzu Kubernetes Grid Integrated Edition environment you must first connect to your jumpbox before executing `bbr` backup commands.

### Connect to Your Jumpbox

You can establish a connection to your jumpbox in one of the following ways:

- [Connect with SSH](#)
- [Connect with BOSH\\_ALL\\_PROXY](#)

For general information about the jumpbox, see [Installing BOSH Backup and Restore](#).

## Connect with SSH

To connect to your jumpbox with SSH, do one of the following:

- **If you are using the Ops Manager VM as your jumpbox, log in to the Ops Manager VM.**  
See [Log in to the Ops Manager VM with SSH](#) in *Advanced Troubleshooting with the BOSH CLI*.
- **If you want to connect to your jumpbox using the command line, run the following command:**

```
ssh -i PATH-TO-KEY JUMPBOX-USERNAME@JUMPBOX-ADDRESS
```

Where:

- ◊ **PATH-TO-KEY** is the local path to your private key for the jumpbox host.
- ◊ **JUMPBOX-USERNAME** is your jumpbox username.
- ◊ **JUMPBOX-ADDRESS** is the address of the jumpbox.



**Note:** If you connect to your jumpbox with SSH, you must run the BBR commands in the following sections from within your jumpbox.

## Connect with BOSH\_ALL\_PROXY

You can use the `BOSH_ALL_PROXY` environment variable to open an SSH tunnel with SOCKS5 to your jumpbox. This tunnel enables you to forward requests from your local machine to the BOSH Director through the jumpbox. When `BOSH_ALL_PROXY` is set, BBR always uses its value to forward requests to the BOSH Director.



**Note:** For the following procedures to work, ensure the SOCKS port is not already in use by a different tunnel or process.

To connect with `BOSH_ALL_PROXY`, do one of the following:

- **If you want to establish the tunnel separate from the BOSH CLI, do the following:**
  1. Establish the tunnel and make it available on a local port by running the following command:

```
ssh -4 -D SOCKS-PORT -fNC JUMPBOX-USERNAME@JUMPBOX-ADDRESS -i JUMPBOX-KEY
-FILE -o ServerAliveInterval=60
```

Where:

- **SOCKS-PORT** is the local SOCKS port.
- **JUMPBOX-USERNAME** is your jumpbox username.
- **JUMPBOX-ADDRESS** is the address of the jumpbox.

- `JUMPBOX-KEY-FILE` is the local SSH private key for accessing the jumpbox.

For example:

```
$ ssh -4 -D 12345 -fNC jumpbox@203.0.113.0 -i jumpbox.key -o ServerAliveInterval=60
```

2. Provide the BOSH CLI with access to the tunnel through `BOSH_ALL_PROXY` by running the following command:

```
export BOSH_ALL_PROXY=socks5://localhost:SOCKS-PORT
```

Where `SOCKS-PORT` is your local SOCKS port.

- If you want to establish the tunnel using the BOSH CLI, do the following:

1. Provide the BOSH CLI with the necessary SSH credentials to create the tunnel by running the following command:

```
export BOSH_ALL_PROXY=ssh+socks5://JUMPBOX-USERNAME@JUMPBOX-ADDRESS:SOCKS-PORT?private_key=JUMPBOX-KEY-FILE
```

Where:

- `JUMPBOX-USERNAME` is your jumpbox username.
- `JUMPBOX-ADDRESS` is the address of the jumpbox.
- `SOCKS-PORT` is your local SOCKS port.
- `JUMPBOX-KEY-FILE` is the local SSH private key for accessing the jumpbox.

For example:

```
$ export BOSH_ALL_PROXY=ssh+socks5://jumpbox@203.0.113.0:12345?private_key=jumpbox.key
```



**Note:** Using `BOSH_ALL_PROXY` can result in longer backup and restore times because of network performance degradation. All operations must pass through the proxy which means moving backup artifacts can be significantly slower.



**Warning:** In BBR v1.5.0 and earlier, the tunnel created by the BOSH CLI does not include the `ServerAliveInterval` flag. This might result in your SSH connection timing out when transferring large artifacts. In BBR v1.5.1, the `ServerAliveInterval` flag is included. For more information, see [bosh-backup-and-restore v1.5.1](#) on GitHub.

## Back Up Kubernetes Clusters Provisioned by TKGI

Before backing up your TKGI cluster deployments you should verify that they can be backed up.

### Verify Your Provisioned Clusters

To verify that you can reach your TKGI cluster deployments and that the deployments can be backed up, follow the steps below.

1. SSH into your jumpbox. For more information about the jumpbox, see [Configure Your Jumpbox](#) in *Installing BOSH Backup and Restore*.
2. To perform the BBR pre-backup check, run the following command from your jumpbox:

```
BOSH_CLIENT_SECRET=TKGI-UAA-CLIENT-SECRET bbr deployment \
--all-deployments --target BOSH-TARGET --username TKGI-UAA-CLIENT-NAME \
--ca-cert PATH-TO-BOSH-SERVER-CERT \
pre-backup-check
```

Where:

- ◊ `TKGI-UAA-CLIENT-SECRET` is the value you recorded for `uaa_client_secret` in [Download the UAA Client Credentials](#) above.
- ◊ `BOSH-TARGET` is the value you recorded for the BOSH Director's address in [Retrieve the BOSH Director Address](#) above. You must be able to reach the target address from the machine where you run `bbr` commands.
- ◊ `TKGI-UAA-CLIENT-NAME` is the value you recorded for `uaa_client_name` in [Download the UAA Client Credentials](#) above.
- ◊ `PATH-TO-BOSH-SERVER-CERT` is the path to the root CA certificate that you downloaded in [Download or Locate Root CA Certificate](#) above.

For example:

```
$ BOSH_CLIENT_SECRET=p455w0rd bbr deployment
-all-deployments -target bosh.example.com -username pivotal-container-
service-12345abcdefghijklmn
-ca-cert /var/tempest/workspaces/default/root_ca_certificate
pre-backup-check
```

3. If the pre-backup-check command is successful, the command returns a list of cluster deployments that can be backed up.

For example:

```
[21:51:23] Pending: service-instance_abcd-1234-5678-hijk-9010111213
1415 [21:51:23] ----- [21:51:31] Deployment 'service-instance_abcd-
eg-1234-5678-hijk-90101112131415' can be backed up. [21:51:31] -----
-- [21:51:31] Successfully can be backed up: service-instance_abcd-1-
234-5678-hijk-90101112131415
```

In the output above, `service-instance_abcd-1234-5678-hijk-90101112131415` is the BOSH deployment name of a TKGI cluster.

4. If the pre-backup-check command fails, do one or more of the following:

- ◊ Make sure you are using the correct Tanzu Kubernetes Grid Integrated Edition credentials.
- ◊ Run the command again, adding the `--debug` flag to enable debug logs. For more information, see [BBR Logging](#).
- ◊ Make the changes suggested in the output and run the pre-backup check again. For example, the deployments might not have the correct backup scripts, or the connection to the BOSH Director failed.

## Back Up Kubernetes Clusters Provisioned by TKGI

When backing up your TKGI cluster, you can choose to back up only one cluster or to backup all cluster deployments in scope. The procedures to do this are the following:

- [Back up All Cluster Deployments](#)
- [Back Up One Cluster Deployment](#)

### Back Up All Kubernetes Clusters

The following procedure backs up all cluster deployments.

Make sure you use the TKGI UAA credentials that you recorded in [Download the UAA Client Credentials](#). These credentials limit the scope of the backup to cluster deployments only.



**Note:** The BBR backup command can take a long time to complete. You can run it independently of the SSH session so that the process can continue running even if your connection to the jumpbox fails. The command above uses `nohup`, but you could also run the command in a `screen` or `tmux` session.

1. To back up all cluster deployments, run the following command from your jumpbox:

```
BOSH_CLIENT_SECRET=TKGI-UAA-CLIENT-SECRET nohup bbr deployment \
--all-deployments --target BOSH-TARGET --username TKGI-UAA-CLIENT-NAME \
--ca-cert PATH-TO-BOSH-SERVER-CERT \
backup [--with-manifest] [--artifact-path]
```

Where:

- ◊ `TKGI-UAA-CLIENT-SECRET` is the value you recorded for `uaa_client_secret` in [Download the UAA Client Credentials](#) above.
- ◊ `BOSH-TARGET` is the value you recorded for the BOSH Director's address in [Retrieve the BOSH Director Address](#) above. You must be able to reach the target address from the machine where you run `bbr` commands.
- ◊ `TKGI-UAA-CLIENT-NAME` is the value you recorded for `uaa_client_name` in [Download the UAA Client Credentials](#) above.
- ◊ `PATH-TO-BOSH-SERVER-CERT` is the path to the root CA certificate that you downloaded in [Download the Root CA Certificate](#) above.
- ◊ `--with-manifest` is an optional `backup` parameter to include the manifest in the

backup artifact. If you use this flag, the backup artifact then contains credentials that you should keep secret.

- `--artifact-path` is an optional `backup` parameter to specify the output path for the backup artifact.

For example:

```
$ BOSH_CLIENT_SECRET=p455w0rd
nohup bbr deployment
-all-deployments
-target bosh.example.com
-username pivotal-container-service-12345abcdefghijklmn
-ca-cert /var/tempest/workspaces/default/root_ca_certificate
backup
```



**Note:** The optional `-with-manifest` flag directs BBR to create a backup containing credentials. You should manage the generated backup artifact knowing it contains secrets for administering your environment.

2. If the `backup` command completes successfully, follow the steps in [Manage Your Backup Artifact](#) below.
3. If the backup command fails, the backup operation exits. BBR does not attempt to continue backing up any non-backed up clusters. To troubleshoot a failing backup, do one or more of the following:
  - Run the command again, adding the `--debug` flag to enable debug logs. For more information, see [BBR Logging](#).
  - Follow the steps in [Recover from a Failing Command](#) below.

#### Back Up a Single Kubernetes Cluster

1. To backup a single, specific cluster deployment, run the following command from your jumpbox:

```
BOSH_CLIENT_SECRET=TKGI-UAA-CLIENT-SECRET \
nohup bbr deployment \
--deployment CLUSTER-DEPLOYMENT-NAME \
--target BOSH-DIRECTOR-IP \
--username TKGI-UAA-CLIENT-NAME \
--ca-cert PATH-TO-BOSH-SERVER-CERT \
backup [--with-manifest] [--artifact-path]
```

Where:

- ◊ `TKGI-UAA-CLIENT-SECRET` is the value you recorded for `uaa_client_secret` in [Download the UAA Client Credentials](#) above.
- ◊ `CLUSTER-DEPLOYMENT-NAME` is the value you recorded in [Retrieve your Cluster Deployment Name](#) above.
- ◊ `BOSH-TARGET` is the value you recorded for the BOSH Director's address in [Retrieve the BOSH Director Address](#) above. You must be able to reach the target address from the machine where you run `bbr` commands.
- ◊ `TKGI-UAA-CLIENT-NAME` is the value you recorded for `uaa_client_name` in [Download the UAA Client Credentials](#) above.
- ◊ `PATH-TO-BOSH-SERVER-CERT` is the path to the root CA certificate that you downloaded in [Download the Root CA Certificate](#) above.
- ◊ `--with-manifest` is an optional `backup` parameter to include the manifest in the backup artifact. If you use this flag, the backup artifact then contains credentials that you should keep secret.
- ◊ `--artifact-path` is an optional `backup` parameter to specify the output path for the backup artifact.

For example:

```
$ BOSH_CLIENT_SECRET=p455w0rd nohup bbr deployment
-deployment service-instance_abcdefg-1234-5678-hijk-9010111213141
-target bosh.example.com -username pivotal-container-service-12345abc
defghijklmn
-ca-cert /var/tempest/workspaces/default/root_ca_certificate
backup
```



**Note:** The optional `-with-manifest` flag directs BBR to create a backup containing credentials. You should manage the generated backup artifact knowing it contains secrets for administering your environment.

2. If the `backup` command completes successfully, follow the steps in [Manage Your Backup Artifact](#) below.
3. If the backup command fails, do one or more of the following:
  - ◊ Run the command again, adding the `--debug` flag to enable debug logs. For more information, see [BBR Logging](#).
  - ◊ Follow the steps in [Recover from a Failing Command](#) below.

## Cancel a Cluster Backup

Backups can take a long time. If you realize that the backup is going to fail or that your developers need to push an app immediately, you might need to cancel the backup.

To cancel a backup, perform the following steps:

1. Terminate the BBR process by pressing Ctrl-C and typing `yes` to confirm.
2. Because stopping a backup can leave the system in an unusable state and prevent additional backups, follow the procedures in [Clean up After a Failed Backup](#) below.

## After Backing Up Tanzu Kubernetes Grid Integrated Edition

After the backup has completed you should review and manage the generated backup artifacts.

### Manage Your Backup Artifact

The BBR-created backup consists of a directory containing the backup artifacts and metadata files. BBR stores each completed backup directory within the current working directory.



**Note:** The optional `-with-manifest` flag directs BBR to create a backup containing credentials. You should manage the generated backup artifact knowing it contains secrets for administering your environment.

BBR backup artifact directories are named using the following formats:

- `DIRECTOR-IP-TIMESTAMP` for the BOSH Director backups.
- `DEPLOYMENT-TIMESTAMP` for the Control Plane backup.
- `DEPLOYMENT-TIMESTAMP` for the cluster deployment backups.

Keep your backup artifacts safe by following these steps:

1. Move the backup artifacts off the jumpbox to your storage space.
2. Compress and encrypt the backup artifacts when storing them.
3. Make redundant copies of your backup and store them in multiple locations. This minimizes the risk of losing your backups in the event of a disaster.
4. Each time you redeploy Tanzu Kubernetes Grid Integrated Edition, test your backup artifact by following the procedures in:
  - ◊ [Restore the Tanzu Kubernetes Grid Integrated Edition BOSH Director](#)
  - ◊ [Restore the Tanzu Kubernetes Grid Integrated Edition Control Plane](#)
  - ◊ [Restore Tanzu Kubernetes Grid Integrated Edition Clusters](#)

### Recover from a Failing Command

If the backup fails, follow these steps:

1. Ensure that you set all the parameters in the backup command.
2. Ensure the credentials previously obtained are valid.
3. Ensure the deployment that you specify in the BBR command exists.
4. Ensure that the jumpbox can reach the BOSH Director.

5. Consult [BBR Logging](#).
6. If you see the error message: `Directory /var/vcap/store/bbr-backup already exists on instance`, run the appropriate cleanup command. See [Clean Up After a Failed Backup](#) below for more information.
7. If the backup artifact is corrupted, discard the failing artifacts and run the backup again.

## Clean Up after a Failed Backup

If your backup process fails, use the BBR cleanup script to clean up the failed run.



**Warning:** It is important to run the BBR cleanup script after a failed BBR backup run. A failed backup run might leave the BBR backup directory on the instance, causing any subsequent attempts to backup to fail. In addition, BBR might not have run the post-backup scripts, leaving the instance in a locked state.

- If the TKGI BOSH Director backup failed, run the following BBR cleanup script command to clean up:

```
bbr director --host BOSH-DIRECTOR-IP \
--username bbr --private-key-path PRIVATE-KEY-FILE \
backup-cleanup
```

Where:

- `BOSH-DIRECTOR-IP` is the address of the BOSH Director. If the BOSH Director is public, `BOSH-DIRECTOR-IP` is a URL, such as <https://my-bosh.xxx.cf-app.com>. Otherwise, this is the internal IP `BOSH-DIRECTOR-IP` which you can retrieve as shown in [Retrieve the BOSH Director Address](#) above.
- `PRIVATE-KEY-FILE` is the path to the private key file that you can create from [Bbr Ssh Credentials](#) as shown in [Download the BBR SSH Credentials](#) above. Replace the placeholder text using the information in the following table.

For example:

```
$ bbr director -host 10.0.0.5 -username bbr
-priate-key-path private-key.pem
backup-cleanup
```

- If the TKGI control plane or TKGI clusters backups fail, run the following BBR cleanup script command to clean up:

```
BOSH_CLIENT_SECRET=BOSH-CLIENT-SECRET \
bbr deployment \
--target BOSH-TARGET \
--username BOSH-CLIENT \
--deployment DEPLOYMENT-NAME \
--ca-cert PATH-TO-BOSH-CA-CERT \
backup-cleanup
```

Where:

- ◊ **BOSH-CLIENT-SECRET** is your BOSH client secret. If you do not know your BOSH Client Secret, open your BOSH Director tile, navigate to **Credentials > Bosh Commandline Credentials** and record the value for **`BOSH_CLIENT_SECRET`**.
- ◊ **BOSH-TARGET** is your BOSH Environment setting. If you do not know your BOSH Environment setting, open your BOSH Director tile, navigate to **Credentials > Bosh Commandline Credentials** and record the value for **`BOSH_ENVIRONMENT`**. You must be able to reach the target address from the workstation where you run **bbr** commands.
- ◊ **BOSH-CLIENT** is your BOSH Client Name. If you do not know your BOSH Client Name, open your BOSH Director tile, navigate to **Credentials > Bosh Commandline Credentials** and record the value for **`BOSH_CLIENT`**.
- ◊ **DEPLOYMENT-NAME** is the Tanzu Kubernetes Grid Integrated Edition BOSH deployment name that you located in the [Locate the Tanzu Kubernetes Grid Integrated Edition Deployment Names](#) section above.
- ◊ **PATH-TO-BOSH-CA-CERT** is the path to the root CA certificate that you downloaded in [Download the Root CA Certificate](#) above.

For example:

```
$ BOSH_CLIENT_SECRET=p455w0rd bbr deployment
-target bosh.example.com -username admin -deployment cf-acceptance-0
-ca-cert bosh.ca.crt
backup-cleanup
```

If the cleanup script fails, consult the following table to match the exit codes to an error message.

| V | Error                                                                                                                    |
|---|--------------------------------------------------------------------------------------------------------------------------|
| a |                                                                                                                          |
| l |                                                                                                                          |
| u |                                                                                                                          |
| e |                                                                                                                          |
| 0 | Success                                                                                                                  |
| 1 | General failure                                                                                                          |
| 8 | The post-backup unlock failed. One of your deployments might be in a bad state and require attention.                    |
| 1 | The cleanup failed. This is a non-fatal error indicating that the utility has been unable to clean up open BOSH          |
| 6 | SSH connections to a deployment's VMs. Manual cleanup might be required to clear any hanging BOSH users and connections. |

## Restoring Kubernetes Clusters Provisioned Using the Tanzu Kubernetes Grid Integrated Edition

This topic describes how to use BOSH Backup and Restore (BBR) to restore the BOSH Director, VMware Tanzu Kubernetes Grid Integrated Edition control plane, and Kubernetes clusters.

## Overview

In the event of a disaster, you might lose your environment's VMs, disks, and your IaaS network and load balancer resources as well. You can re-create your environment, configured with your saved Tanzu Kubernetes Grid Integrated Edition Ops Manager Installation settings, using your BBR backup artifacts.

Before restoring using BBR:

- Review the requirements listed in [Compatibility of Restore](#) below.
- Complete all of the steps documented in [Preparing to Restore a Backup](#) below.

Use BBR to restore the following:

- The BOSH Director plane, see [Restore the BOSH Director](#) below.
- The Tanzu Kubernetes Grid Integrated Edition control plane, see [Restore Tanzu Kubernetes Grid Integrated Edition Control Plane](#) below.
- The Tanzu Kubernetes Grid Integrated Edition clusters, see [Restore Tanzu Kubernetes Grid Integrated Edition Clusters](#) below.

## Compatibility of Restore

The following are the requirements for a backup artifact to be restorable to another environment:

- **Topology:** BBR requires the BOSH topology of a deployment to be the same in the restore environment as it was in the backup environment.
- **Naming of instance groups and jobs:** For any deployment that implements the backup and restore scripts, the instance groups and jobs must have the same names.
- **Number of instance groups and jobs:** For instance groups and jobs that have backup and restore scripts, the same number of instances must exist.

Additional considerations:

- **Limited validation:** BBR puts the backed up data into the corresponding instance groups and jobs in the restored environment, but cannot validate the restore beyond that.
- **Same Cluster:** Currently, BBR supports the in-place restore of a cluster backup artifact onto the same cluster. Migration from one cluster to another using a BBR backup artifact has not yet been validated.



**Note:** This section is for guidance only. You should always validate your backups by using the backup artifacts in a restore.

## Prepare to Restore a Backup

Before you use BBR to either back up TKGI or restore TKGI from backup, follow these steps to retrieve deployment information and credentials:

- [Verify your BBR Version](#)

- [Retrieve the BBR SSH Credentials](#)
- [Retrieve the BOSH Director Credentials](#)
- [Retrieve the UAA Client Credentials](#)
- [Retrieve the BOSH Director Address](#)
- [Download the Root CA Certificate](#)
- [Retrieve the BOSH Command Line Credentials](#)
- [Retrieve Your Cluster Deployment Names](#)

## Verify Your BBR Version

Before running BBR, verify that the installed version of BBR is compatible with the version of Ops Manager your TKGI tile is on:

1. To determine the Ops Manager BBR version requirements, see the [Ops Manager Release Notes](#) for the version of Ops Manager you are using.
2. To verify the currently installed BBR version, run the following command:

```
bbr version
```

3. If the installed BBR version does not meet the Ops Manager BBR version requirement, or BBR is not installed, you must upgrade BBR. For more information, see [Installing BOSH Backup and Restore](#).

## Retrieve the BBR SSH Credentials

There are two ways to retrieve BOSH Director credentials:

- [Ops Manager Installation Dashboard](#)
- [Ops Manager API](#)

### Ops Manager Installation Dashboard

To retrieve your BBR SSH Credentials using the Ops Manager Installation Dashboard:

1. Navigate to the Ops Manager Installation Dashboard.
2. Click the BOSH Director tile.
3. Click the **Credentials** tab.
4. Locate **Bbr Ssh Credentials**.
5. Click **Link to Credentials** next to it.
6. Copy the `private_key_pem` field value.

### Ops Manager API

To retrieve your BBR SSH Credentials using the Ops Manager API:

1. Obtain your UAA access token. For more information, see [Access the Ops Manager API](#).

2. Retrieve the **Bbr Ssh Credentials** by running the following command:

```
curl "https://OPS-MAN-FQDN/api/v0/deployed/director/credentials/bbr_ssh_credentials" \
-X GET \
-H "Authorization: Bearer UAA-ACCESS-TOKEN"
```

Where:

- ◊ **OPS-MAN-FQDN** is the fully-qualified domain name (FQDN) for your Ops Manager deployment.
- ◊ **UAA-ACCESS-TOKEN** is your UAA access token.

3. Copy the value of the **private\_key\_pem** field.

## Save the BBR SSH Credentials to File

To save the BBR SSH credentials to a private key file:

1. To reformat the copied **private\_key\_pem** value and save it to a file in the current directory:

```
printf -- "YOUR-PRIVATE-KEY" > PRIVATE-KEY-FILE
```

Where:

- ◊ **YOUR-PRIVATE-KEY** is the text of your private key.
- ◊ **PRIVATE-KEY-FILE** is the path to the private key file you are creating.

For example:

```
$ printf - "--begin rsa private key-- fake key contents --end rsa private key--" > bbr_key.pem
```

## Retrieve the BOSH Director Credentials

There are two ways to retrieve BOSH Director credentials:

- [Ops Manager Installation Dashboard](#)
- [Ops Manager API](#)

### Ops Manager Installation Dashboard

To retrieve your BOSH Director credentials using the Ops Manager Installation Dashboard, perform the following steps:

1. Navigate to the Ops Manager Installation Dashboard.
2. Click the BOSH Director tile.
3. Click the **Credentials** tab.
4. Locate **Director Credentials**.
5. Click **Link to Credentials** next to it.
6. Copy and record the value of the **password** field.

## Ops Manager API

To retrieve your BOSH Director credentials using the Ops Manager API, perform the following steps:

1. Obtain your UAA access token. For more information, see [Access the Ops Manager API](#).
2. Retrieve the **Director Credentials** by running the following command:

```
curl "https://OPS-MAN-FQDN/api/v0/deployed/director/credentials/bbr_ssh_credentials" \
-X GET \
-H "Authorization: Bearer UAA-ACCESS-TOKEN"
```

Where:

- ◊ **OPS-MAN-FQDN** is the fully-qualified domain name (FQDN) for your Ops Manager deployment.
- ◊ **UAA-ACCESS-TOKEN** is your UAA access token.

3. Copy and record the value of the **password** field.

## Retrieve the UAA Client Credentials

To obtain BOSH credentials for your BBR operations, perform the following steps:

1. From the Ops Manager Installation Dashboard, click the **Tanzu Kubernetes Grid Integrated Edition** tile.
2. Select the **Credentials** tab.
3. Navigate to **Credentials > UAA Client Credentials**.
4. Record the value for **uaa\_client\_secret**.
5. Record the value for **uaa\_client\_name**.



**Note:** You must use BOSH credentials that limit the scope of BBR activity to your cluster deployments.

## Retrieve the BOSH Director Address

You access the BOSH Director using an IP address.

To obtain your BOSH Director's IP address:

1. Open the Ops Manager Installation Dashboard.
2. Select **BOSH Director > Status**.
3. Select the listed Director IP Address.

## Log In To BOSH Director

1. If you are not using the Ops Manager VM as your jumpbox, install the latest [BOSH CLI](#) on your jumpbox.

- To log in to BOSH Director, using the IP address that you recorded above, run the following command line:

```
bosh -e BOSH-DIRECTOR-IP \
--ca-cert PATH-TO-BOSH-SERVER-CERTIFICATE log-in
```

Where:

- BOSH-DIRECTOR-IP** is the BOSH Director IP address recorded above.
- PATH-TO-BOSH-SERVER-CERTIFICATE** is the path to the root Certificate Authority (CA) certificate as outlined in [Download the Root CA Certificate](#).

- To specify **Email**, specify `director`.
- To specify **Password**, enter the **Director Credentials** that you obtained in [Retrieve the BOSH Director Credentials](#).

For example:

```
$ bosh -e 10.0.0.3
--ca-cert /var/tempest/workspaces/default/root_ca_certificate log-in Email (): director Password (): **** Successfully authenticated with UAA Succeeded
```

## Download the Root CA Certificate

To download the root CA certificate for your Tanzu Kubernetes Grid Integrated Edition deployment, perform the following steps:

- Open the Ops Manager Installation Dashboard.
- In the top right corner, click your username.
- Navigate to **Settings > Advanced**.
- Click **Download Root CA Cert**.

## Retrieve the BOSH Command Line Credentials

- Open the Ops Manager Installation Dashboard.
- Click the **BOSH Director** tile.
- In the BOSH Director tile, click the **Credentials** tab.
- Navigate to **Bosh Commandline Credentials**.
- Click **Link to Credential**.
- Copy the credential value.

## Retrieve Your Cluster Deployment Names

To locate and record a cluster deployment name, follow the steps below for each cluster:

- On the command line, run the following command to log in:

```
tkgi login -a TKGI-API -u USERNAME -k
```

Where:

- TKGI-API is the domain name for the TKGI API that you entered in **Ops Manager > Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN)**.  
For example, [api.tkgi.example.com](http://api.tkgi.example.com).
- USERNAME is your user name.

See [Logging in to Tanzu Kubernetes Grid Integrated Edition](#) for more information about the `tkgi login` command.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

## 2. Identify the cluster ID:

```
tkgi cluster CLUSTER-NAME
```

Where CLUSTER-NAME is the name of your cluster.

- From the output of this command, record the **UUID** value.
- Open the Ops Manager Installation Dashboard.
- Click the **BOSH Director** tile.
- Select the **Credentials** tab.
- Navigate to **Bosh Commandline Credentials** and click **Link to Credential**.
- Copy the credential value.
- SSH into your jumpbox. For more information about the jumpbox, see [Installing BOSH Backup and Restore](#).
- To retrieve your cluster deployment name, run the following command:

```
BOSH-CLI-CREDENTIALS deployments | grep UUID
```

Where:

- BOSH-CLI-CREDENTIALS is the full value that you copied from the BOSH Director tile in [Retrieve the BOSH Command Line Credentials](#).
- UUID is the cluster UUID that you recorded in the previous step.

## Transfer Artifacts to Your Jumpbox

To restore BOSH director, Tanzu Kubernetes Grid Integrated Edition control plane or cluster you must transfer your BBR backup artifacts from your safe storage location to your jumpbox.

1. To copy an artifact onto a jumpbox, run the following SCP command:

```
scp -r LOCAL-PATH-TO-BACKUP-ARTIFACT JUMPBOX-USER@JUMPBOX-ADDRESS:
```

Where:

- ◊ `LOCAL-PATH-TO-BACKUP-ARTIFACT` is the path to your BBR backup artifact.
- ◊ `JUMPBOX-USER` is the ssh username of the jumpbox.
- ◊ `JUMPBOX-ADDRESS` is the IP address, or hostname, of the jumpbox.

2. (Optional) Decrypt your backup artifact if the artifact is encrypted.

## Restore Kubernetes Clusters Provisioned by TKGI

Restoration of Kubernetes clusters provisioned by TKGI is a two step process: redeploy the clusters, then restore them.

Perform the following steps to redeploy the TKGI-provisioned Kubernetes clusters and restore their state from backup.

### Redeploy Clusters

Before restoring your TKGI-provisioned clusters, you must redeploy them to BOSH. To redeploy TKGI-provisioned clusters:

- If you want to redeploy all clusters simultaneously, see [Redeploy All Clusters](#).
- If you want to redeploy one cluster at a time, see [Redeploy a Single Cluster](#).

#### Redeploy All Kubernetes Clusters

To redeploy all clusters:

1. In Ops Manager, navigate to the **Tanzu Kubernetes Grid Integrated Edition** tile.
2. Click **Errands**.
3. Ensure the **Upgrade all clusters** errand is **On**. This errand redeploys all your TKGI-provisioned clusters.
4. Return to the **Installation Dashboard**.
5. Click **Review Pending Changes**, review your changes, and then click **Apply Changes**. For more information, see [Reviewing Pending Product Changes](#).

#### Redeploy a Single Kubernetes Cluster

To redeploy a TKGI-provisioned cluster through the TKGI CLI:

1. Identify the names of your TKGI-provisioned clusters:

```
tkgi clusters
```

- For each cluster you want to redeploy, run the following command:

```
tkgi upgrade-cluster CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of your Kubernetes cluster. For more information, see [Upgrade Clusters](#).

## Restore Kubernetes Clusters

After redeploying your TKGI-provisioned clusters, restore their stateless workloads and cluster state from backup by running the BOSH `restore` command from your jumpbox. Stateless workloads are tracked in the cluster etcd database, which BBR backs up.



**Warning:** BBR does not back up persistent volumes, load balancers, or other IaaS resources.



**Warning:** When you restore a cluster, etcd is stopped in the API server. During this process, only currently-deployed clusters function, and you cannot create new workloads.

To restore a cluster:

- Move the cluster backup artifact to a folder from which you will run the BBR restore process.
- SSH into your jumpbox. For more information about the jumpbox, see [Configure Your Jumpbox](#) in *Installing BOSH Backup and Restore*.
- Run the following command:

```
BOSH_CLIENT_SECRET=BOSH-CLIENT-SECRET \
nohup bbr deployment --target BOSH-TARGET \
--username BOSH-CLIENT --deployment DEPLOYMENT-NAME \
--ca-cert PATH-TO-BOSH-SERVER-CERT \
restore \
--artifact-path PATH-TO-DEPLOYMENT-BACKUP
```

Where:

- `BOSH-CLIENT-SECRET` is the `BOSH_CLIENT_SECRET` property. This value is in the BOSH Director tile under **Credentials > Bosh Commandline Credentials**.
- `BOSH-TARGET` is the `BOSH_ENVIRONMENT` property. This value is in the BOSH Director tile under **Credentials > Bosh Commandline Credentials**. You must be able to reach the target address from the workstation where you run `bbr` commands.
- `BOSH-CLIENT` is the `BOSH_CLIENT` property. This value is in the BOSH Director tile under **Credentials > Bosh Commandline Credentials**.
- `DEPLOYMENT-NAME` is the cluster BOSH deployment name that you recorded in [Retrieve Your Cluster Deployment Names](#) above.
- `PATH-TO-BOSH-CA-CERT` is the path to the root CA certificate that you downloaded in the [Download the Root CA Certificate](#) section above.

- ◊ **PATH-TO-DEPLOYMENT-BACKUP** is the path to your deployment backup. Make sure you have transferred your artifact into your jumpbox as described in [Transfer Artifacts to Jumpbox](#) above.

For example:

```
$ BOSH_CLIENT_SECRET=p455w0rd
nohup bbr deployment
-target bosch.example.com
-username admin
-deployment service-instance_3839394
-ca-cert bosch.ca.cert
restore
-artifact-path deployment-backup
```



**Note:** The BBR restore command can take a long time to complete. The BBR restore command above uses `nohup` and the restore process is run within your SSH session. If you instead run the BBR command in a `screen` or `tmux` session the task will run separately from your SSH session and will continue to run, even if your SSH connection to the jumpbox fails.

4. To cancel a running `bbr restore`, see [Cancel a Restore](#) below.
5. After you restore a Kubernetes cluster, you must register its workers with their control plane nodes by following the [Register Restored Worker VMs](#) steps below.
6. If your Tanzu Kubernetes Grid Integrated Edition cluster restore fails, do one or more of the following:
  - ◊ Run the command again, adding the `--debug` flag to activate debug logs. For more information, see [BBR Logging](#).
  - ◊ Follow the steps in [Resolve a Failing BBR Restore Command](#) below.

Be sure to complete the steps in [Clean Up After a Failed Restore](#) below.

## Register Restored Worker VMs

After restoring a Kubernetes cluster, you must register all of the cluster's worker nodes with their control plane nodes. To register cluster worker nodes, complete the following:

1. [Delete Nodes](#)
2. [Restart kubelet](#)

## Delete Nodes

To delete a cluster's restored nodes:

1. To determine your cluster's namespace, run the following command:

```
kubectl get all --all-namespaces
```

2. To retrieve the list of worker nodes in the cluster, run the following command:

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

Document the worker node names listed in the `NAME` column. The worker nodes should all be listed with a status of `NotReady`.

3. To delete a node, run the following:

```
kubectl delete node NODE-NAME
```

Where `NODE-NAME` is a node `NAME` returned by the `kubectl get nodes` command.

4. Repeat the preceding `kubectl delete node` step for each of your cluster's nodes.

## Restart kubelet

To restart `kubelet` on your worker node VMs:

1. To restart `kubelet` on all of your cluster's worker node VMs, run the following command:

```
bosh ssh -d DEPLOYMENT-NAME worker -c 'sudo /var/vcap/bosh/bin/monit restart kubelet'
```

Where `DEPLOYMENT-NAME` is the cluster BOSH deployment name that you recorded in [Retrieve Your Cluster Deployment Names](#) above.

2. To confirm all worker nodes in your cluster have been restored to a `Ready` state, run the following command:

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

## Resolve a Failing BBR Restore Command

To resolve a failing BBR restore command:

1. Ensure that you set all the parameters in the command.
2. Ensure that the BOSH Director credentials are valid.
3. Ensure that the specified BOSH deployment or Director exists.
4. Ensure that the jumpbox can reach the BOSH Director.
5. Ensure the source backup artifact is compatible with the target BOSH deployment or Director.
6. If you see the error message `Directory /var/vcap/store/bbr-backup already exists on`

`instance`, run the relevant commands from the [Clean up After Failed Restore](#) section of this topic.

7. See the [BBR Logging](#) topic.

## Cancel a Restore

If you must cancel a restore, perform the following steps:

1. Terminate the BBR process by pressing Ctrl-C and typing `yes` to confirm.
2. Perform the procedures in the [Clean up After Failed Restore](#) section to support future restores. Stopping a restore can leave the system in an unusable state and prevent future restores.

## Clean Up After a Failed Restore

If a BBR restore process fails, BBR might not have run the post-restore scripts, potentially leaving the instance in a locked state. Additionally, the BBR restore folder might remain on the target instance and subsequent restore attempts might also fail.

- To resolve issues following a failed BOSH Director restore, run the following BBR command:

```
nohup bbr director \
--host BOSH-DIRECTOR-IP \
--username bbr \
--private-key-path PRIVATE-KEY-FILE \
restore-cleanup
```

Where:

- ◊ `BOSH-DIRECTOR-IP` is the address of the BOSH Director. If the BOSH Director is public, `BOSH-DIRECTOR-IP` is a URL, such as <https://my-bosh.xxx.cf-app.com>. Otherwise, this is the internal IP `BOSH-DIRECTOR-IP` which you can retrieve as shown in [Retrieve the BOSH Director Address](#) above.
- ◊ `PRIVATE-KEY-FILE` is the path to the private key file that you can create from [Bbr Ssh Credentials](#) as shown in [Download the BBR SSH Credentials](#) above.

For example:

```
$ nohup bbr director
-target 10.0.0.5
-username bbr
--private-key-path private.pem
restore-cleanup
```

- To resolve issues following a failed control plane restore, run the following BBR command:

```
BOSH_CLIENT_SECRET=BOSH-CLIENT-SECRET \
bbr deployment \
```

```
--target BOSH-TARGET \
--username BOSH-CLIENT \
--deployment DEPLOYMENT-NAME \
--ca-cert PATH-TO-BOSH-CA-CERT \
restore-cleanup
```

Where:

- ◊ `BOSH-CLIENT-SECRET` is the value for `BOSH_CLIENT_SECRET` retrieved in [Download the BOSH Commandline Credentials](#) above.
- ◊ `BOSH-TARGET` is the value for `BOSH_ENVIRONMENT` retrieved in [Download the BOSH Commandline Credentials](#) above. You must be able to reach the target address from the workstation where you run `bbr` commands.
- ◊ `BOSH-CLIENT` is the value for `BOSH_CLIENT` retrieved in [Download the BOSH Commandline Credentials](#) above.
- ◊ `DEPLOYMENT-NAME` is the name retrieved in [Retrieve Your Cluster Deployment Name](#) above.
- ◊ `PATH-TO-BOSH-CA-CERT` is the path to the root CA certificate that you downloaded in [Download the Root CA Certificate](#) above.

For example:

```
$ BOSH_CLIENT_SECRET=p455w0rd
bbr deployment
-target bash.example.com
-username admin
-deployment pivotal-container-service-453f2f
-ca-cert bash.ca.crt
restore-cleanup
```

- To resolve issues following a failed cluster restore, run the following BBR command:

```
BOSH_CLIENT_SECRET=BOSH-CLIENT-SECRET \
bbr deployment \
--target BOSH-TARGET \
--username BOSH-CLIENT \
--deployment DEPLOYMENT-NAME \
--ca-cert PATH-TO-BOSH-CA-CERT \
restore-cleanup
```

Where:

- ◊ `BOSH-CLIENT-SECRET` is the value for `BOSH_CLIENT_SECRET` retrieved in [Download the BOSH Commandline Credentials](#).
- ◊ `BOSH-TARGET` is the value for `BOSH_ENVIRONMENT` retrieved in [Download the BOSH Commandline Credentials](#). You must be able to reach the target address from the

workstation where you run `bbr` commands.

- ◊ `BOSH-CLIENT` is the value for `BOSH_CLIENT` retrieved in [Download the BOSH Commandline Credentials](#).
- ◊ `DEPLOYMENT-NAME` is the name retrieved in [Retrieve Your Cluster Deployment Names](#) above.
- ◊ `PATH-TO-BOSH-CA-CERT` is the path to the root CA certificate that you downloaded in [Download the Root CA Certificate](#).

For example:

```
$ BOSH_CLIENT_SECRET=p455w0rd
bbr deployment
-target bosh.example.com
-username admin
-deployment pivotal-container-service-453f2f
-ca-cert bosh.ca.crt
restore-cleanup
```

## Backing Up and Restoring the TKGI Management Plane

This section describes how to back up and restore VMware Tanzu Kubernetes Grid Integrated Edition Management Plane.

### Overview

Back up and restore of the TKGI Management Plane includes the following components:

- Ops Manager configuration, including the BOSH Director and TKGI tiles.
- The TKGI Management Plane VMs, including the [BOSH Director VM](#), and the [TKGI Control Plane VM](#).

You use Ops Manager to backup and restore the BOSH Director and TKGI tiles. You use [BOSH Backup and Restore \(BBR\)](#) to backup and restore the TKGI Management Plane VMs. Restoring the Ops Manager VM is a manual process.

To backup and restore the TKGI Management Plane, see the following topics:

- [Install and Configure BBR](#).
- [Backing Up TKGI Management Plane Components](#).
- [Restoring TKGI Management Plane Components](#).

### Testing Considerations

As part of your TKGI backup and restore planning and testing, consider the following test scenario.

- Export Ops Manager configuration.
- Take backup of TKGI Management Plane using BBR.
- Power off the Ops Manager, BOSH, and TKGI Control Plane VMs.
- Deploy a new Ops Manager VM and import the exported configuration.
- Restore BOSH and TKGI VMs using BBR.

On restore of all TKGI Management Plane components, there should be no loss of data for configurations included in the backups. Any TKGI configuration changes made after the backup was taken are not restored since they were made after the backup.

## Installing and Configuring BOSH Backup and Restore

This topic describes how to install BOSH Backup and Restore (BBR).

### Overview

To install BBR, first validate that your jumpbox VM is a valid BOSH backup host, then copy the `bbr` executable to the jumpbox and configure BBR.

For more information, see [Install and Configure BOSH Backup and Restore](#) below.

After installing BBR, you can run `bbr` commands to back up and restore your Tanzu Kubernetes Grid Integrated Edition deployment.

For more information about using BOSH Backup and Restore, see:

- To backup and restore Kubernetes clusters provisioned by TKGI, see [Backing Up and Restoring TKGI Clusters](#).
- To backup and restore TKGI Management Plane Components, see [Backing Up and Restoring TKGI Management Plane](#).

### Prerequisites

Using BBR requires the following:

- A jumpbox.

A jumpbox is a separate, hardened server on your network that provides a controlled means of accessing the other VMs on your network. See the [jumpbox-deployment GitHub repository](#) for an example jumpbox deployment.

You must have a jumpbox before you can install BBR to the jumpbox.

- The OpenBSD version of netcat must be installed on the jumpbox host.
- A `bbr` executable file. You must have the correct BBR executable version for your TKGI installation.
  - ◊ To determine the correct version of BBR for your deployment, see the [Tanzu Kubernetes Grid Integrated Edition Release Notes](#).

- ◊ To download a BBR installation file, see [BOSH Backup and Restore](#) on the VMware Tanzu Network.



**Note:** BBR does not support SSH gateways.

## Install and Configure BOSH Backup and Restore

To install and configure BBR:

1. [Configure Your Jumpbox for BBR](#)
2. [Install BBR on Your Jumpbox](#)
3. [Verify Your BBR Installation](#)
4. [Configure BBR Logging](#)

### Configure Your Jumpbox for BBR

Your jumpbox must meet or exceed minimum BBR requirements. You can use the Ops Manager VM as your jumpbox if it can be configured to meet all of the requirements below.

To configure your jumpbox to meet BBR requirements:

1. Size the jumpbox to have sufficient storage space for your backups.
2. Ensure the jumpbox can communicate with the network containing your Tanzu Kubernetes Grid Integrated Edition deployment.

BBR uses SSH to orchestrate the backup of your Tanzu Kubernetes Grid Integrated Edition instances using port 22 by default.

3. Configure the jumpbox to be in the same network as the deployed VMs.

BBR connects to the deployed VMs at their private IP addresses.

4. Ensure there is minimal network latency between the jumpbox and the source VMs BBR backs up.

### Install BBR on Your Jumpbox

To install the `bbr` executable to your jumpbox:

1. Download the latest compatible [BOSH Backup and Restore release](#) from the VMware Tanzu Network.
2. To add executable permissions to the `bbr` binary file, run the following command:

```
chmod a+x bbr
```

3. To securely copy the `bbr` binary file to your jumpbox, run the following command:

```
scp LOCAL-PATH-TO-BBR/bbr JUMPBOX-USER@JUMPBOX-ADDRESS:
```

Where:

- ◊ `LOCAL-PATH-TO-BBR` is the path to the `bbr` binary you downloaded from VMware Tanzu Network.
- ◊ `JUMPBOX-USER` is the ssh username for connecting to the jumpbox.
- ◊ `JUMPBOX-ADDRESS` is the IP address, or hostname, of the jumpbox.

## Verify Your BBR Installation

To verify that BBR is installed:

1. Run the following command:

```
bbr version
```

Verify the returned BBR version.

## Configure BBR Logging

BBR writes backup and restore logs to the current directory in a file named `bbr-TIMESTAMP.err.log`.

By default BBR writes errors associated with stack traces to the log file.

BBR also reports default information about the backup and restore run:

- The backup and restore scripts that it finds.
- When it starts or finishes a stage, such as `pre-backup scripts` or `backup scripts`.
- When the process is complete.
- When any error occurs.

To troubleshoot a failed BBR run, enable verbose logging. When executed in verbose mode, BBR reports the following additional information:

- Logs about the API requests made to the BOSH server.
- All commands executed on remote instances.
- All commands executed on local environment.
- Standard in and standard out streams for the backup and restore scripts when they are executed.

To enable verbose logging, use the optional `--debug` flag.

## Backing Up TKGI Management Plane Components

This topic describes how to use BOSH Backup and Restore (BBR) to back up the VMware Tanzu Kubernetes Grid Integrated Edition Management Plane components.

## Overview

The BOSH Director, Tanzu Kubernetes Grid Integrated Edition Control Plane, and cluster deployments include custom backup and restore scripts which encapsulate the correct procedure for

backing up and restoring the Director and Control Plane.

BBR orchestrates running the backup and restore scripts and transferring the generated backup artifacts to and from a backup directory. If configured correctly, BBR can use TLS to communicate securely with backup targets.

- To perform a restore of the BOSH Director, see [Restore the BOSH Director](#).
- To perform a restore of the TKGI Control Plane, see [Restore the Tanzu Kubernetes Grid Integrated Edition Control Plane](#).

To view the BBR release notes, see the Cloud Foundry documentation, [BOSH Backup and Restore Release Notes](#).

## Recommendations

VMware recommends:

- Follow the full procedure documented in this topic when creating a backup. This ensures that you always have a consistent backup of Ops Manager and Tanzu Kubernetes Grid Integrated Edition to restore from.
- Back up frequently, especially before upgrading your Tanzu Kubernetes Grid Integrated Edition deployment.
- For BOSH v270.0 and above (currently in Ops Manager 2.7), prune the BOSH blobstore by running `bosh clean-up --all` prior to running a backup of the BOSH director. This removes all unused resources, including packages compiled against older stemcell versions, which can result in a smaller, faster backup of the BOSH Director. For more information see the [clean-up](#) command.



**Note:** The command `bosh clean-up -all` is a destructive operation and can remove resources that are unused but needed. For example, if an On-Demand Service Broker such as Tanzu Kubernetes Grid Integrated Edition is deployed **and** no service instances have been created, the releases needed to create a service instance will be categorized as unused and removed.

## Prepare to Back Up

Before you use BBR to either back up TKGI or restore TKGI from backup, follow these steps to retrieve deployment information and credentials:

- [Verify your BBR Version](#)
- [Retrieve the BBR SSH Credentials](#)
- [Retrieve the BOSH Director Credentials](#)
- [Retrieve the UAA Client Credentials](#)
- [Retrieve the BOSH Director Address](#)
- [Download the Root CA Certificate](#)
- [Retrieve the BOSH Command Line Credentials](#)

- [Retrieve Your Cluster Deployment Names](#)

## Verify Your BBR Version

Before running BBR, verify that the installed version of BBR is compatible with the version of Ops Manager your TKGI tile is on:

1. To determine the Ops Manager BBR version requirements, see the [Ops Manager Release Notes](#) for the version of Ops Manager you are using.
2. To verify the currently installed BBR version, run the following command:

```
bbr version
```

3. If the installed BBR version does not meet the Ops Manager BBR version requirement, or BBR is not installed, you must upgrade BBR. For more information, see [Installing BOSH Backup and Restore](#).

## Retrieve the BBR SSH Credentials

There are two ways to retrieve BOSH Director credentials:

- [Ops Manager Installation Dashboard](#)
- [Ops Manager API](#)

### Ops Manager Installation Dashboard

To retrieve your BBR SSH Credentials using the Ops Manager Installation Dashboard:

1. Navigate to the Ops Manager Installation Dashboard.
2. Click the BOSH Director tile.
3. Click the **Credentials** tab.
4. Locate **Bbr Ssh Credentials**.
5. Click **Link to Credentials** next to it.
6. Copy the `private_key_pem` field value.

### Ops Manager API

To retrieve your BBR SSH Credentials using the Ops Manager API:

1. Obtain your UAA access token. For more information, see [Access the Ops Manager API](#).
2. Retrieve the **Bbr Ssh Credentials** by running the following command:

```
curl "https://OPS-MAN-FQDN/api/v0/deployed/director/credentials/bbr_ssh_credentials" \
-X GET \
-H "Authorization: Bearer UAA-ACCESS-TOKEN"
```

Where:

- ◊ `OPS-MAN-FQDN` is the fully-qualified domain name (FQDN) for your Ops Manager

deployment.

- ◊ `UAA-ACCESS-TOKEN` is your UAA access token.
3. Copy the value of the `private_key_pem` field.

## Save the BBR SSH Credentials to File

To save the BBR SSH credentials to a private key file:

1. To reformat the copied `private_key_pem` value and save it to a file in the current directory:

```
printf -- "YOUR-PRIVATE-KEY" > PRIVATE-KEY-FILE
```

Where:

- ◊ `YOUR-PRIVATE-KEY` is the text of your private key.
- ◊ `PRIVATE-KEY-FILE` is the path to the private key file you are creating.

For example:

```
$ printf - "--begin rsa private key-- fake key contents --end rsa private key--" > bbr_key.pem
```

## Retrieve the BOSH Director Credentials

There are two ways to retrieve BOSH Director credentials:

- [Ops Manager Installation Dashboard](#)
- [Ops Manager API](#)

### Ops Manager Installation Dashboard

To retrieve your BOSH Director credentials using the Ops Manager Installation Dashboard, perform the following steps:

1. Navigate to the Ops Manager Installation Dashboard.
2. Click the BOSH Director tile.
3. Click the **Credentials** tab.
4. Locate **Director Credentials**.
5. Click **Link to Credentials** next to it.
6. Copy and record the value of the `password` field.

### Ops Manager API

To retrieve your BOSH Director credentials using the Ops Manager API, perform the following steps:

1. Obtain your UAA access token. For more information, see [Access the Ops Manager API](#).
2. Retrieve the **Director Credentials** by running the following command:

```
curl "https://OPS-MAN-FQDN/api/v0/deployed/director/credentials/bbr_ssh_credential"
```

```
ials" \
-X GET \
-H "Authorization: Bearer UAA-ACCESS-TOKEN"
```

Where:

- ◊ **OPS-MAN-FQDN** is the fully-qualified domain name (FQDN) for your Ops Manager deployment.
- ◊ **UAA-ACCESS-TOKEN** is your UAA access token.

3. Copy and record the value of the **password** field.

## Retrieve the UAA Client Credentials

To obtain BOSH credentials for your BBR operations, perform the following steps:

1. From the Ops Manager Installation Dashboard, click the **Tanzu Kubernetes Grid Integrated Edition** tile.
2. Select the **Credentials** tab.
3. Navigate to **Credentials > UAA Client Credentials**.
4. Record the value for **uaa\_client\_secret**.
5. Record the value for **uaa\_client\_name**.



**Note:** You must use BOSH credentials that limit the scope of BBR activity to your cluster deployments.

## Retrieve the BOSH Director Address

You access the BOSH Director using an IP address.

To obtain your BOSH Director's IP address:

1. Open the Ops Manager Installation Dashboard.
2. Select **BOSH Director > Status**.
3. Select the listed Director IP Address.

## Log In To BOSH Director

1. If you are not using the Ops Manager VM as your jumpbox, install the latest [BOSH CLI](#) on your jumpbox.
2. To log in to BOSH Director, using the IP address that you recorded above, run the following command line:

```
bosh -e BOSH-DIRECTOR-IP \
--ca-cert PATH-TO-BOSH-SERVER-CERTIFICATE log-in
```

Where:

- ◊ **BOSH-DIRECTOR-IP** is the BOSH Director IP address recorded above.

- ◊ **PATH-TO-BOSH-SERVER-CERTIFICATE** is the path to the root Certificate Authority (CA) certificate as outlined in [Download the Root CA Certificate](#).
3. To specify **Email**, specify `director`.
  4. To specify **Password**, enter the **Director Credentials** that you obtained in [Retrieve the BOSH Director Credentials](#).

For example:

```
$ bosh -e 10.0.0.3
-ca-cert /var/tempest/workspaces/default/root_ca_certificate log-in Email (): d
irector Password (): **** Successfully authenticated with UAA Su
cceeded
```

## Download the Root CA Certificate

To download the root CA certificate for your Tanzu Kubernetes Grid Integrated Edition deployment, perform the following steps:

1. Open the Ops Manager Installation Dashboard.
2. In the top right corner, click your username.
3. Navigate to **Settings > Advanced**.
4. Click **Download Root CA Cert**.

## Retrieve the BOSH Command Line Credentials

1. Open the Ops Manager Installation Dashboard.
2. Click the **BOSH Director** tile.
3. In the BOSH Director tile, click the **Credentials** tab.
4. Navigate to **Bosh Commandline Credentials**.
5. Click **Link to Credential**.
6. Copy the credential value.

## Retrieve Your Cluster Deployment Names

To locate and record a cluster deployment name, follow the steps below for each cluster:

1. On the command line, run the following command to log in:

```
tkgi login -a TKGI-API -u USERNAME -k
```

Where:

- ◊ **TKGI-API** is the domain name for the TKGI API that you entered in **Ops Manager > Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN)**.  
For example, `api.tkgi.example.com`.
- ◊ **USERNAME** is your user name.

See [Logging in to Tanzu Kubernetes Grid Integrated Edition](#) for more information about the `tkgi login` command.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

2. Identify the cluster ID:

```
tkgi cluster CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of your cluster.

3. From the output of this command, record the **UUID** value.
4. Open the Ops Manager Installation Dashboard.
5. Click the **BOSH Director** tile.
6. Select the **Credentials** tab.
7. Navigate to **Bosh Commandline Credentials** and click **Link to Credential**.
8. Copy the credential value.
9. SSH into your jumpbox. For more information about the jumpbox, see [Installing BOSH Backup and Restore](#).
10. To retrieve your cluster deployment name, run the following command:

```
BOSH-CLI-CREDENTIALS deployments | grep UUID
```

Where:

- ◊ `BOSH-CLI-CREDENTIALS` is the full value that you copied from the BOSH Director tile in [Retrieve the BOSH Command Line Credentials](#).
- ◊ `UUID` is the cluster UUID that you recorded in the previous step.

## Back Up Tanzu Kubernetes Grid Integrated Edition

To back up your Tanzu Kubernetes Grid Integrated Edition environment you must first connect to your jumpbox before executing `bbr` backup commands.

### Connect to Your Jumpbox

You can establish a connection to your jumpbox in one of the following ways:

- [Connect with SSH](#)
- [Connect with BOSH\\_ALL\\_PROXY](#)

For general information about the jumpbox, see [Installing BOSH Backup and Restore](#).

## Connect with SSH

To connect to your jumpbox with SSH, do one of the following:

- To use the Ops Manager VM as your jumpbox:
  1. Log in to the Ops Manager VM. For more information, see [Log in to the Ops Manager VM with SSH](#) in *Advanced Troubleshooting with the BOSH CLI*.
- To connect to your jumpbox using the command line:
  1. Run the following command:

```
ssh -i PATH-TO-KEY JUMPBOX-USERNAME@JUMPBOX-ADDRESS
```

Where:

- **PATH-TO-KEY** is the local path to your private key for the jumpbox host.
- **JUMPBOX-USERNAME** is your jumpbox username.
- **JUMPBOX-ADDRESS** is the address of the jumpbox.



**Note:** If you connect to your jumpbox with SSH, you must run the BBR commands in the following sections from within your jumpbox.

## Connect with BOSH\_ALL\_PROXY

You can use the **BOSH\_ALL\_PROXY** environment variable to open an SSH tunnel with SOCKS5 to your jumpbox. This tunnel enables you to forward requests from your local machine to the BOSH Director through the jumpbox. When **BOSH\_ALL\_PROXY** is set, BBR always uses its value to forward requests to the BOSH Director.



**Note:** For the following procedures to work, ensure the SOCKS port is not already in use by a different tunnel or process.

To connect with **BOSH\_ALL\_PROXY**, do one of the following:

- To establish a tunnel separate from the BOSH CLI:
  1. Establish the tunnel and make it available on a local port by running the following command:

```
ssh -4 -D SOCKS-PORT -fNC JUMPBOX-USERNAME@JUMPBOX-ADDRESS -i JUMPBOX-KEY
-FILE -o ServerAliveInterval=60
```

Where:

- **SOCKS-PORT** is the local SOCKS port.
- **JUMPBOX-USERNAME** is your jumpbox username.

- `JUMPBOX-ADDRESS` is the address of the jumpbox.
- `JUMPBOX-KEY-FILE` is the local SSH private key for accessing the jumpbox.

For example:

```
$ ssh -4 -D 12345 -fNC jumpbox@203.0.113.0 -i jumpbox.key -o ServerAliveInterval=60
```

2. Provide the BOSH CLI with access to the tunnel through `BOSH_ALL_PROXY` by running the following command:

```
export BOSH_ALL_PROXY=socks5://localhost:SOCKS-PORT
```

Where is `SOCKS-PORT` is your local SOCKS port.

- To establish a tunnel using the BOSH CLI:

1. Provide the BOSH CLI with the necessary SSH credentials to create the tunnel by running the following command:

```
export BOSH_ALL_PROXY=ssh+socks5://JUMPBOX-USERNAME@JUMPBOX-ADDRESS:SOCKS-PORT?private_key=JUMPBOX-KEY-FILE
```

Where:

- `JUMPBOX-USERNAME` is your jumpbox username.
- `JUMPBOX-ADDRESS` is the address of the jumpbox.
- `SOCKS-PORT` is your local SOCKS port.
- `JUMPBOX-KEY-FILE` is the local SSH private key for accessing the jumpbox.

For example:

```
$ export BOSH_ALL_PROXY=ssh+socks5://jumpbox@203.0.113.0:12345?private_key=jumpbox.key
```



**Note:** Using `BOSH_ALL_PROXY` can result in longer backup and restore times because of network performance degradation. All operations must pass through the proxy which means moving backup artifacts can be significantly slower.



**Warning:** In BBR v1.5.0 and earlier, the tunnel created by the BOSH CLI does not include the `ServerAliveInterval` flag. This might result in your SSH connection timing out when transferring large artifacts. In BBR v1.5.1, the `ServerAliveInterval` flag is included. For more information, see [bosh-backup-and-restore v1.5.1](#) on GitHub.

## Back Up Ops Manager Installation Settings

To ensure your BBR backup is reliable, you should also frequently export your Ops Manager installation settings as a backup.

There are two ways to export Ops Manager installation settings:

- [Export settings using the Ops Manager UI](#)
- [Export settings using the Ops Manager API](#)



**Note:** If you want to automate the back up process, you can use the Ops Manager API to export your installation settings.

When exporting your installation settings, keep in mind the following:

- You should always export your installation settings before following the steps in the [Restore the BOSH Director](#) section of the *Restoring Tanzu Kubernetes Grid Integrated Edition* topic.
- You can only export Ops Manager installation settings after you have deployed at least once.
- Your Ops Manager settings export is only a backup of Ops Manager configuration settings. The export is not a backup of your VMs or any external MySQL databases.
- Your Ops Manager settings export is encrypted. Make sure you keep track of your Decryption Passphrase because this is needed to restore the Ops Manager settings.

## Export Settings Using the Ops Manager UI

To export your Ops Manager installation settings using the Ops Manager UI, perform the following steps:

1. From the **Installation Dashboard** in the Ops Manager interface, click your username at the top right navigation.
2. Select **Settings**.
3. Select **Export Installation Settings**.
4. Click **Export Installation Settings**.

## Export Settings Using the Ops Manager API

To export your Ops Manager installation settings using the Ops Manager API, perform the following steps:

1. To export your installation settings using the Ops Manager API, run the following command:

```
curl https://OPS-MAN-FQDN/api/v0/installation_asset_collection \
-H "Authorization: Bearer UAA-ACCESS-TOKEN" > installation.zip
```

Where:

- ◊ **OPS-MAN-FQDN** is the fully-qualified domain name (FQDN) for your Ops Manager deployment.
- ◊ **UAA-ACCESS-TOKEN** is your UAA access token. For more information, see Access the API.

## Back Up the Tanzu Kubernetes Grid Integrated Edition BOSH

## Director

To back up BOSH Director you will validate your current configuration, then execute the `bbr` backup command.

### Validate the Tanzu Kubernetes Grid Integrated Edition BOSH Director

1. To confirm that your BOSH Director is reachable and has the correct BBR scripts, run the following command:

```
bbr director --host BOSH-DIRECTOR-IP --username bbr \
--private-key-path PRIVATE-KEY-FILE pre-backup-check
```

Where:

- ❖ `BOSH-DIRECTOR-IP` is the address of the BOSH Director. If the BOSH Director is public, `BOSH-DIRECTOR-IP` is a URL, such as <https://my-bosh.xxx.cf-app.com>. Otherwise, this is the internal IP `BOSH-DIRECTOR-IP` which you can retrieve as shown in [Retrieve the BOSH Director Address](#).
- ❖ `PRIVATE-KEY-FILE` is the path to the private key file that you can create from [Bbr Ssh Credentials](#) as shown in [Download the BBR SSH Credentials](#).

For example:

```
$ bbr director -host 10.0.0.5 -username bbr
--private-key-path private-key.pem pre-backup-check
```

2. If the pre-backup check command fails, perform the following actions:

1. Run the command again, adding the `--debug` flag to enable debug logs. For more information, see [BBR Logging](#).
2. Make any correction suggested in the output and run the pre-backup check again.

### Back Up the Tanzu Kubernetes Grid Integrated Edition BOSH Director

1. If the pre-backup check succeeds, run the BBR backup command from your jumpbox to back up the TKGI BOSH Director:

```
bbr director --host BOSH-DIRECTOR-IP --username bbr \
--private-key-path PRIVATE-KEY-FILE backup
```

Where:

- ❖ `BOSH-DIRECTOR-IP` is the address of the BOSH Director. If the BOSH Director is public, `BOSH-DIRECTOR-IP` is a URL, such as <https://my-bosh.xxx.cf-app.com>. Otherwise, this is the internal IP. See [Retrieve the BOSH Director Address](#) for more information.
- ❖ `PRIVATE-KEY-FILE` is the path to the private key file that you can create from [Bbr Ssh Credentials](#) as shown in [Download the BBR SSH Credentials](#).

For example:

```
$ bbr director -host 10.0.0.5 -username bbr
-private-key-path private-key.pem backup
```



**Note:** The BBR backup command can take a long time to complete. You can run it independently of the SSH session so that the process can continue running even if your connection to the jumpbox fails. The command above uses `nohup`, but you can run the command in a `screen` or `tmux` session instead.

2. If the command completes successfully, follow the steps in [Manage Your Backup Artifact](#) below.
3. If the backup command fails, perform the following actions:
  - ◊ Run the command again, adding the `--debug` flag to enable debug logs. For more information, see [BBR Logging](#).
  - ◊ Follow the steps in [Recover from a Failing Command](#).

## Back Up the Tanzu Kubernetes Grid Integrated Edition Control Plane

To back up your Tanzu Kubernetes Grid Integrated Edition Control Plane you will validate the Control Plane, then execute the `bbr` backup command.

### Locate the Tanzu Kubernetes Grid Integrated Edition Deployment Name

Locate and record your Tanzu Kubernetes Grid Integrated Edition BOSH deployment name as follows:

1. Open an SSH connection to either your jumpbox, as described in the previous section, or the Ops Manager VM. For instructions on how to SSH into the Ops Manager VM, see [Log in to the Ops Manager VM with SSH](#) in *Advanced Troubleshooting with the BOSH CLI*.
2. On the command line, run the following command to retrieve your Tanzu Kubernetes Grid Integrated Edition BOSH deployment name.

```
BOSH-CLI-CREDENTIALS deployments | grep pivotal-container-service
```

Where `BOSH-CLI-CREDENTIALS` is the full value that you copied from the BOSH Director tile in [Retrieve the BOSH Command Line Credentials](#).

For example:

```
$ BOSH_CLIENT=ops_manager BOSH_CLIENT_SECRET=p455w0rd BOSH_CA_CERT=/v
ar/tempest/workspaces/default/root_ca_certificate BOSH_ENVIRONMENT=10.
0.0.5 bosh deployments | grep pivotal-container-service

pivotal-container-service-51f08f6402aaa960f041 backup-and-re
```

```

store-sdk/1.9.0 bosh-google-kvm-ubuntu-jammy-go_agent/1.75

service-instance_4ffeb5b5-5182-4faa-9d92-696d97cc9ae1 bosh-dns/1.10
.0 bosh-google-kvm-ubuntu-jammy-go_agent/1.75

pivotal-container-service-51f08f6402aaa960f041

```

- Review the returned output. The Tanzu Kubernetes Grid Integrated Edition BOSH deployment name begins with `pivotal-container-service` and includes a unique identifier. In the example output above, the BOSH deployment name is `pivotal-container-service-51f08f6402aaa960f041`.

## Validate the Tanzu Kubernetes Grid Integrated Edition Control Plane

- To confirm that your TKGI control plane is reachable and has a deployment that can be backed up, run the BBR pre-backup check command:

```

BOSH_CLIENT_SECRET=BOSH-CLIENT-SECRET bbr deployment \
--target BOSH-TARGET --username BOSH-CLIENT --deployment DEPLOYMENT-NAME \
--ca-cert PATH-TO-BOSH-SERVER-CERT \
pre-backup-check

```

Where:

- `BOSH-CLIENT-SECRET` is your BOSH client secret. If you do not know your BOSH Client Secret, open your BOSH Director tile, navigate to **Credentials > Bosh Commandline Credentials** and record the value for `BOSH_CLIENT_SECRET`.
- `BOSH-TARGET` is your BOSH Environment setting. If you do not know your BOSH Environment setting, open your BOSH Director tile, navigate to **Credentials > Bosh Commandline Credentials** and record the value for `BOSH_ENVIRONMENT`. You must be able to reach the target address from the workstation where you run `bbr` commands.
- `BOSH-CLIENT` is your BOSH Client Name. If you do not know your BOSH Client Name, open your BOSH Director tile, navigate to **Credentials > Bosh Commandline Credentials** and record the value for `BOSH_CLIENT`.
- `DEPLOYMENT-NAME` is the Tanzu Kubernetes Grid Integrated Edition BOSH deployment name that you located in the [Locate the Tanzu Kubernetes Grid Integrated Edition Deployment Name](#) section above.
- `PATH-TO-BOSH-CA-CERT` is the path to the root CA certificate that you downloaded in [Download the Root CA Certificate](#) above.

For example:

```

$ BOSH_CLIENT_SECRET=p455w0rd bbr deployment
-target bosh.example.com -username admin -deployment cf-acceptance-
-ca-cert bosh.ca.cert

```

```
pre-backup-check
```

2. If the pre-backup check command fails, perform the following actions:
  1. Run the command again, adding the `--debug` flag to enable debug logs. For more information, see [BBR Logging](#).
  2. Make any correction suggested in the output and run the pre-backup check again. For example, the deployment that you selected might not have the correct backup scripts, or the connection to the BOSH Director failed.

## Back Up the Tanzu Kubernetes Grid Integrated Edition Control Plane

If the pre-backup check succeeds, run the BBR backup command.

1. To back up the TKGI control plane, run the following BBR backup command from your jumpbox:

```
BOSH_CLIENT_SECRET=BOSH-CLIENT-SECRET nohup bbr deployment \
--target BOSH-TARGET --username BOSH-CLIENT --deployment DEPLOYMENT-NAME \
--ca-cert PATH-TO-BOSH-SERVER-CERT \
backup --with-manifest [--artifact-path]
```

Where:

- ◊ `BOSH-CLIENT-SECRET` is your BOSH client secret. If you do not know your BOSH Client Secret, open your BOSH Director tile, navigate to **Credentials > Bosh Commandline Credentials** and record the value for `BOSH_CLIENT_SECRET`.
- ◊ `BOSH-TARGET` is your BOSH Environment setting. If you do not know your BOSH Environment setting, open your BOSH Director tile, navigate to **Credentials > Bosh Commandline Credentials** and record the value for `BOSH_ENVIRONMENT`. You must be able to reach the target address from the workstation where you run `bbr` commands.
- ◊ `BOSH-CLIENT` is your BOSH Client Name. If you do not know your BOSH Client Name, open your BOSH Director tile, navigate to **Credentials > Bosh Commandline Credentials** and record the value for `BOSH_CLIENT`.
- ◊ `DEPLOYMENT-NAME` is the Tanzu Kubernetes Grid Integrated Edition BOSH deployment name that you located in the **Locate the Tanzu Kubernetes Grid Integrated Edition Deployment Name** section above.
- ◊ `PATH-TO-BOSH-CA-CERT` is the path to the root CA certificate that you downloaded in **Download the Root CA Certificate** above.
- ◊ `--with-manifest` is necessary in order to redeploy your TKGI Control Plane in the case of its loss. `--with-manifest` is an optional `backup` parameter to include the manifest in the backup artifact.
- ◊ `--artifact-path` is an optional `backup` parameter to specify the output path for the backup artifact.

For example:

```
$ BOSH_CLIENT_SECRET=p455w0rd nohup bbr deployment
```

```
-target bosh.example.com -username admin -deployment cf-acceptance-0
-ca-cert bosh.ca.cert
backup -with-manifest
```



**Note:** The `-with-manifest` flag is necessary in order to redeploy your TKGI Control Plane in the case of its loss. The backup artifact created by this process contains credentials that you should keep secret.

2. Wait for the BBR backup command to complete:

- ◊ If the command completes successfully, follow the steps in [Manage Your Backup Artifact](#) below.
- ◊ If the backup command fails, perform the following actions:
  1. Run the command again, adding the `--debug` flag to enable debug logs. For more information, see [BBR Logging](#).
  2. Follow the steps in [Recover from a Failing Command](#).



**Note:** The BBR backup command can take a long time to complete. You can run it independently of the SSH session so that the process can continue running even if your connection to the jumpbox fails. The command above uses `nohup`, but you can run the command in a `screen` or `tmux` session instead.

## Cancel a Backup

Backups can take a long time. If you realize that the backup is going to fail or that your developers need to push an app immediately, you might need to cancel the backup.

To cancel a backup, perform the following steps:

1. Terminate the BBR process by pressing Ctrl-C and typing `yes` to confirm.
2. Because stopping a backup can leave the system in an unusable state and prevent additional backups, follow the procedures in [Clean up After a Failed Backup](#) below.

## After Backing Up Tanzu Kubernetes Grid Integrated Edition

After the backup has completed you should review and manage the generated backup artifacts.

### Manage Your Backup Artifact

The BBR-created backup consists of a directory containing the backup artifacts and metadata files. BBR stores each completed backup directory within the current working directory.



**Note:** The optional `-with-manifest` flag directs BBR to create a backup containing credentials. You should manage the generated backup artifact knowing it contains

secrets for administering your environment.

BBR backup artifact directories are named using the following formats:

- `DIRECTOR-IP-TIMESTAMP` for the BOSH Director backups.
- `DEPLOYMENT-TIMESTAMP` for the Control Plane backup.
- `DEPLOYMENT-TIMESTAMP` for the cluster deployment backups.

Keep your backup artifacts safe by following these steps:

1. Move the backup artifacts off the jumpbox to your storage space.
2. Compress and encrypt the backup artifacts when storing them.
3. Make redundant copies of your backup and store them in multiple locations. This minimizes the risk of losing your backups in the event of a disaster.
4. Each time you redeploy Tanzu Kubernetes Grid Integrated Edition, test your backup artifact by following the procedures in:
  - [Restore the Tanzu Kubernetes Grid Integrated Edition BOSH Director](#)
  - [Restore the Tanzu Kubernetes Grid Integrated Edition Control Plane](#)
  - [Restore Tanzu Kubernetes Grid Integrated Edition Clusters](#)

## Recover from a Failing Command

If the backup fails, follow these steps:

1. Ensure that you set all the parameters in the backup command.
2. Ensure the credentials previously obtained are valid.
3. Ensure the deployment that you specify in the BBR command exists.
4. Ensure that the jumpbox can reach the BOSH Director.
5. Consult [BBR Logging](#).
6. If you see the error message: `Directory /var/vcap/store/bbr-backup already exists on instance`, run the appropriate cleanup command. See [Clean Up After a Failed Backup](#) below for more information.
7. If the backup artifact is corrupted, discard the failing artifacts and run the backup again.

## Clean Up after a Failed Backup

If your backup process fails, use the BBR cleanup script to clean up the failed run.



**Warning:** It is important to run the BBR cleanup script after a failed BBR backup run. A failed backup run might leave the BBR backup directory on the instance, causing any subsequent attempts to backup to fail. In addition, BBR might not have run the post-backup scripts, leaving the instance in a locked state.

- If the TKGI BOSH Director backup failed:

- Run the following BBR cleanup script command to clean up:

```
bbr director --host BOSH-DIRECTOR-IP \
--username bbr --private-key-path PRIVATE-KEY-FILE \
backup-cleanup
```

Where:

- BOSH-DIRECTOR-IP** is the address of the BOSH Director. If the BOSH Director is public, **BOSH-DIRECTOR-IP** is a URL, such as <https://my-bosh.xxx.cf-app.com>. Otherwise, this is the internal IP **BOSH-DIRECTOR-IP** which you can retrieve as shown in [Retrieve the BOSH Director Address](#) above.
- PRIVATE-KEY-FILE** is the path to the private key file that you can create from [Bbr Ssh Credentials](#) as shown in [Download the BBR SSH Credentials](#) above. Replace the placeholder text using the information in the following table.

For example:

```
$ bbr director -host 10.0.0.5 -username bbr
-priate-key-path private-key.pem
backup-cleanup
```

- If the TKGI control plane or TKGI clusters backups fail:

- Run the following BBR cleanup script command to clean up:

```
BOSH_CLIENT_SECRET=BOSH-CLIENT-SECRET \
bbr deployment \
--target BOSH-TARGET \
--username BOSH-CLIENT \
--deployment DEPLOYMENT-NAME \
--ca-cert PATH-TO-BOSH-CA-CERT \
backup-cleanup
```

Where:

- BOSH-CLIENT-SECRET** is your BOSH client secret. If you do not know your BOSH Client Secret, open your BOSH Director tile, navigate to [Credentials > Bosch Commandline Credentials](#) and record the value for **BOSH\_CLIENT\_SECRET**.
- BOSH-TARGET** is your BOSH Environment setting. If you do not know your BOSH Environment setting, open your BOSH Director tile, navigate to [Credentials > Bosch Commandline Credentials](#) and record the value for **BOSH\_ENVIRONMENT**. You must be able to reach the target address from the workstation where you run **bbr** commands.
- BOSH-CLIENT** is your BOSH Client Name. If you do not know your BOSH Client Name, open your BOSH Director tile, navigate to [Credentials > Bosch Commandline Credentials](#) and record the value for **BOSH\_CLIENT**.
- DEPLOYMENT-NAME** is the Tanzu Kubernetes Grid Integrated Edition BOSH

deployment name that you located in the [Locate the Tanzu Kubernetes Grid Integrated Edition Deployment Names](#) section above.

- `PATH-TO-BOSH-CA-CERT` is the path to the root CA certificate that you downloaded in [Download the Root CA Certificate](#) above.

For example:

```
$ BOSH_CLIENT_SECRET=p455w0rd bbr deployment
-target bosh.example.com -username admin -deployment cf-acceptance-0
-ca-cert bosh.ca.crt
backup-cleanup
```

2. If the cleanup script fails, consult the following table to match the exit codes to an error message:

| V | Error                                                                                                                                                                                                                                    |
|---|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| a |                                                                                                                                                                                                                                          |
| l |                                                                                                                                                                                                                                          |
| u |                                                                                                                                                                                                                                          |
| e |                                                                                                                                                                                                                                          |
| 0 | Success                                                                                                                                                                                                                                  |
| 1 | General failure                                                                                                                                                                                                                          |
| 8 | The post-backup unlock failed. One of your deployments might be in a bad state and require attention.                                                                                                                                    |
| 1 | The cleanup failed. This is a non-fatal error indicating that the utility has been unable to clean up open BOSH SSH connections to a deployment's VMs. Manual cleanup might be required to clear any hanging BOSH users and connections. |
| 6 |                                                                                                                                                                                                                                          |

## Restoring TKGI Management Plane Components

This topic describes how to use BOSH Backup and Restore (BBR) to restore the BOSH Director, VMware Tanzu Kubernetes Grid Integrated Edition control plane.

### Overview

In the event of a disaster, you might lose your environment's VMs, disks, and your IaaS network and load balancer resources as well. You can re-create your environment, configured with your saved Tanzu Kubernetes Grid Integrated Edition Ops Manager Installation settings, using your BBR backup artifacts.

Before restoring using BBR:

- Review the requirements listed in [Compatibility of Restore](#) below.
- Complete all of the steps documented in [Preparing to Restore a Backup](#) below.

Use BBR to restore the following:

- The BOSH Director plane, see [Restore the BOSH Director](#) below.
- The Tanzu Kubernetes Grid Integrated Edition control plane, see [Restore Tanzu Kubernetes Grid Integrated Edition Control Plane](#) below.

## Compatibility of Restore

The following are the requirements for a backup artifact to be restorable to another environment:

- **Topology:** BBR requires the BOSH topology of a deployment to be the same in the restore environment as it was in the backup environment.
- **Naming of instance groups and jobs:** For any deployment that implements the backup and restore scripts, the instance groups and jobs must have the same names.
- **Number of instance groups and jobs:** For instance groups and jobs that have backup and restore scripts, the same number of instances must exist.

Additional considerations:

- **Limited validation:** BBR puts the backed up data into the corresponding instance groups and jobs in the restored environment, but cannot validate the restore beyond that.
- **Same Cluster:** Currently, BBR supports the in-place restore of a cluster backup artifact onto the same cluster. Migration from one cluster to another using a BBR backup artifact has not yet been validated.



**Note:** This section is for guidance only. You should always validate your backups by using the backup artifacts in a restore.

## Prepare to Restore a Backup

Before you use BBR to either back up TKGI or restore TKGI from backup, follow these steps to retrieve deployment information and credentials:

- [Verify your BBR Version](#)
- [Retrieve the BBR SSH Credentials](#)
- [Retrieve the BOSH Director Credentials](#)
- [Retrieve the UAA Client Credentials](#)
- [Retrieve the BOSH Director Address](#)
- [Download the Root CA Certificate](#)
- [Retrieve the BOSH Command Line Credentials](#)
- [Retrieve Your Cluster Deployment Names](#)

## Verify Your BBR Version

Before running BBR, verify that the installed version of BBR is compatible with the version of Ops Manager your TKGI tile is on:

1. To determine the Ops Manager BBR version requirements, see the [Ops Manager Release](#)

Notes for the version of Ops Manager you are using.

2. To verify the currently installed BBR version, run the following command:

```
bbr version
```

3. If the installed BBR version does not meet the Ops Manager BBR version requirement, or BBR is not installed, you must upgrade BBR. For more information, see [Installing BOSH Backup and Restore](#).

## Retrieve the BBR SSH Credentials

There are two ways to retrieve BOSH Director credentials:

- [Ops Manager Installation Dashboard](#)
- [Ops Manager API](#)

### Ops Manager Installation Dashboard

To retrieve your BBR SSH Credentials using the Ops Manager Installation Dashboard:

1. Navigate to the Ops Manager Installation Dashboard.
2. Click the BOSH Director tile.
3. Click the **Credentials** tab.
4. Locate **Bbr Ssh Credentials**.
5. Click **Link to Credentials** next to it.
6. Copy the `private_key_pem` field value.

### Ops Manager API

To retrieve your BBR SSH Credentials using the Ops Manager API:

1. Obtain your UAA access token. For more information, see [Access the Ops Manager API](#).
2. Retrieve the **Bbr Ssh Credentials** by running the following command:

```
curl "https://OPS-MAN-FQDN/api/v0/deployed/director/credentials/bbr_ssh_credentials" \
-X GET \
-H "Authorization: Bearer UAA-ACCESS-TOKEN"
```

Where:

- ◊ `OPS-MAN-FQDN` is the fully-qualified domain name (FQDN) for your Ops Manager deployment.
- ◊ `UAA-ACCESS-TOKEN` is your UAA access token.

3. Copy the value of the `private_key_pem` field.

### Save the BBR SSH Credentials to File

To save the BBR SSH credentials to a private key file:

1. To reformat the copied `private_key_pem` value and save it to a file in the current directory:

```
printf -- "YOUR-PRIVATE-KEY" > PRIVATE-KEY-FILE
```

Where:

- `YOUR-PRIVATE-KEY` is the text of your private key.
- `PRIVATE-KEY-FILE` is the path to the private key file you are creating.

For example:

```
$ printf - "--begin rsa private key-- fake key contents --end rsa private key--" > bbr_key.pem
```

## Retrieve the BOSH Director Credentials

There are two ways to retrieve BOSH Director credentials:

- [Ops Manager Installation Dashboard](#)
- [Ops Manager API](#)

### Ops Manager Installation Dashboard

To retrieve your BOSH Director credentials using the Ops Manager Installation Dashboard, perform the following steps:

1. Navigate to the Ops Manager Installation Dashboard.
2. Click the BOSH Director tile.
3. Click the **Credentials** tab.
4. Locate **Director Credentials**.
5. Click **Link to Credentials** next to it.
6. Copy and record the value of the `password` field.

### Ops Manager API

To retrieve your BOSH Director credentials using the Ops Manager API, perform the following steps:

1. Obtain your UAA access token. For more information, see [Access the Ops Manager API](#).
2. Retrieve the **Director Credentials** by running the following command:

```
curl "https://OPS-MAN-FQDN/api/v0/deployed/director/credentials/bbr_ssh_credentials" \
-X GET \
-H "Authorization: Bearer UAA-ACCESS-TOKEN"
```

Where:

- `OPS-MAN-FQDN` is the fully-qualified domain name (FQDN) for your Ops Manager deployment.

- ◊ **UAA-ACCESS-TOKEN** is your UAA access token.
3. Copy and record the value of the **password** field.

## Retrieve the UAA Client Credentials

To obtain BOSH credentials for your BBR operations, perform the following steps:

1. From the Ops Manager Installation Dashboard, click the **Tanzu Kubernetes Grid Integrated Edition** tile.
2. Select the **Credentials** tab.
3. Navigate to **Credentials > UAA Client Credentials**.
4. Record the value for **uaa\_client\_secret**.
5. Record the value for **uaa\_client\_name**.



**Note:** You must use BOSH credentials that limit the scope of BBR activity to your cluster deployments.

## Retrieve the BOSH Director Address

You access the BOSH Director using an IP address.

To obtain your BOSH Director's IP address:

1. Open the Ops Manager Installation Dashboard.
2. Select **BOSH Director > Status**.
3. Select the listed Director IP Address.

### Log In To BOSH Director

1. If you are not using the Ops Manager VM as your jumpbox, install the latest [BOSH CLI](#) on your jumpbox.
2. To log in to BOSH Director, using the IP address that you recorded above, run the following command line:

```
bosh -e BOSH-DIRECTOR-IP \
--ca-cert PATH-TO-BOSH-SERVER-CERTIFICATE log-in
```

Where:

- ◊ **BOSH-DIRECTOR-IP** is the BOSH Director IP address recorded above.
  - ◊ **PATH-TO-BOSH-SERVER-CERTIFICATE** is the path to the root Certificate Authority (CA) certificate as outlined in [Download the Root CA Certificate](#).
3. To specify **Email**, specify **director**.
  4. To specify **Password**, enter the **Director Credentials** that you obtained in [Retrieve the BOSH Director Credentials](#).

For example:

```
$ bosh -e 10.0.0.3
-ca-cert /var/tempest/workspaces/default/root_ca_certificate log-in Email (): d
irector Password (): **** Successfully authenticated with UAA Su
cceded
```

## Download the Root CA Certificate

To download the root CA certificate for your Tanzu Kubernetes Grid Integrated Edition deployment, perform the following steps:

1. Open the Ops Manager Installation Dashboard.
2. In the top right corner, click your username.
3. Navigate to **Settings > Advanced**.
4. Click **Download Root CA Cert**.

## Retrieve the BOSH Command Line Credentials

1. Open the Ops Manager Installation Dashboard.
2. Click the **BOSH Director** tile.
3. In the BOSH Director tile, click the **Credentials** tab.
4. Navigate to **Bosh Commandline Credentials**.
5. Click **Link to Credential**.
6. Copy the credential value.

## Retrieve Your Cluster Deployment Names

To locate and record a cluster deployment name, follow the steps below for each cluster:

1. On the command line, run the following command to log in:

```
tkgi login -a TKGI-API -u USERNAME -k
```

Where:

- ◊ **TKGI-API** is the domain name for the TKGI API that you entered in **Ops Manager > Tanzu Kubernetes Grid Integrated Edition > TKGI API > API Hostname (FQDN)**. For example, [api.tkgi.example.com](http://api.tkgi.example.com).
- ◊ **USERNAME** is your user name.

See [Logging in to Tanzu Kubernetes Grid Integrated Edition](#) for more information about the `tkgi login` command.



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information

about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

- Identify the cluster ID:

```
tkgi cluster CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of your cluster.

- From the output of this command, record the **UUID** value.
- Open the Ops Manager Installation Dashboard.
- Click the **BOSH Director** tile.
- Select the **Credentials** tab.
- Navigate to **Bosh Commandline Credentials** and click **Link to Credential**.
- Copy the credential value.
- SSH into your jumpbox. For more information about the jumpbox, see [Installing BOSH Backup and Restore](#).
- To retrieve your cluster deployment name, run the following command:

```
BOSH-CLI-CREDENTIALS deployments | grep UUID
```

Where:

- `BOSH-CLI-CREDENTIALS` is the full value that you copied from the BOSH Director tile in [Retrieve the BOSH Command Line Credentials](#).
- `UUID` is the cluster UUID that you recorded in the previous step.

## Transfer Artifacts to Your Jumpbox

To restore BOSH director, Tanzu Kubernetes Grid Integrated Edition control plane or cluster you must transfer your BBR backup artifacts from your safe storage location to your jumpbox.

- To copy an artifact onto a jumpbox, run the following SCP command:

```
scp -r LOCAL-PATH-TO-BACKUP-ARTIFACT JUMPBOX-USER@JUMPBOX-ADDRESS:
```

Where:

- `LOCAL-PATH-TO-BACKUP-ARTIFACT` is the path to your BBR backup artifact.
- `JUMPBOX-USER` is the ssh username of the jumpbox.
- `JUMPBOX-ADDRESS` is the IP address, or hostname, of the jumpbox.

- (Optional) Decrypt your backup artifact if the artifact is encrypted.

## Restore the BOSH Director

In the event of losing your BOSH Director or Ops Manager environment, you must first recreate the BOSH Director VM before restoring the BOSH Director.

You can restore your BOSH Director configuration by using Tanzu Kubernetes Grid Integrated Edition Ops Manager to restore the installation settings artifacts saved when following the [Export Installation Settings](#) backup procedure steps.

To redeploy and restore your Ops Manager and BOSH Director follow the procedures below.

## Deploy Ops Manager

In the event of a disaster, you might lose your IaaS resources. You must recreate your IaaS resources before restoring using your BBR artifacts.

1. To recreate your IaaS resources, such as networks and load balancers, prepare your environment for Tanzu Kubernetes Grid Integrated Edition by following the installation instructions specific to your IaaS in [Installing Tanzu Kubernetes Grid Integrated Edition](#).
2. After recreating IaaS resources, you must add those resources to Ops Manager by performing the procedures in the [\(Optional\) Configure Ops Manager for New Resources](#) section.

## Import Installation Settings



**WARNING:** After importing installation settings, do not click **Apply Changes** in Ops Manager before instructed to in the steps [Deploy the BOSH Director](#) or [Redeploy the Tanzu Kubernetes Grid Integrated Edition Control Plane](#).

You can import installation settings in two ways:

- Use the Ops Manager UI:
  1. Access your new Ops Manager by navigating to `YOUR-OPS-MAN-FQDN` in a browser.
  2. On the **Welcome to Ops Manager** page, click **Import Existing Installation**.
  3. In the import panel, perform the following tasks:
    - Enter the **Decryption Passphrase** in use when you exported the installation settings from Ops Manager.
    - Click **Choose File** and browse to the installation zip file that you exported in [Back Up Installation Settings](#).
  4. Click **Import**.



**Note:** Some browsers do not provide the import process progress status, and might appear to hang. The import process takes at least 10 minutes, and requires additional time for each restored Ops Manager tile.

5. **Successfully imported installation** is displayed upon successful completion of importing all installation settings.

- Use the Ops Manager API:
  1. To use the Ops Manager API to import installation settings, run the following command:

```
curl "https://OPS-MAN-FQDN/api/v1/installation_asset_collection" \
-X POST \
-H "Authorization: Bearer UAA-ACCESS-TOKEN" \
-F 'installation[file]=@installation.zip' \
-F 'passphrase=DECRYPTION-PASSPHRASE'
```

Where:

- **OPS-MAN-FQDN** is the fully-qualified domain name (FQDN) for your Ops Manager deployment.
- **UAA-ACCESS-TOKEN** is the UAA access token. For more information about how to retrieve this token, see [Using the Ops Manager API](#).
- **DECRYPTION-PASSPHRASE** is the decryption passphrase in use when you exported the installation settings from Ops Manager.

## (Optional) Configure Ops Manager for New Resources

If you recreated IaaS resources such as networks and load balancers by following the steps in the [Deploy Ops Manager](#) section above, perform the following steps to update Ops Manager with your new resources:

1. Activate Ops Manager advanced mode. For more information, see [How to Enable Advanced Mode in the Ops Manager](#) in the Knowledge Base.



**Note:** Ops Manager advanced mode allows you to make changes that are normally deactivated. You might see warning messages when you save changes.

2. Navigate to the Ops Manager Installation Dashboard and click the BOSH Director tile.
3. If you are using Google Cloud Platform (GCP), click **Google Config** and update:
  1. **Project ID** to reflect the GCP project ID.
  2. **Default Deployment Tag** to reflect the environment name.
  3. **AuthJSON** to reflect the service account.
4. Click **Create Networks** and update the network names to reflect the network names for the new environment.
5. If your BOSH Director had an external hostname, you must change it in **Director Config > Director Hostname** to ensure it does not conflict with the hostname of the backed up Director.
6. Ensure that there are no outstanding warning messages in the BOSH Director tile, then deactivate Ops Manager advanced mode. For more information, see [How to Enable Advanced Mode in the Ops Manager](#) in the Knowledge Base.



**Note:** A change in VM size or underlying hardware should not affect the ability for BBR restore data, as long as adequate storage space to restore the data exists.

## Remove BOSH State File

1. SSH into your Ops Manager VM. For more information, see the [Log in to the Ops Manager VM with SSH](#) section of the *Advanced Troubleshooting with the BOSH CLI* topic.
2. To delete the `/var/tempest/workspaces/default/deployments/bosh-state.json` file, run the following on the Ops Manager VM:

```
sudo rm /var/tempest/workspaces/default/deployments/bosh-state.json
```

3. In a browser, navigate to your Ops Manager's fully-qualified domain name.
4. Log in to Ops Manager.

## Deploy the BOSH Director

You can deploy the BOSH Director by itself in two ways:

- Use the Ops Manager UI:
  1. Open the Ops Manager Installation Dashboard.
  2. Click **Review Pending Changes**.
  3. On the Review Pending Changes page, click the **BOSH Director** checkbox.
  4. Click **Apply Changes**.
- Use the Ops Manager API:
  1. Use the Ops Manager API to deploy the BOSH Director.

## Restore the BOSH Director

Restore the BOSH Director by running BBR commands on your jumpbox.

To restore the BOSH Director:

1. Ensure the Tanzu Kubernetes Grid Integrated Edition BOSH Director backup artifact is in the folder from which you run BBR.
2. Run the BBR restore command to restore the TKG I BOSH Director:

```
nohup bbr director --host BOSH-DIRECTOR-IP \
--username bbr --private-key-path PRIVATE-KEY-FILE \
restore \
--artifact-path PATH-TO-DIRECTOR-BACKUP
```

Where:

- ◊ `BOSH-DIRECTOR-IP` is the address of the BOSH Director. If the BOSH Director is public, `BOSH-DIRECTOR-IP` is a URL, such as <https://my-bosh.xxx.cf-app.com>. Otherwise, this is the internal IP `BOSH-DIRECTOR-IP` which you can retrieve as shown in [Retrieve the BOSH Director Address](#).

- ❖ `PRIVATE-KEY-FILE` is the path to the private key file that you can create from [Bbr Ssh Credentials](#) as shown in [Download the BBR SSH Credentials](#).
- ❖ `PATH-TO-DEPLOYMENT-BACKUP` is the path to the TKGI BOSH Director backup that you want to restore.

For example:

```
$ nohup bbr director -host 10.0.0.5
-username bbr -private-key-path private.pem
restore
-artifact-path /home/10.0.0.5-abcd1234abcd1234
```



**Note:** The BBR restore command can take a long time to complete. The example command in this section uses `nohup` and the restore process is run within your SSH session. If you instead run the BBR command in a `screen` or `tmux` session the task will run separately from your SSH session and will continue to run, even if your SSH connection to the jumpbox fails.

### 3. If your BOSH Director restore fails, do one or more of the following:

- ❖ Run the command again, adding the `--debug` flag to activate debug logs. For more information, see [BBR Logging](#).
- ❖ Follow the steps in [Resolve a Failing BBR Restore Command](#) below.

Be sure to complete the steps in [Clean Up After a Failed Restore](#) below.

## Remove All Stale Deployment Cloud IDs

After BOSH Director has been restored, you must reconcile BOSH Director's internal state with the state of the IaaS.

### 1. To determine the existing deployments in your environment, run the following command:

```
BOSH-CLI-CREDENTIALS bosh deployments
```

Where `BOSH-CLI-CREDENTIALS` is the full `Bosh Commandline Credentials` value that you copied from the BOSH Director tile in [Download the BOSH Commandline Credentials](#).

### 2. To reconcile the BOSH Director's internal state with the state of a single deployment, run the following command:

```
BOSH-CLI-CREDENTIALS bosh -d DEPLOYMENT-NAME -n cck \
--resolution delete_disk_reference \
--resolution delete_vm_reference
```

Where:

- ❖ `BOSH-CLI-CREDENTIALS` is the full `Bosh Commandline Credentials` value that you copied from the BOSH Director tile in [Download the BOSH Commandline](#)

### Credentials.

- ◊ `DEPLOYMENT-NAME` is a deployment name retrieved in the previous step.
3. Repeat the last command for each deployment in the IaaS.

## Restore the Tanzu Kubernetes Grid Integrated Edition Control Plane

You must redeploy the Tanzu Kubernetes Grid Integrated Edition tile before restoring the Tanzu Kubernetes Grid Integrated Edition control plane. By redeploying the Tanzu Kubernetes Grid Integrated Edition tile you create the VMs that constitute the control plane deployment.

To redeploy the Tanzu Kubernetes Grid Integrated Edition tile, do the following:

- [Determine the Required Stemcell](#) needed by the tile.
- Upload that stemcell as described in [Upload Stemcells](#).
- [Redeploy the Tanzu Kubernetes Grid Integrated Edition Control Plane](#).
- [Restore the TKGI Control Plane](#) from a BBR backup on top of the deployment.

### Determine the Required Stemcell

Do either of the following procedures to determine the stemcell that TKGI uses:

- Review the Stemcell Library:
  1. Open Ops Manager.
  2. Click **Stemcell Library**.
  3. Record the TKGI stemcell release number from the **Staged** column.
- Review a Stemcell List Using BOSH CLI:
  1. To retrieve the stemcell release using the BOSH CLI, run the following command:

```
BOSH-CLI-CREDENTIALS bosh deployments
```

Where `BOSH-CLI-CREDENTIALS` is the full `Bosh Commandline Credentials` value that you copied from the BOSH Director tile in [Download the BOSH Commandline Credentials](#).

For example:

```
$ bosh deployments

Using environment '10.0.0.5' as user 'director' (bosh.*.read, op
enid, bosh.*.admin, bosh.read, bosh.admin)

Name Release(s)
) Stemcell(s)
 Team(s)
pivotal-container-service-453f2faa3bd2e16f52b7 backup-an
```

```
d-restore-sdk/1.9.0 bosh-google-kvm-ubuntu-jammy-g
o_agent/1.75 -
```

...



**Note:** At most, the TKGI tile can have two stemcells, where one stemcell is Linux and the other stemcell is Windows.

For more information about stemcells in Ops Manager, see [Importing and Managing Stemcells](#).

## Upload Stemcells

To upload the stemcell used by your Tanzu Kubernetes Grid Integrated Edition tile:

1. Download the stemcell from [VMware Tanzu Network](#).
2. Run the following command to upload the stemcell used by TKGI:

```
BOSH-CLI-CREDENTIALS bosh -d DEPLOYMENT-NAME \
--ca-cert PATH-TO-BOSH-SERVER-CERTIFICATE \
upload-stemcell \
--fix PATH-TO-STEMCELL
```

Where:

- ◊ **BOSH-CLI-CREDENTIALS** is the full [Bosh Commandline Credentials](#) value that you copied from the BOSH Director tile in [Download the BOSH Commandline Credentials](#).
  - ◊ **PATH-TO-BOSH-SERVER-CERTIFICATE** is the path to the root CA certificate that you downloaded in [Download the Root CA Certificate](#).
  - ◊ **PATH-TO-STEMCELL** is the path to your tile's stemcell.
3. To ensure the stemcells for all of your other installed tiles have been uploaded, repeat the last step, running the `bosh upload-stemcell --fix PATH-TO-STEMCELL` command, for each required stemcell that is different from the already uploaded TKGI stemcell.

## Redeploy the Tanzu Kubernetes Grid Integrated Edition Control Plane

To redeploy your Tanzu Kubernetes Grid Integrated Edition tile's control plane:

1. From the Ops Manager Installation Dashboard, navigate to **VMware Tanzu Kubernetes Grid Integrated Edition > Resource Config**.
2. Ensure the **Upgrade all clusters** errand is **Off**.
3. Ensure that all other errands needed by your system are set to run.
4. Return to the Ops Manager Installation Dashboard.
5. Click **Review Pending Changes**.
6. Review your changes. For more information, see [Reviewing Pending Product Changes](#).
7. Click **Apply Changes** to redeploy the control plane.

## Restore the TKGI Control Plane

Restore the Tanzu Kubernetes Grid Integrated Edition control plane by running BBR commands on your jumpbox.

To restore the Tanzu Kubernetes Grid Integrated Edition control plane:

1. Ensure the Tanzu Kubernetes Grid Integrated Edition deployment backup artifact is in the folder from which you run BBR.
2. Run the BBR restore command to restore the TKGI control plane:

```
BOSH_CLIENT_SECRET=BOSH-CLIENT-SECRET \
nohup bbr deployment --target BOSH-TARGET \
--username BOSH-CLIENT --deployment DEPLOYMENT-NAME \
--ca-cert PATH-TO-BOSH-SERVER-CERT \
restore \
--artifact-path PATH-TO-DEPLOYMENT-BACKUP
```

Where:

- ◊ **BOSH-CLIENT-SECRET** is the value for **BOSH\_CLIENT\_SECRET** retrieved in [Download the BOSH Commandline Credentials](#).
- ◊ **BOSH-TARGET** is the value for **BOSH\_ENVIRONMENT** retrieved in [Download the BOSH Commandline Credentials](#). You must be able to reach the target address from the workstation where you run **bbr** commands.
- ◊ **BOSH-CLIENT** is the value for **BOSH\_CLIENT** retrieved in [Download the BOSH Commandline Credentials](#).
- ◊ **DEPLOYMENT-NAME** is the deployment name retrieved in [Locate the Tanzu Kubernetes Grid Integrated Edition Deployment Name](#).
- ◊ **PATH-TO-BOSH-CA-CERT** is the path to the root CA certificate that you downloaded in [Download the Root CA Certificate](#).
- ◊ **PATH-TO-DEPLOYMENT-BACKUP** is the path to the TKGI control plane backup that you want to restore.

For example:

```
$ BOSH_CLIENT_SECRET=p455w0rd

nohup bbr deployment -target bosh.example.com

-username admin -deployment pivotal-container-0

-ca-cert bosh.ca.crt

restore

-artifact-path /home/pivotal-container-service_abcd1234abcd1234abcd-ab
cd1234abcd1234
```



**Note:** The BBR restore command can take a long time to complete. The command above uses `nohup` and the restore process is run within your SSH session. If you instead run the BBR command in a `screen` or `tmux` session the task will run separately from your SSH session and will continue to run, even if your SSH connection to the jumpbox fails.

3. If your Tanzu Kubernetes Grid Integrated Edition control plane restore fails, do one or more of the following:

- Run the command again, adding the `--debug` flag to activate debug logs. For more information, see [BBR Logging](#).
- Follow the steps in [Resolve a Failing BBR Restore Command](#) below.

Be sure to complete the steps in [Clean Up After a Failed Restore](#) below.

## Resolve a Failing BBR Restore Command

To resolve a failing BBR restore command:

1. Ensure that you set all the parameters in the command.
2. Ensure that the BOSH Director credentials are valid.
3. Ensure that the specified BOSH deployment or Director exists.
4. Ensure that the jumpbox can reach the BOSH Director.
5. Ensure the source backup artifact is compatible with the target BOSH deployment or Director.
6. If you see the error message `Directory /var/vcap/store/bbr-backup already exists on instance`, run the relevant commands from the [Clean up After Failed Restore](#) section of this topic.
7. See the [BBR Logging](#) topic.

## Cancel a Restore

If you must cancel a restore, perform the following steps:

1. Terminate the BBR process by pressing Ctrl-C and typing `yes` to confirm.
2. Perform the procedures in the [Clean up After Failed Restore](#) section to support future restores. Stopping a restore can leave the system in an unusable state and prevent future restores.

## Clean Up After a Failed Restore

If a BBR restore process fails, BBR might not have run the post-restore scripts, potentially leaving the instance in a locked state. Additionally, the BBR restore folder might remain on the target instance and subsequent restore attempts might also fail.

- To resolve issues following a failed BOSH Director restore, run the following BBR command:

```
nohup bbr director \
--host BOSH-DIRECTOR-IP \
--username bbr \
--private-key-path PRIVATE-KEY-FILE \
restore-cleanup
```

Where:

- **BOSH-DIRECTOR-IP** is the address of the BOSH Director. If the BOSH Director is public, BOSH-DIRECTOR-IP is a URL, such as <https://my-bosh.xxx.cf-app.com>. Otherwise, this is the internal IP **BOSH-DIRECTOR-IP** which you can retrieve as shown in [Retrieve the BOSH Director Address](#) above.
- **PRIVATE-KEY-FILE** is the path to the private key file that you can create from [Bbr Ssh Credentials](#) as shown in [Download the BBR SSH Credentials](#) above.

For example:

```
$ nohup bbr director

-target 10.0.0.5

-username bbr

-private-key-path private.pem

restore-cleanup
```

- To resolve issues following a failed control plane restore, run the following BBR command:

```
BOSH_CLIENT_SECRET=BOSH-CLIENT-SECRET \
bbr deployment \
--target BOSH-TARGET \
--username BOSH-CLIENT \
--deployment DEPLOYMENT-NAME \
--ca-cert PATH-TO-BOSH-CA-CERT \
restore-cleanup
```

Where:

- **BOSH-CLIENT-SECRET** is the value for **BOSH\_CLIENT\_SECRET** retrieved in [Download the BOSH Commandline Credentials](#) above.
- **BOSH-TARGET** is the value for **BOSH\_ENVIRONMENT** retrieved in [Download the BOSH Commandline Credentials](#) above. You must be able to reach the target address from the workstation where you run **bbr** commands.
- **BOSH-CLIENT** is the value for **BOSH\_CLIENT** retrieved in [Download the BOSH Commandline Credentials](#) above.
- **DEPLOYMENT-NAME** is the name retrieved in [Retrieve Your Cluster Deployment Name](#) above.
- **PATH-TO-BOSH-CA-CERT** is the path to the root CA certificate that you downloaded in [Download the Root CA Certificate](#) above.

For example:

```
$ BOSH_CLIENT_SECRET=p455w0rd
bbr deployment
-target bosh.example.com
-username admin
-deployment pivotal-container-service-453f2f
-ca-cert bosh.ca.crt
restore-cleanup
```

- To resolve issues following a failed cluster restore, run the following BBR command:

```
BOSH_CLIENT_SECRET=BOSH-CLIENT-SECRET \
bbr deployment \
--target BOSH-TARGET \
--username BOSH-CLIENT \
--deployment DEPLOYMENT-NAME \
--ca-cert PATH-TO-BOSH-CA-CERT \
restore-cleanup
```

Where:

- ◊ **BOSH-CLIENT-SECRET** is the value for `BOSH_CLIENT_SECRET` retrieved in [Download the BOSH Commandline Credentials](#).
- ◊ **BOSH-TARGET** is the value for `BOSH_ENVIRONMENT` retrieved in [Download the BOSH Commandline Credentials](#). You must be able to reach the target address from the workstation where you run `bbr` commands.
- ◊ **BOSH-CLIENT** is the value for `BOSH_CLIENT` retrieved in [Download the BOSH Commandline Credentials](#).
- ◊ **DEPLOYMENT-NAME** is the name retrieved in [Retrieve Your Cluster Deployment Names](#) above.
- ◊ **PATH-TO-BOSH-CA-CERT** is the path to the root CA certificate that you downloaded in [Download the Root CA Certificate](#).

For example:

```
$ BOSH_CLIENT_SECRET=p455w0rd
bbr deployment
-target bosh.example.com
-username admin
```

```
-deployment pivotal-container-service-453f2f
-ca-cert bosh.ca.crt
restore-cleanup
```

## Backing Up and Restoring the Data Center for TKGI

This section describes how to backup and restore the TKGI infrastructure.

### Overview

Backup and restoring the TKGI infrastructure includes the following components:

- NSX-T Data Center. See [Backing Up and Restoring NSX-T Manager](#).
- vCenter Server. See [Backing Up and Restoring the vCenter Server](#).



**Note:** In terms of TKGI, the primary focus is the backup and restore of NSX-T Data Center. vCenter backup and restore is outside the scope of this documentation, but a mention of it is included to ensure it is part of your end-to-end backup and restore workflow.

### Test Considerations

As part of your TKGI backup and restore planning and testing, consider the following test scenarios for NSX-T.

- Take backup of NSX-T.
- Shut down (or delete) NSX Manager nodes.
- Deploy new NSX-T Manager node and restore configuration from backup.
- Restore additional NSX Manager nodes.
- If you deploy a Kubernetes application with a service endpoint, and it is included in the NSX-T backup, on restore of NSX-T there is no loss in application connectivity. However, the application must be redeployed if it was deployed after the backup was taken.
- If you create a Kubernetes namespace, and it is included in the NSX-T backup, on restore of NSX-T, the namespace is present. However, the namespace must be recreated if it was created after the backup was taken. If you delete a namespace when NSX-T is down, on restore of NSX-T the namespace is removed.
- If you create a Kubernetes cluster, and it is included in the NSX-T backup, on restore of NSX-T the cluster is reachable. However, if the cluster was created after the NSX-T backup was taken, the cluster is not reachable. In this case, delete cluster and create new one.
- If you create a Kubernetes namespace, and take an NSX-T backup, then deploy pods, on restore of NSX-T, restarting pods brings them to running state since the namespace was created before backup. However, if the namespace was not created before the NSX-T backup, the pods are stuck in ContainerCreating state. Both the namespace and the pods

must be recreated.

- If you create a load balancer service before taking a backup of NSX-T, on restore the service is present. However, if you create the service after taking a backup of NSX-T, on restore of NSX-T the load balancer VIP is pending. You must redeploy the service for it to work. If you delete a load balancer service in-between taking a back up of NSX-T and restoring NSX-T, on restore the load balancer VIP is deleted.

## Backing Up and Restoring VMware NSX Manager

This topic describes how to back up and restore NSX-T Data Center for TKGI.

### NSX-T Data Center Backup and Recover

NSX-T Data Center provides in-product backup and recovery that supports backup and restore of the NSX Manager Nodes. For more information, see [Backing Up and Restoring NSX Manager](#) in the NSX-T documentation.

### Deployment Assumptions

To backup and restore NSX-T Data Center, it is assumed that 3 NSX Manager Nodes are deployed, and there is an HA VIP configured for access to the NSX Management Plane. In addition, there are at least 2 Edge Nodes deployed with an HA VIP for the Edge Nodes.

For more information, refer the to the [NSX-T for TKGI installation instructions](#).

### Backup Procedure

Create a backup of the NSX-T Manager Nodes as follows:

1. Log in to the NSX Manager web console.
2. Navigate to **System > Backup & Restore**.
3. Select **Edit** and configure the backup location for the NSX Configuration. For more information, refer to [Configure Backups](#) in the NSX-T Data Center documentation.

## Backup Configuration

|                                                                                                                                       |       |
|---------------------------------------------------------------------------------------------------------------------------------------|-------|
| FQDN or IP Address *                                                                                                                  |       |
| Protocol                                                                                                                              | SFTP  |
| Port *                                                                                                                                | 22    |
| Directory Path *                                                                                                                      |       |
| Username *                                                                                                                            | admin |
| Password                                                                                                                              | ***** |
| SSH Fingerprint                                                                                                                       |       |
| <small>Please provide only SHA-256 fingerprint of ECDSA key. Example:<br/>SHA256:vu1HIG6qIsWbRfPnPognmtK+mMQ3sfU+ablLxvFk58QI</small> |       |
| <small>You need to use the same passphrase to restore from the backup</small>                                                         |       |
| Passphrase                                                                                                                            | ***** |
| Confirm Passphrase                                                                                                                    | ***** |
| <input type="button" value="CANCEL"/> <input type="button" value="SAVE"/>                                                             |       |

- Click **Start Backup** to begin the backup of the NSX Manager database.

## Restore Procedure

To restore NSX-T Data Center, you restore the configuration using the backup and start sending traffic. See [Restore a Backup](#) in the NSX-T Data Center documentation.



**Note:** Configuration changes made between backup and restore will not be saved.

## Testing Procedure

The following test scenario assumes TKGI is installed on vSphere with NSX-T 3.0, and that a full backup of NSX-T Manager has been performed. This scenario tests the restoration of NSX-T.

- Verify NSX-T connectivity by testing access to a deployed Kubernetes application that is fronted by a service of type LoadBalancer. This verifies that the NSX-T load balancer is functioning correctly.
- Shut down all 3 NSX Manager VMs, and delete them.
- Deploy a new NSX Manager node. For more information, refer to the [NSX-T for TKGI installation documentation](#).
- Restore the NSX Manager configuration from the backup. See [See Restore a Backup](#) in the NSX-T documentation.
- Add 2 additional Managers.

## Backing Up and Restoring the vCenter Server

This topic describes how to back up and restore the vCenter Server in context of a TKGI deployment.



**Note:** Backup and restore of the vSphere SDDC is typically included as part of a site-wide disaster recovery plan. The information provided here is to be used as part of confirmation of such a plan.

## vCenter Cluster Configuration

To support highly available clusters, configure the vCenter clusters for TKGI with HA and DRS enabled. For more information, see [vSphere Availability](#) in the VMware vSphere documentation.

## vCenter Server Backup and Recover

vCenter Server supports file backup to network attached storage.

To backup and restore vCenter, create a backup of the vCenter primary server. Refer to the topic [File-Based Backup and Restore of vCenter Server](#) in the vCenter documentation.

## Data Protection

VMware provides a robust set of vSphere Storage APIs for host data protection, and partners with third-party vendors to provide backup and recovery solutions for vSphere datastores, including the following:

- [Dell Avamar](#)
- [Veeam](#)
- [Datrium](#)

# Tanzu Kubernetes Grid Integrated Edition Security

This section includes the following security topics for Tanzu Kubernetes Grid Integrated Edition (TKGI):

- **Security Disclosure and Release Process:** See [Tanzu Kubernetes Grid Integrated Edition Security Disclosure and Release Process](#)
- **Certificates:** See [Tanzu Kubernetes Grid Integrated Edition Certificates](#)
- **Benchmarks:** See [TKGI Cluster Benchmarks](#), below.

## CIS Kubernetes Benchmarks

For security compliance assessments, you can use [Compliance Scanner for VMware Tanzu](#) to benchmark TKGI clusters against the Center for Internet Security [CIS Kubernetes Benchmark v1.6.0](#):

1. Follow the procedure in [Installing and Configuring Compliance Scanner](#), and enable the following in the **Scan Configuration** pane, under **Benchmarks**:
  - ◊ **TKGI Master Node - Level 1 and Level 2**
  - ◊ **TKGI Worker Node - Level 1 and Level 2**

## Tanzu Kubernetes Grid Integrated Edition Security Disclosure and Release Process

This topic describes the processes for disclosing security issues and releasing related fixes for VMware Tanzu Kubernetes Grid Integrated Edition, Kubernetes, VMware NSX, and VMware Harbor.

## Security Issues in Tanzu Kubernetes Grid Integrated Edition

VMware provides security coverage for Tanzu Kubernetes Grid Integrated Edition. Please report any vulnerabilities directly to the [VMware Security Response Center](#).

Security fixes are provided in accordance with the [Ops Manager Security Overview and Policy](#).

Where applicable, security issues might be coordinated with the responsible disclosure process for the open source security teams in Kubernetes and Cloud Foundry projects.

## Security Issues in Kubernetes

VMware follows the Kubernetes responsible disclosure process to work within the Kubernetes project to report and address suspected security issues with Kubernetes.

This process is discussed in [Kubernetes Security and Disclosure Information](#).

When the Kubernetes project releases security fixes, Tanzu Kubernetes Grid Integrated Edition releases fixes according to the [Ops Manager Security Overview and Policy](#).

## Security Issues from CFF

VMware follows the Cloud Foundry Foundation (CFF) responsible disclosure process to report and address suspected security issues.

This process is discussed in [Cloud Foundry Security](#).

When the Cloud Foundry Foundation releases security fixes, Tanzu Kubernetes Grid Integrated Edition releases fixes according to the [Ops Manager Security Overview and Policy](#).

## Security Issues in VMware NSX

Security issues in VMware NSX are coordinated with the [VMware Security Response Center](#).

## Security Issues in VMware Harbor

Security issues in VMware Harbor are coordinated with the [VMware Security Response Center](#).

## Tanzu Kubernetes Grid Integrated Edition Certificates

This topic summarizes Tanzu Kubernetes Grid Integrated Edition (TKGI) certificates and how to rotate them.

### Overview of TKGI Certificates

TKGI secures all communication between TKGI control plane components and TKGI-managed Kubernetes clusters using Transport Layer Security (TLS) validated by RSA Certificate Authority (CA) certificates and leaf certificates that they issue:

- **TKGI Control Plane Certificates**

TKGI control plane certificates secure TKGI control plane component communication with the TKGI API server and TKGI DB server.

- **TKGI User-Configurable Certificates**

TKGI user-configurable TKGI certificates must be configured by TKGI administrators.

- **BOSH Certificates used by TKGI**

The BOSH certificates used by TKGI, are generated by BOSH.

- **TKGI-Provisioned Kubernetes Cluster Certificates**

The TKGI-Provisioned Kubernetes Cluster certificates, and their leaf certificates, used by Kubernetes clusters are unique to each Kubernetes cluster and are automatically generated by TKGI. TKGI manages the life-cycle of the per-cluster CA and the certificates it signs.

- **NSX-T-Only TKGI Kubernetes Cluster Certificates**

NSX-T-only TKGI Kubernetes cluster certificates must be registered with NSX

Manager.

The certificates used by TKGI automatically expire and must be rotated. For more information, see [Check Certificate Expiration Dates](#) and [Rotating Certificates](#) below.

For more information about certificates, see [Certificate Types](#) in the Ops Manager documentation.

## Check Certificate Expiration Dates

Before rotating TKGI certificates you should determine which certificates are approaching expiration.

To review certificate expiration dates:

- Review certificate expiration dates using Ops Manager. To see certificate expiration dates, open the Ops Manager and select the **Certificates** tab.
- Use CredHub to export the certificate expiration dates for the TKGI deployment or one or more clusters. For more information, see [How to get the expiry dates of all CA's & certificates in the PKS deployment and clusters](#) in the VMware Tanzu Knowledge Base.
- Use the TKGI CLI to export the certificate expiration dates for Kubernetes cluster certificates. For more information, see [List TLS Certificates](#) in *Rotate Kubernetes Cluster Certificates*.

## Rotating Certificates

VMware Tanzu Kubernetes Grid Integrated Edition administrators occasionally verify the expiration dates of their TKGI certificates, and when needed, rotate expiring certificates. See [Check Certificate Expiration Dates](#) above to review your TKGI certificate expiration dates.



**Warning:** Never use the CredHub Maestro `maestro regenerate ca/leaf -all` command to rotate TKGI certificates.

The following summarizes the rotation requirements for the certificates used in TKGI environments:

- **TKGI Control Plane Certificates**

| Certificate                                                      | Default Rotation | Rotation Description                                                                                                                                                        |
|------------------------------------------------------------------|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>pxc_server_ca</code> and leaf certificates                 | Fours years      | See <a href="#">Rotate TKGI Control Plane Certificates</a> or <a href="#">How to rotate TKGI control plane CA and leaf certificates</a> in the VMware Tanzu Knowledge Base. |
| <code>pxc_galera_ca</code> and leaf certificates                 | Years            | See <a href="#">Rotate TKGI Control Plane Certificates</a> or <a href="#">How to rotate TKGI control plane CA and leaf certificates</a> in the VMware Tanzu Knowledge Base. |
| <code>uaa_active_pkcs_saml_key_2018</code> and leaf certificates | Years            | See <a href="#">Rotate TKGI Control Plane Certificates</a> or <a href="#">How to rotate TKGI control plane CA and leaf certificates</a> in the VMware Tanzu Knowledge Base. |
| <code>kubo_odb_ca_2018</code> and leaf certificates              | Years            | See <a href="#">Rotate TKGI Control Plane Certificates</a> or <a href="#">How to rotate TKGI control plane CA and leaf certificates</a> in the VMware Tanzu Knowledge Base. |

- **TKGI User-Configurable TKGI Certificates**

| Certificate                              | Default Expiration                                | Rotation Description                                                                                                                                                                                                                                                                                                                                                                              |
|------------------------------------------|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>pks_tls</code>                     | A<br>d<br>m<br>in<br>-<br>d<br>ef<br>in<br>e<br>d | Open the TKGI API tab on the TKGI tile.<br>The TKGI API Service certificate is used to secure access to the TKGI API endpoint.                                                                                                                                                                                                                                                                    |
| <code>nsx-t-superuser-certificate</code> | A<br>d<br>m<br>in<br>-<br>d<br>ef<br>in<br>e<br>d | See <a href="#">Rotate the Principal Identity Certificate and Key</a> in <i>Generating and Registering the NSX Manager Superuser Principal Identity Certificate and Key</i> or <a href="#">How to renew the nsx-t-superuser-certificate used by Principal Identity user</a> in the VMware Tanzu Knowledge Base.<br>The NSX-T SuperUser certificate is used to secure access to the NSX-T manager. |

- **BOSH Certificates used by TKGI**

| Certificate                             | Default Expiration | Rotation Description                                                                                                   |
|-----------------------------------------|--------------------|------------------------------------------------------------------------------------------------------------------------|
| <code>bosh_dns_health_client_tls</code> | One year           | See <a href="#">How to rotate bosh-dns certificates in TKGI 1.9+ using maestro</a> in the VMware Tanzu Knowledge Base. |
| <code>bosh_dns_health_server_tls</code> |                    |                                                                                                                        |
| <code>dns_api_client_tls</code>         |                    |                                                                                                                        |
| <code>dns_api_server_tls</code>         |                    |                                                                                                                        |

To extend the default expiration period, see [Overriding Duration for Certificates](#) in the Ops Manager documentation.

- **TKGI Kubernetes Cluster Certificates**

| Certificate and leaf certificates                                                                                                                                                           | Default Expiration | Rotation Description                                                          |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|-------------------------------------------------------------------------------|
| <code>kubo_master_ca_2021</code>                                                                                                                                                            | Four years         | See Rotate Kubernetes Cluster Certificates.                                   |
| and leaf certificates:<br><code>tls-kubernetes-2018,</code><br><code>tls-ncp-2018 (with NSX-T),</code><br><code>tls-nsx-kube-proxy-2018 (with NSX-T)</code>                                 |                    |                                                                               |
| <code>kubo_ca_2018</code>                                                                                                                                                                   |                    |                                                                               |
| and leaf certificates:<br><code>tls-kubelet-2018,</code><br><code>tls-metrics-server-2018,</code><br><code>tls-kubelet-client-2018,</code><br><code>tls-kube-controller-manager-2018</code> |                    |                                                                               |
| <code>etcd_ca_2018</code>                                                                                                                                                                   |                    |                                                                               |
| and leaf certificates:<br><code>tls-etcd-2018-2,</code><br><code>tls-etcdctl-2018-2,</code><br><code>tls-etcdctl-root-2018-2,</code><br><code>tls-etcdctl-flanneld-2018-2</code>            |                    |                                                                               |
| <b>* NSX-T-Only TKGI Kubernetes Cluster Certificates</b>                                                                                                                                    |                    |                                                                               |
| Certificate                                                                                                                                                                                 | Default Expiration | Rotation Description                                                          |
| <code>tls-nsx-t</code>                                                                                                                                                                      | Two years          | See Rotate NSX-T Certificates Only in Rotate Kubernetes Cluster Certificates. |
| <code>tls-nsx-lb</code>                                                                                                                                                                     | Five years         | The NSX-T only certificates must be registered with NSX Manager.              |

## Rotating Tanzu Kubernetes Grid Integrated Edition Control Plane Certificates

This topic describes how to rotate certificates used only by the Tanzu Kubernetes Grid Integrated Edition (TKGI) control plane and tile.

This topic covers rotating TKGI control plane certificates only. For more information about TKGI Certificates:

- For conceptual information about certificates in TKGI, see [TKGI Certificates](#).
- To rotate the certificates used by TKGI-deployed Kubernetes clusters, see [Rotating Cluster Certificates](#).



**Warning:** If you use the TKGI Management Console to manage TKGI on vSphere with NSX-T, you must use the Management Console to rotate the NSX Manager CA Certificate. To manage your NSX Manager CA Certificate using the TKGI

Management Console, see [Which Options Can I Reconfigure?](#) in *Reconfigure Your Tanzu Kubernetes Grid Integrated Edition Deployment*.

## Overview

TKGI control plane certificates, and their leaf certificates, are automatically generated by TKGI during installation:

- `pxc_server_ca`
- `pxc_galera_ca`
- `uaa_active_pkcs_saml_key_2018`
- `kubo_odb_ca_2018`

Control plane certificates have a default expiration period of four years.

To rotate TKGI control plane certificates, first determine which certificates are due to expire and then rotate them:

- [Check Certificate Expiration Dates](#)
- [Rotate TKGI Control Plane Certificates](#)

The procedures below can be used to rotate TKGI control plane certificates, certificates for TKGI communication with underlying Ops Manager and BOSH infrastructure, and certificates for components such as database, CredHub, UAA, and Telemetry.



**Warning:** Never use the CredHub Maestro `maestro regenerate ca/leaf -all` command to rotate TKGI certificates.

## Check Certificate Expiration Dates

Before rotating your certificates, verify which certificates require rotation.

To check certificate expiration dates, see [Check Expiration Dates and Certificate Types](#) in the Ops Manager documentation.

## Rotate TKGI Control Plane Certificates

TKGI control plane and tile certificates are configurable and non-configurable certificates stored in CredHub. For an explanation of configurable, non-configurable, and other certificate types, see [Certificate Types](#) in the Ops Manager documentation.

Rotate configurable and non-configurable certificates as follows:

- Rotate **configurable** certificates in the tile interface by entering new values and redeploying the tile.
  - **All infrastructures:** The `pks_tls` cert is configured in the TKGI tile > **TKGI API** pane.
  - **vSphere with NSX-T:** After rotating the following two configurable certificates, you must also re-register them with the NSX Manager. For instructions, see [Rotate the](#)

[Principal Identity Certificate and Key](#) in *Generating and Registering the NSX Manager Superuser Principal Identity Certificate and Key*:

- **NSX Manager Super User Principal Identity Certificate**, `nsx-t-superuser-certificate`
- **NSX Manager CA Cert**, `nsx-t-ca-cert`
- **Non-configurable** certificates rotate automatically with selected tile upgrades. Most of these certificates have four- or five-year expiry periods, so users do not ordinarily need to rotate them.
  - To rotate **Non-configurable** certificates, use the Rotate Control Plane Certificates Tool. For more information, see [How to rotate TKGI control plane CA and leaf certificates](#) in the VMware Tanzu Knowledge Base.

## Rotate Kubernetes Cluster Certificates

This topic describes how to rotate certificates used by Tanzu Kubernetes Grid Integrated Edition (TKGI) Kubernetes clusters.

For more information about TKGI Certificates:

- For conceptual information about certificates in TKGI, see [TKGI Certificates](#).
- To rotate the certificates used by the TKGI control plane, see [Rotating TKGI Control Plane Certificates](#).

## Overview

When TKGI provisions a Kubernetes cluster, the system generates certificate authority (CA) certificates and leaf certificates that have values and expiration dates unique to that cluster.

The following table summarizes the TKGI-provisioned Kubernetes cluster certificates and how to rotate them.

| Certificates                                                                                                           | When Used                                                           | How to Rotate                                                           |
|------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------|-------------------------------------------------------------------------|
| <code>kubo_master_ca_2021</code> , <code>kubo_ca_2018</code> , <code>etcd_ca_2018</code> , and their leaf certificates | All clusters.                                                       | <a href="#">See Rotate Kubernetes Cluster Certificates below</a> .      |
| <code>tls_nsx_t</code> and <code>tls_nsx_lb</code>                                                                     | NSX-T only. These certificates must be registered with NSX Manager. | <a href="#">See Rotate NSX-T Certificates for Kubernetes Clusters</a> . |

For more information about Kubernetes Cluster certificates in TKGI, see [TKGI Certificates](#).



**Warning:** Never use the CredHub Maestro `maestro regenerate ca/leaf -all` command to rotate TKGI certificates.

## Procedure

To rotate TKGI-provisioned Kubernetes cluster certificates, first determine which certificates are due to expire and then rotate them:

- [List TLS Certificates](#)

- Rotate TLS Certificates

## List TLS Certificates

To list the TLS certificates used by TKGI-provisioned Kubernetes cluster, run the following command:

```
tkgi certificates CLUSTER-NAME -d DAYS
```

Where:

- **CLUSTER-NAME** is the name of the cluster.
- **DAYS** is the maximum number of days remaining until the certificate expires.

For example:

```
tkgi certificates tkgi-cluster-01 -d 10000
```

The sample output lists all TLS certificates that TKGI uses for the specified cluster.

| NAME                                                                                           | Type | Days Left | Valid until          |
|------------------------------------------------------------------------------------------------|------|-----------|----------------------|
| /p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-nsx-lb                       | Leaf | 1803      | 2025-12-14T06:47:46Z |
| /p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-nsx-kube-proxy-2018          | Leaf | 1439      | 2024-12-15T06:47:41Z |
| /p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-ncp-2018                     | Leaf | 1439      | 2024-12-15T06:47:41Z |
| /p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-nsx-t                        | Leaf | 708       | 2022-12-15T06:47:40Z |
| /p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-kube-controller-manager-2018 | Leaf | 1439      | 2024-12-15T06:47:40Z |
| /p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-metrics-server-2018          | Leaf | 1439      | 2024-12-15T06:47:39Z |
| /p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-etcdctl-flanneld-2018-2      | Leaf | 1439      | 2024-12-15T06:47:39Z |
| /p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-etcdctl-root-2018-2          | Leaf | 1439      | 2024-12-15T06:47:38Z |
| /p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-etcdctl-2018-2               | Leaf | 1439      | 2024-12-15T06:47:37Z |
| /p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-etcd-2018-2                  | Leaf | 1439      | 2024-12-15T06:47:36Z |
| /p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-kubelet-client-2018          | Leaf | 1439      | 2024-12-15T06:47:36Z |
| /p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-kubelet-2018                 | Leaf | 1439      | 2024-12-15T06:47:35Z |
| /p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/etcd_ca_2018                     | Root | 1439      | 2024-12-15T06:47:35Z |
| /p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-kubernetes-2018              | Leaf | 1439      | 2024-12-15T06:47:34Z |
| /p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/kubo_ca_2018                     | Root | 1439      | 2024-12-15T06:47:34Z |

## Rotate TLS Certificates

The TKGI CLI supports rotating TLS certificates for the following scenarios:

- Rotate All Cluster Certificates
- Rotate All Cluster Certificates Except NSX-T
- Rotate NSX-T Certificates Only
- Rotate Custom CA

For more information about how to use TKGI CLI to rotate Kubernetes cluster TLS certificates, see [Rotate TLS Certificates Using the TKGI CLI](#) below.

## Rotate All Cluster Certificates

To rotate all cluster certificates:

```
tkgi rotate-certificates CLUSTER-NAME --all
```

This command rotates [all certificates](#) except a custom CA `kubo_master_ca_2021` (if implemented).

## Rotate All Cluster Certificates Except NSX-T

To rotate all cluster certificates except the NSX-T certificates:

```
tkgi rotate-certificates CLUSTER-NAME --skip-nsx --all
```

This command rotates [all certificates](#) except `tls-nsx-t` and `tls-nsx-lb`.

## Rotate NSX-T Certificates Only

To rotate only NSX certificates:

```
tkgi rotate-certificates CLUSTER-NAME --only-nsx
```

This command only rotates the NSX-T certificates `tls-nsx-t` and `tls-nsx-lb`.

For example:

```
tkgi rotate-certs tkgi-cluster-01 --only-nsx

You are about to rotate nsx related certificates for cluster tkgi-cluster-01. This operation requires bosh deployment, and will take a significant time. Are you sure you want to continue? (y/n) :
```

For more information, see [Rotate NSX-T Certificates for Kubernetes Clusters](#).

## Rotate Custom CA

If you have implemented a custom CA for the `kubo_master_ca_2021`, rotation is handled by the `update-cluster` CLI command.

To rotate a custom `kubo_master_ca_2021` CA:

1. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a](#)

Public Cloud CSI Driver in *Release Notes* for additional requirements.

2. Run the following command:

```
tkgi update-cluster CLUSTER-NAME --config-file CONFIG-FILENAME
```

Where:

- ◊ **CLUSTER-NAME** is the name of the cluster.
- ◊ **CONFIG-FILENAME** is the name of the configuration file.

For complete usage, see [Use a Custom CA for Kubernetes Clusters](#).

## Rotate TLS Certificates Using the TKGI CLI

You can use the TKGI CLI to list and rotate the TLS certificates created for a Kubernetes cluster.

Usage:

```
tkgi rotate-certs | rotate-certificates CLUSTER-NAME [flags]
```

Flags:

|                   |                                                                              |
|-------------------|------------------------------------------------------------------------------|
| --all             | Rotate all certs, not implemented yet, will be available in future releases. |
| -h, --help        | help for rotate-certs                                                        |
| --json            | Return the PKS-API output as json                                            |
| --non-interactive | Don't ask for user input                                                     |
| --only-nsx        | Rotate the tls-nsx-lb and tls-nsx-t certificates.                            |
| --wait            | Wait for the operation to finish                                             |

## Rotate VMware NSX Certificates for Kubernetes Clusters

This topic describes how to list and rotate TLS certificates for Kubernetes clusters provisioned by Tanzu Kubernetes Grid Integrated Edition.

## About NSX-T Certificate Rotation for Kubernetes Clusters Provisioned by TKGI

You can use the TKGI CLI to rotate the certificates for the NSX-T load balancer and the certificate for the NSX-T Principal Identity for the NSX Manager for the specified Kubernetes cluster provisioned by TKGI. To rotate other cluster certificates using the TKGI CLI, see [Rotate Kubernetes Cluster Certificates](#).



**WARNING:** During NSX-T TLS certificate rotation, the system will update the Principal Identity certificate to access the NSX Manager API (for the specified TKGI cluster instance). The rotation process will impact network related operations (for the specified TKGI cluster instance), but there should be no impact for existing cluster workloads.

## List TLS Certificates Created for NSX-T

To list the TLS certificates created for a TKGI-provisioned Kubernetes cluster:

- Run the following:

```
tkgi certificates CLUSTER-NAME -d EXPIRATION-WINDOW
```

Where:

- CLUSTER-NAME** is the name of your cluster.
- EXPIRATION-WINDOW** is the expiration range for the listed certificates. The listed certificates will expire within the designated number of days.

The returned list of certificates includes TLS certificates used by TKGI for the specified cluster and the certificates named `tls-nsx-lb` and `tls-nsx-t`, which are used for NSX-T.

For example:

```
tkgi certificates tkgi-cluster-01 -d 10000

NAME
 Type Days Left Valid until

/p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-nsx-
lb Leaf 1803 2025-12-14T06:47:46Z

/p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-nsx-
kube-proxy-2018 Leaf 1439 2024-12-15T06:47:41Z

/p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-ncp-
2018 Leaf 1439 2024-12-15T06:47:41Z

/p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-nsx-
t Leaf 708 2022-12-15T06:47:40Z

/p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-kube-
-controller-manager-2018 Leaf 1439 2024-12-15T06:47:40Z

/p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-metr
ics-server-2018 Leaf 1439 2024-12-15T06:47:39Z

/p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-etcd
ctl-flanneld-2018-2 Leaf 1439 2024-12-15T06:47:39Z

/p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-etcd
ctl-root-2018-2 Leaf 1439 2024-12-15T06:47:38Z

/p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-etcd
ctl-2018-2 Leaf 1439 2024-12-15T06:47:37Z
```

```
/p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-etcd
-2018-2 Leaf 1439 2024-12-15T06:47:36Z

/p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-kube
let-client-2018 Leaf 1439 2024-12-15T06:47:36Z

/p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-kube
let-2018 Leaf 1439 2024-12-15T06:47:35Z

/p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/etcd_ca_
2018 Root 1439 2024-12-15T06:47:35Z

/p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/tls-kube
rnetes-2018 Leaf 1439 2024-12-15T06:47:34Z

/p-bosh/service-instance_62e2a43a-dc2a-47a8-a361-3911589e60aa/kubo_ca_
2018 Root 1439 2024-12-15T06:47:34Z
```

## Rotate the TLS Certificates for NSX-T

To rotate the TLS certificates for NSX-T:

1. To skip SSL verification during the certificate rotation, you must first deactivate SSL verification on the TKGI tile. For more information, see the [Disable SSL certification verification](#) configuration instructions in [Networking](#).
2. Run the following:

```
tkgi rotate-certs | rotate-certificates CLUSTER-NAME [flags]
```

Where `CLUSTER-NAME` is the name of your cluster.

Flags:

|                   |                                                                              |
|-------------------|------------------------------------------------------------------------------|
| --all             | Rotate all certs, not implemented yet, will be available in future releases. |
| -h, --help        | help for rotate-certs                                                        |
| --json            | Return the PKS-API output as json                                            |
| --non-interactive | Don't ask for user input                                                     |
| --only-nsx        | Rotate the tls-nsx-lb and tls-nsx-t certificates.                            |
| --wait            | Wait for the operation to finish                                             |

For example:

```
tkgi rotate-certs tkgi-cluster-01 --only-nsx
```

You are about to rotate nsx related certificates for cluster tkgi-cluster-01. This operation requires bosh deployment, and will take a significant time. Are you sure you want to continue? (y/n):

3. If running `tkgi rotate-certs` fails to rotate the certificates, you must manually rotate the

certificates. To manually rotate certificates, see [How to rotate Tanzu Kubernetes Grid Integrated Edition tls-nsx-t cluster certificate](#) in the VMware Tanzu Knowledge Base.

For more information, see [Rotate Kubernetes Cluster Certificates](#).

## Use a Custom CA for Kubernetes Clusters

This topic describes how to use a custom certificate authority (CA) to secure your Kubernetes clusters provisioned by Tanzu Kubernetes Grid Integrated Edition.

### Custom CA Support

By default TKGI creates a new per-cluster CA (`kubo_master_ca_2021`) for each Kubernetes cluster. TKGI manages the lifecycle of the per-cluster CA and the certificates it signs. For more information, see [Overview of Kubernetes Cluster Certificates](#).

For most use cases, the system-managed per-cluster CA is appropriate. If it is required, you can apply a custom CA to a Kubernetes cluster, either during [creation](#) or [update](#).

Configuring a Kubernetes cluster with a custom CA is an advanced operation. If you use a custom CA, you are responsible for managing its lifecycle, including monitoring its expiry and [rotating](#) it while still valid.

To use custom CA functionality, existing clusters must be upgraded to support per-cluster CA.



**Warning:** Using a custom CA is an advanced operation. It is recommended that you contact [VMware Support](#) before proceeding.

### Create Cluster with Custom CA

To apply a custom CA to a new cluster, specify the custom CA using the `--config-file` option with the `create-cluster` operation.

For example:

```
tkgi create-cluster <clustername> --external-hostname <clustername.example.com> --plan
'Plan 1' --config-file custom_ca.yaml
```

See [custom CA requirements and formats](#).

### Update Cluster with Custom CA

To apply a custom CA to an existing cluster:

1. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
2. Run the following command:

```
tkgi update-cluster CLUSTER-NAME --config-file CONFIG-FILE-NAME
```

Where:

- ◊ `CLUSTER-NAME` is the name of your cluster.
- ◊ `CONFIG-FILE-NAME` is the configuration file to use. For example, `custom_ca.yaml`.

See custom CA [requirements](#) and [formats](#).

## Rotate Custom CA

You can rotate a [custom CA](#) by updating the cluster with the new CA.



**Note:** This procedure is designed to work if the custom CA you are replacing has not expired.

To rotate the custom CA of an existing cluster:

1. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
2. Run the following command:

```
tkgi update-cluster CLUSTER-NAME --config-file CONFIG-FILE-NAME
```

Where:

- ◊ `CLUSTER-NAME` is the name of your cluster.
- ◊ `CONFIG-FILE-NAME` is the configuration file to use. For example, `custom_ca.yaml`.

See custom CA [requirements](#) and [formats](#).

## Troubleshoot Custom CA

The log is located at `/var/vcap/sys/log/pks-api.log`.

The following command will trigger a Bosh deployment. You can track the progress using the Bosh CLI:

```
bosh tasks
bosh task <task-id> --debug
```

After rotation of the custom CA, check the cluster status and certificate expiration date.

## Custom CA Requirements

The custom CA must include a certificate and private key in standard PEM format as follows:

```
{
 "custom_ca": [
 {
 "certificate": "certificate in PEM format",
 "private_key": "private key in PEM format"
 }
]
}
```

The custom CA must satisfy following validation rules:

- It must be a CA.
- It must be a root CA.
- It must not expire within 6 months.
- The private key and the certificate must match.

## Custom CA Formats

TKGI supports both YAML (`*.yml` or `*.yaml`) and JSON (`*.json`) configuration file format for supplying the custom CA.

Below is an example configuration file in YAML format for a custom CA (`custom_ca.yaml`):

```
YAML for custom CA with certificate and private key

custom_ca:
 certificate: |
 -----BEGIN CERTIFICATE-----
 MIIDdTCCAl2gAwIBAgIUAYwzsbEoRKrd+j2L74Noh3CDEEwDQYJKoZIhvcNAQEL
 ...
 ...
 BWlVhWvqBkTMTcNE83gGLPvRgTWqSqs+Q==
 -----END CERTIFICATE-----
 private_key: |
 -----BEGIN RSA PRIVATE KEY-----
 MIEowIBAAKCAQEAti6NiJny8D+dffZ2TXY4VWTyjyr/tFuVLr9XcwOhe9q/lQ6E
 ...
 vf3/DKcf9K/iHfdrfDDO/fVE/TO/4Gclqmc2ArwluWruXlqV2+6w
 -----END RSA PRIVATE KEY-----
```

Below is an example configuration file in JSON format for a custom CA (`custom_ca.json`):

```
{
 "custom_ca": {
 "certificate": "-----BEGIN CERTIFICATE-----\nMIIDdTCCAl2gAwIBAgIUAY...83TWqSqs+Q\n==\n-----END CERTIFICATE-----\n",
 "private_key": "-----BEGIN RSA PRIVATE KEY-----\nMIEowIBAAKCAQi6iJny...XlqV2+6w\n-----END RSA PRIVATE KEY-----\n"
 }
}
```

## Encrypt Secrets in an etcd Database

This topic describes how to create and use a Kubernetes profile to encrypt a cluster's etcd database.

For more information and other uses of Kubernetes profiles, see [Using Kubernetes Profiles](#).

## Create Kubernetes Profile

To create a new Kubernetes profile:

1. Create a new a base64-encoded, 32-byte random key string to use as a secret.

- On Linux or MacOS, you can generate the secret by running:

```
head -c 32 /dev/urandom | base64
```

This command returns a base64-encoded, 32-byte key string, for example:

```
jHc3NMp7s7T7JoJuZF7NUSkHVYCSikJCNJ+LrltbkJk=
```

2. To create an encryption provider configuration file, create a file named `encryption-provider-config.yaml` containing the following content:

```
apiVersion: apiserver.config.k8s.io/v1
kind: EncryptionConfiguration
resources:
- resources:
 - secrets
 providers:
 - aescbc:
 keys:
 - name: key1
 secret: BASE-64-ENCODED-SECRET
 - identity: {}
```

Where `BASE-64-ENCODED-SECRET` is the secret key string created in the last step.



**Note:** To read unencrypted secrets, include the `identity` provider as the last provider as shown above.

3. To create a Kubernetes profile configuration file that customizes the kube-apiserver using your encryption provider configuration file, create a JSON file containing the following content:

```
{
 "name": "PROFILE-TITLE",
 "description": "PROFILE-DESC",
 "customizations": [
 {
 "component": "kube-apiserver",
 "arguments": {},
 "file-arguments": {
 "encryption-provider-config": "/LOCAL-DIR/encryption-provider-config.yaml"
 }
 }
]
}
```

Where:

- `PROFILE-TITLE` is the name of your profile, for example `profile1`.
- `PROFILE-DESC` is the description you want to use for the profile.
- `LOCAL-DIR` is the directory containing your encryption provider configuration file.

- To create a Kubernetes profile based on your profile configuration file, use the TKGI CLI:

```
tkgi create-k8s-profile PROFILE-PATH
```

Where `PROFILE-PATH` is the path and filename of the JSON profile file you created in the step above.

For example:

```
$ tkgi create-k8s-profile /tmp/profile1.json Kubernetes profile profile1 successfully created
```

## Create Kubernetes Cluster

To create a Kubernetes cluster based on a Kubernetes profile:

- To create a cluster based on a Kubernetes profile, use the TKGI CLI:

```
tkgi create-cluster CLUSTER-NAME -e EXTERNAL-HOSTNAME -p small -n 1 --kubernetes-profile K8S-PROFILE
```

Where:

- `CLUSTER-NAME` is the name to apply to the new cluster.



**Note:** Use only lowercase characters when naming your cluster if you manage your clusters with Tanzu Mission Control (TMC). Clusters with names that include an uppercase character cannot be attached to TMC.

- `EXTERNAL-HOSTNAME` is the address to use to access Kubernetes API.
- `K8S-PROFILE` is the Kubernetes profile name. For more information, see `tkgi create-cluster` in *TKGI CLI*.

Running this command restarts your kube-apiserver with your encryption provider configuration file set as the `--encryption-provider-config` parameter.

For example:

```
$ tkgi create-cluster cluster1 -e cluster1-internal.com -p small -n 1 --kubernetes-profile profile1

TKGI Version: 1.9.0-build.1

Name: cluster1

K8s Version: 1.18.8

Plan Name: small
```

|                          |                                      |
|--------------------------|--------------------------------------|
| UUID:                    | 22f78823-0b70-4684-be2f-8457d6f3b1f1 |
| Last Action:             | CREATE                               |
| Last Action State:       | in progress                          |
| Last Action Description: | Creating cluster                     |
| Kubernetes Master Host:  | cluster1-internal.com                |
| Kubernetes Master Port:  | 8443                                 |
| Worker Nodes:            | 1                                    |
| Kubernetes Master IP(s): | In Progress                          |
| Network Profile Name:    |                                      |
| Kubernetes Profile Name: | profile1                             |
| Tags:                    |                                      |

## Ensure Existing Data is Encrypted

If you have existing data, the data has been stored without being encrypted.

To encrypt your existing data:

1. Run the following:

```
kubectl get secrets --all-namespaces -o json | kubectl replace -f -
```

This command reads all secrets, applies encryption, and saves the data encrypted. For more information, see [Ensure all secrets are encrypted](#) in the Kubernetes documentation.

2. Be sure to complete the steps in [Verify Your Data is Encrypted](#) below.

If you use a Key Management Service (KMS) provider, see [Using a KMS provider for data encryption](#), and [Encrypting Secret Data at Rest](#) in the Kubernetes documentation.

## Verify Your Data is Encrypted

After the kube-apiserver restarts, all existing data in its etcd database should be encrypted, and it should encrypt any new data that it stores.

To ensure the etcd data has been encrypted, create and store a test secret and then retrieve it:

1. To create and store a test secret:

- To use a basic `create secret` command:

```
kubectl create secret generic SECRET-NAME -n NAMESPACE --from-literal=KEY
-NAME=KEY-VALUE
```

Where:

- **SECRET-NAME** is the name to apply to the secret.
- **NAMESPACE** is the namespace.
- **KEY-NAME** is the name of the key.
- **KEY-VALUE** is the value to encrypt.

For example:

```
kubectl create secret generic secret1 -n default -from-literal=
mykey=mydata
```

- ◊ To create a secret for vSphere, use your **csi-vsphere.conf** configuration file that contains details for connecting to vSphere:

```
kubectl create secret generic SECRET-NAME --from-file=csi-vsphere.conf -
--namespace=SECRET-NAMESPACE
```

Where:

- **SECRET-NAME** is the name to apply to the secret.
- **SECRET-NAMESPACE** is the secret namespace, for example **velero**.

For example:

```
kubectl create secret generic secret1 -from-file=csi-vsphere.co
nf --namespace=velero
```

For more information, see [create secret generic](#) in the Kubernetes documentation.

2. To read the test secret out of etcd, use the etcdctl command line:

```
ETCDCTL_API=3 etcdctl get /registry/secrets/default/SECRET-NAME [ETCD-ARGS] | h
exdump -C
```

Where:

- ◊ **SECRET-NAME** is the name to apply to the secret.
- ◊ **ETCD-ARGS** are the additional arguments for connecting to the etcd server.

For example:

To get the secret and view it after you SSH into your control plane node with admin privileges:

```
ETCDCTL_API=3 /var/vcap/jobs/etcd/bin/etcdctl get /registry/secrets/d
efault/secret1
```

3. Review the output and verify the stored test secret is prefixed with **k8s:enc:aescbc:v1:**.

For example:

```
master/3f9c5ca9-a2b1-469c-9007-6fdd0844a5ec:/var/vcap/bosh_ssh/bosh_2
```

```
f4aaa514474422# ETCDCTL_API=3 /var/vcap/jobs/etcdbin/etcctl get /registry/secrets/default/secret1

/registry/secrets/default/secret1 k8s:enc:aescbc:v1:key1:◆◆◆◆@◆8◆
◆◆◆◆A◆◆◆◆◆◆◆◆2cM7◆◆◆◆◆◆◆uL◆/
```

The `k8s:enc:aescbc:v1:` prefix indicates the `aescbc` provider has encrypted the resulting data.

- To retrieve the stored test secret:

```
kubectl describe secret secret1 -n default
```

Where: `SECRET-NAME` is the name of the secret.

For example:

```
kubectl describe secret secret1 -n default
```

- Review the output and verify the secret is correctly decrypted. The `mydata` content is returned encoded. For information on how to decode `mydata`, see [decoding a secret](#).

## Rotate Encryption Key for Secrets in etcd Database

This section describes how to use `encryption provider config` to rotate a key. For highly-available deployments running multiple kube-apiserver processes, changing a secret without incurring downtime requires a multi-step process.

To rotate an encryption key for a secret in an etcd database:

- Generate a new key. For more information, see [Create Kubernetes Profile](#) above.
- Add the new key as the second key entry for the current provider in your `encryption-provider-config.yaml` `keys:` list.

For example:

```
apiVersion: apiserver.config.k8s.io/v1
kind: EncryptionConfiguration
resources:
 - resources:
 - secrets
 providers:
 - aescbc:
 keys:
 - name: key1
 secret: jHc3NMp7s7T7JoJuZF7NUSkHVYCSikJCNJ+LrltbkJk=
 - name: key2
 secret: MN17xeE48/1dH+pE5LLHblhId6Ajbzdn2I6rubh8AfE=
 identity: {}
```



**Note:** To read unencrypted secrets, include the `identity` provider as the last

provider, as shown above.

3. Create a Kubernetes profile configuration file that references the modified `encryption-provider-config.yml`.

For example:

```
{
 "name": "profile2",
 "description": "Testing profile two",
 "customizations": [
 {
 "component": "kube-apiserver",
 "arguments": { },
 "file-arguments": {
 "encryption-provider-config": "/tmp/encryption-provider-config.yml"
 }
 }
]
}
```

For information on how to create a Kubernetes profile configuration file, see [Create Kubernetes Profile](#) above.

4. Use the TKGI CLI to create a profile based on the configuration file.

For example:

```
$ tkgi create-k8s-profile /tmp/profile2.json
Kubernetes profile profile2 successfully created
```

For information on how to use the TKGI CLI to create a Kubernetes profile, see [Create Kubernetes Profile](#) above.

5. If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.
6. Update the cluster with this new profile.

```
tkgi update-cluster CLUSTER-NAME --kubernetes-profile PROFILE-TITLE
```

Where:

- ◊ `CLUSTER-NAME` is the name to apply to the new cluster.
- ◊ `PROFILE-TITLE` is the name of your profile, for example `profile2`.

For example:

```
$ tkgi update-cluster cluster1 --kubernetes-profile profile2
Update summary for cluster cluster1:
```

```
Kubernetes Profile Name: profile2

Are you sure you want to continue? (y/n): y

Use 'tkgi cluster cluster1' to monitor the state of your cluster
```

Running this command restarts your kube-apiserver with your encryption provider configuration file set as the `--encryption-provider-config` parameter. This ensures that each server can decrypt using the new key.

7. Edit the encryption provider configuration file so that it lists the new key as the first entry in the `keys:` property, swapping its position with the old key.

For example:

```
apiVersion: apiserver.config.k8s.io/v1
kind: EncryptionConfiguration
resources:
 - resources:
 - secrets
 providers:
 - aescbc:
 keys:
 - name: key2
 secret: MNI7xeE48/1dH+pE5LLHblhId6Ajbzdn2I6rubh8AfE=
 - name: key1
 secret: jHc3NMp7s7T7JoJuZF7NUSkHVYCSikJCNJ+LrltbkJk=
 - identity: {}
```

8. Create a Kubernetes profile configuration file that references the modified `encryption-provider-config.yml`.

For example:

```
{
 "name": "profile3",
 "description": "Testing profile three",
 "customizations": [
 {
 "component": "kube-apiserver",
 "arguments": { },
 "file-arguments": {
 "encryption-provider-config": "/tmp/encryption-provider-config.yml"
 }
 }
]
}
```

9. Use the TKGI CLI to create a profile based on the new Kubernetes profile configuration file.

For example:

```
$ tkgi create-k8s-profile /tmp/profile3.json
Kubernetes profile profile2 successfully created
```

10. Update the cluster with the new profile.

For example:

```
$ tkgi update-cluster cluster1 -kubernetes-profile profile3

Update summary for cluster cluster1:

Kubernetes Profile Name: profile3

Are you sure you want to continue? (y/n): y

Use 'tkgi cluster cluster1' to monitor the state of your cluster
```

This restarts the kube-apiserver processes to ensure that each server now encrypts using the new key.

11. To encrypt all existing secrets with the new key:

```
kubectl get secrets --all-namespaces -o json | kubectl replace -f -
```

12. Back up etcd, snapshotting it with the new key.
13. Edit the `encryption-provider-config.yml` again to remove the old decryption key from the config `keys`: list.

For example:

```
apiVersion: apiserver.config.k8s.io/v1
kind: EncryptionConfiguration
resources:
 - resources:
 - secrets
 providers:
 - aescbc:
 Keys:
 - name: key2
 secret: MNI7xeE48/1dH+pE5LLHblhId6AjbzdN2I6rubh8AfE=
 - identity: {}
```

14. Create a new Kubernetes profile configuration file that references the modified `encryption-provider-config.yml`.

For example:

```
{
```

```

"name": "profile4",
"description": "Testing profile four",
"customizations": [
 {
 "component": "kube-apiserver",
 "arguments": { },
 "file-arguments": {
 "encryption-provider-config": "/tmp/encryption-provider-config.yml"
 }
 }
]
}

```

For information on how to create a Kubernetes profile configuration file, see [Create Kubernetes Profile](#) above.

15. Use the TKGI CLI to create a profile based on the new Kubernetes profile configuration file.

For example:

```
$ tkgi create-k8s-profile /tmp/profile4.json

Kubernetes profile profile4 successfully created
```

16. Update the cluster with the new profile.

For example:

```
$ tkgi update-cluster cluster1 -kubernetes-profile profile4

Update summary for cluster cluster1:

Kubernetes Profile Name: profile4

Are you sure you want to continue? (y/n): y

Use 'tkgi cluster cluster1' to monitor the state of your cluster
```

This restarts the kube-apiserver processes to ensure that each server now encrypts using the new key.

17. Verify the stored secret is prefixed with `k8s:enc:aescbc:v1:key2` which indicates the `aescbc` provider has encrypted the resulting data.

For example:

```
master/3f9c5ca9-a2b1-469c-9007-6fdd0844a5ec:/var/vcap/bosh_ssh/bosh_2
f4aaa514474422#
ETCDCTL_API=3 /var/vcap/packages/etcctl/etcctl -endpoints
```

```

https://master-0.etcd.cfcr.internal:2379 -cert
/var/vcap/jobs/etcd/config/etcdctl.crt -key
/var/vcap/jobs/etcd/config/etcdctl.key -cacert
/var/vcap/jobs/etcd/config/etcdctl-ca.crt get
/registry/secrets/default/secret1

/registry/secrets/default/secret1

k8s:enc:aescbc:v1:key2:◆◆◆◆@◆8◆◆◆◆A◆◆◆◆◆◆◆2cM7◆◆◆◆◆uL◆
/

```

## Decrypt Secrets in the etcd Database

1. To deactivate encryption of data at rest, place the identity provider as the first entry in the encryption provider configuration file.

For example:

```

apiVersion: apiserver.config.k8s.io/v1
kind: EncryptionConfiguration
resources:
 - resources:
 - secrets
 providers:
 - identity: {}
 - aescbc:
 keys:
 - name: key1
 secret: <BASE 64 ENCODED SECRET>

```

2. Create a new Kubernetes profile configuration file that references the modified [encryption-provider-config.yml](#).

For example:

```
{
 "name": "profile5",
 "description": "Testing profile five",
 "customizations": [
 {
 "component": "kube-apiserver",
 "arguments": { },
 "file-arguments": {
 "encryption-provider-config": "/tmp/encryption-provider-config.yml"
 }
 }
}
```

```

 }
]
}
```

For information on how to create a Kubernetes profile configuration file, see [Create Kubernetes Profile](#) above.

3. Use the TKGI CLI to create a profile based on the modified Kubernetes profile configuration file.

For example:

```
$ tkgi create-k8s-profile /tmp/profile5.json
Kubernetes profile profile5 successfully created
```

4. Update the cluster with the new profile.

For example:

```
$ tkgi update-cluster cluster1 -kubernetes-profile profile5
Update summary for cluster cluster1:
Kubernetes Profile Name: profile5
Are you sure you want to continue? (y/n): y
Use 'tkgi cluster cluster1' to monitor the state of your cluster
```

For information on how to use `update-cluster`, see [Rotate Encryption Key for Secrets in etcd Database](#) above.

5. To force all secrets to be decrypted:

```
kubectl get secrets --all-namespaces -o json | kubectl replace -f -
```

## Securing Access to the AWS Metadata Service

This topic describes how to use a Kubernetes Network Policy to secure access to the AWS instance metadata service from Kubernetes clusters created with VMware Tanzu Kubernetes Grid Integrated Edition (TKGI).

### Overview

For Pods on TKGI clusters deployed on AWS, you can manage access to the AWS instance metadata service using a Kubernetes Network Policy:

- You can block app access to the instance metadata service for apps in either all cluster

namespaces or specific namespaces.

- You can grant app access to the instance metadata service for apps in one or more labeled Pods.

To manage access to the AWS instance metadata service, complete one of the following:

- [Secure Access to AWS Instance Metadata for a Namespace](#)
- [Secure Access for All Namespaces Using an Antrea Cluster-Wide Network Policy](#)

For information on why you should secure access to AWS instance metadata, see [Instance metadata and user data](#) in the AWS documentation.

## Secure Access to AWS Instance Metadata for a Namespace

You can use Kubernetes Network Policies to deny access to the AWS instance metadata service by default from all apps in a namespace and if desired, grant access to the service from specific Pods:

- [Deny Access from a Specific Namespace](#)
- [Grant Access to Specific Apps in a Namespace](#)

### Deny Access from a Specific Namespace

To use a Kubernetes Network Policy to deny access to AWS instance metadata by default from a specific namespace:

1. To create a `deny` Network Policy:

1. Create a YAML configuration file named `np.yml`.
2. Populate the YAML file with one of the following `deny` Network Policy configurations:

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
 name: POLICY-NAME
 namespace: NAMESPACE
 annotations:
 kubernetes.io/ingress.class: "nsx"
spec:
 podSelector: {}
 policyTypes:
 - Egress
 egress:
 - to:
 - ipBlock:
 cidr: 0.0.0.0/0
 except:
 - 169.254.169.254/32
```

Where:

- `POLICY-NAME` is the name for this Network Policy. For example `deny-metadata-access`.
- `NAMESPACE` is the name of the namespace to apply the Network Policy to. For

example, `default` to manage access from Pods in the `default` namespace.

3. Save the YAML configuration file.
2. To apply the `deny` Network Policy to your cluster:

```
kubectl apply -f np.yml
```

For example:

```
kubectl apply -f np.yml
```

```
networkpolicy.networking.k8s.io/deny-metadata-access created
```

3. Verify the Network Policy has been applied:

```
kubectl get networkpolicy
```

For example:

```
kubectl get networkpolicy
```

| NAME | POD-SELECTOR | AGE | denied-metadata-access | <no |
|------|--------------|-----|------------------------|-----|
| ne>  | 8s           |     |                        |     |

4. Verify that all apps in Pods in the namespace are blocked from accessing the AWS instance metadata service.

## Grant Access to Specific Apps in a Namespace

To configure a Kubernetes Network Policy to grant access to the AWS instance metadata service for apps in a specific Pod:

1. To create an `allow` Network Policy for apps in a Pod with a specific Pod label:
  1. Create a YAML configuration file named `np-allow.yml`.
  2. Populate the YAML file with one of the following `allow` Network Policy configurations:

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
```

```

name: POLICY-NAME
namespace: NAMESPACE
annotations:
 kubernetes.io/ingress.class: "nsx"
spec:
 podSelector:
 matchLabels:
 POD-LABEL
 policyTypes:
 - Egress
 egress:
 - to:
 - ipBlock:
 cidr: 169.254.169.254/32

```

Where:

- **POLICY-NAME** is the name for this Network Policy. For example `allow-metadata-access`.
- **NAMESPACE** is the name of the namespace to apply the Network Policy to. For example, `default` to manage access from Pods in the `default` namespace.
- **POD-LABEL** is the Pod label for the Pod to grant access to. Only the Pods tagged with the `POD-LABEL` label are affected by this configuration. For example, `app: nginx`.

3. Save the YAML configuration file.
2. To apply the `allow` Network Policy to your cluster:

```
kubectl apply -f np-allow.yml
```

For example:

```
kubectl apply -f np-allow.yml
```

```
networkpolicy.networking.k8s.io/allow-metadata-access created
```

3. Verify the Network Policy has been applied:

```
kubectl get networkpolicy
```

For example:

```
kubectl get networkpolicy
```

| NAME   | POD-SELECTOR | AGE | allow-metadata-access | app |
|--------|--------------|-----|-----------------------|-----|
| =nginx | 3s           |     |                       |     |

- Verify that only the apps in Pod(s) with the specified label have access to AWS instance metadata.

## Secure Access for All Namespaces Using an Antrea Cluster-Wide Network Policy

You can use an Antrea cluster-wide Kubernetes Network Policy to manage Pod access to AWS instance metadata.

The benefit of using an Antrea cluster-wide Network Policy is that a single configuration applies to all namespaces, avoiding the need to create a standard Network Policy for each namespace you want to manage.

To manage app access to AWS instance metadata using an Antrea cluster-wide Kubernetes Network Policy:

- Deny Access to All Namespaces Using Antrea
- Allow Access to a Specific Application Using Antrea

For more information on the benefits of using an Antrea Network Policy configuration, see [Antrea Network Policy CRDs](#) in the Antrea GitHub repository.

### Deny Access to All Namespaces Using Antrea

To deny app access to AWS instance metadata from all cluster namespaces using an Antrea cluster-wide Kubernetes Network Policy:

- To create a `deny` Network Policy:
  - Create a YAML configuration file named `np-cluster-deny.yml`.
  - Populate the YAML file with the following `deny` Network Policy configuration:

```
apiVersion: crd.antrea.io/v1alpha1
kind: ClusterNetworkPolicy
metadata:
 name: POLICY-NAME
spec:
 priority: 3 ##### =====> deny access should have lower priority
 than allow access , or use 'tier' to determine what is taking effect first
 appliedTo:
 - podSelector: {}
 egress:
 - action: Drop
 to:
 - ipBlock:
 cidr: 169.254.169.254/32
```

Where `POLICY-NAME` is the name for this Network Policy. For example `deny-`

```
 metadata-access.
```

3. Save the YAML configuration file.
2. To apply the `deny` Network Policy to your cluster:

```
kubectl apply -f np-cluster-deny.yml
```

For example:

```
kubectl apply -f np-cluster-deny.yml
```

```
clusternetworkpolicy.crd.antrea.io/deny-metadata-access created
```

3. Verify the Network Policy has been applied:

```
kubectl get clusternetworkpolicies.crd.antrea.io -owide
```

For example:

```
kubectl get clusternetworkpolicies.crd.antrea.io -owide
```

| NAME                              | TIER        | PRIORITY | DESIRED NODES | CURRE |
|-----------------------------------|-------------|----------|---------------|-------|
| NT NODES AGE deny-metadata-access | application | 3        | 3             |       |
| 3                                 | 37s         |          |               |       |

4. Verify that apps in all namespaces are blocked from accessing AWS instance metadata.

## Allow Access to a Specific App Using Antrea

To configure an Antrea Kubernetes Network Policy to grant access to AWS instance metadata for apps in a specific Pod:

1. To create an Antrea `allow` Network Policy:
  1. Create a YAML configuration file named `np-cluster-allow.yml`.
  2. Populate the YAML file with the following `allow` Network Policy configuration:

```
apiVersion: crd.antrea.io/v1alpha1
```

```

kind: ClusterNetworkPolicy
metadata:
 name: POLICY-NAME
spec:
 priority: 2
 appliedTo:
 - podSelector:
 matchLabels:
 POD-LABEL
 egress:
 - action: Allow
 to:
 - ipBlock:
 cidr: 169.254.169.254/32

```

Where:

- `POLICY-NAME` is the name for this Network Policy. For example `allow-metadata-access`.
- `POD-LABEL` is the Pod label for the Pod to grant access to. Only the Pods tagged with the `POD-LABEL` label are affected by this configuration. For example, `app: nginx`.

3. Save the YAML configuration file.
2. To apply the `allow` Network Policy to your cluster:

```
kubectl apply -f np-cluster-allow.yml
```

For example:

```
kubectl apply -f np-cluster-allow.yml
```

```
clusternetworkpolicy.crd.antrea.io/allow-metadata-access created
```

3. Verify the Network Policy has been applied:

```
kubectl get clusternetworkpolicies.crd.antrea.io -owide
```

For example:

```
kubectl get clusternetworkpolicies.crd.antrea.io -owide
```

| NAME     | TIER | PRIORITY              | DESIRED NODES | CURRE |
|----------|------|-----------------------|---------------|-------|
| NT NODES | AGE  | allow-metadata-access | application   | 2     |
| 1        |      | 33s                   |               | 1     |

4. Verify that only the apps in Pod(s) with the specified label have access to AWS instance metadata.

# Diagnosing and Troubleshooting Tanzu Kubernetes Grid Integrated Edition

This topic is intended to provide assistance when diagnosing and troubleshooting issues installing or using VMware Tanzu Kubernetes Grid Integrated Edition.

See the following sections:

- General Troubleshooting
- Using BOSH Diagnostic Commands in Tanzu Kubernetes Grid Integrated Edition
- Verifying Deployment Health
- Service Interruptions
- Troubleshooting the Management Console

## General Troubleshooting

### TKGI API is Slow or Times Out

#### Symptom

When you run TKGI CLI commands, the TKGI API times out or is slow to respond.

#### Explanation

The TKGI API VM requires more resources.

#### Solution

1. Navigate to <https://YOUR-OPS-MANAGER-FQDN/> in a browser to log in to the Ops Manager Installation Dashboard.
2. Select the **Tanzu Kubernetes Grid Integrated Edition** tile.
3. Select the **Resource Config** page.
4. For the **TKGI API** job, select a **VM Type** with greater CPU and memory resources.
5. Click **Save**.
6. Click the **Installation Dashboard** link to return to the Installation Dashboard.
7. Click **Review Pending Changes**. Review the changes that you made. For more information, see [Reviewing Pending Product Changes](#).
8. Click **Apply Changes**.

### All Cluster Operations Fail

## Symptom

All TKGI CLI cluster operations fail including attempts to create or delete clusters with `tkgi create-cluster` and `tkgi delete-cluster`.

The output of `tkgi cluster CLUSTER-NAME` contains `Last Action State: error`, and the output of `bosh -e ENV-ALIAS -d SERVICE-INSTANCE vms` indicates that the `Process State` of at least one deployed node is `failing`.

## Explanation

If any deployed control plane or worker nodes run out of disk space in `/var/vcap/store`, all cluster operations such as the creation or deletion of clusters will fail.

## Diagnostics

To confirm that there is a disk space issue, check recent BOSH activity for any disk space error messages.

1. Log in to the BOSH Director and run `bosh tasks`. The output from `bosh tasks` provides details about the tasks that the BOSH Director has run. See [Using BOSH Diagnostic Commands in Tanzu Kubernetes Grid Integrated Edition](#) for more information about logging in to the BOSH Director.
2. In the BOSH command output, locate a task that attempted to perform a cluster operation, such as cluster creation or deletion.
3. To retrieve more information about the task, run the following command:

```
bosh -e MY-ENVIRONMENT task TASK-NUMBER
```

Where:

- `MY-ENVIRONMENT` is the name of your BOSH environment.
- `TASK-NUMBER` is the number of the task that attempted to create the cluster.

For example:

```
$ bosh -e tkgi task 23
```

4. In the output, look for the following text string:

```
no space left on device
```

5. Check the health of your deployed Kubernetes clusters by following the procedure in [Verifying Deployment Health](#).
6. In the output of `bosh -e ENV-ALIAS -d SERVICE-INSTANCE vms`, look for any nodes that display `failing` as their `Process State`. For example:

| Instance<br>IPs                                          | VM CID                                  | Process State    | AZ           | VM Type | Active |
|----------------------------------------------------------|-----------------------------------------|------------------|--------------|---------|--------|
| master/3a3adc92-14ce-4cd4-a12c-6b5eb03e33d6<br>0.0.11.10 | vm-09027f0e-dac5-498e-474e-b47f2cda614d | failing<br>small | az-1<br>true |         |        |

7. Make a note of the plan assigned to the failing node.

### Solution

1. In the Tanzu Kubernetes Grid Integrated Edition tile, locate the plan assigned to the failing node.
2. In the plan configuration, select a larger VM type for the plan's control plane or worker nodes or both.

For more information about scaling existing clusters by changing the VM types, see [Scale Vertically by Changing Cluster Node VM Sizes in the TKGI Tile](#).

## Cluster Creation Fails

### Symptom

When creating a cluster, you run `tkgi cluster CLUSTER-NAME` to monitor the cluster creation status. In the command output, the value for **Last Action State** is `error`.

### Explanation

There was an error creating the cluster.

### Diagnostics

1. Log in to the BOSH Director and run `bosh tasks`. The output from `bosh tasks` provides details about the tasks that the BOSH Director has run. See [Using BOSH Diagnostic Commands in Tanzu Kubernetes Grid Integrated Edition](#) for more information about logging in to the BOSH Director.
2. In the BOSH command output, locate the task that attempted to create the cluster.
3. To retrieve more information about the task, run the following command:

```
bosh -e MY-ENVIRONMENT task TASK-NUMBER
```

Where:

- ◊ `MY-ENVIRONMENT` is the name of your BOSH environment.
- ◊ `TASK-NUMBER` is the number of the task that attempted to create the cluster.

For example:

```
$ bosh -e tkgi task 23
```

BOSH logs are used for error diagnostics but if the issue you see in the BOSH logs is related to using or managing Kubernetes, you should consult the [Kubernetes Documentation](#) for troubleshooting that issue.

For troubleshooting failed BOSH tasks, see the [BOSH documentation](#).

## Cluster Deletion Fails

## Symptom

When deleting a cluster in a large-scale NSX-T environment, `TKGI delete-cluster` becomes stuck.

## Explanation

A TKGI-API process has timed out and cluster deletion is stuck.

## Solution

To avoid the TKGI-API process time out, increase the **TKGI Operation Timeout**:

1. SSH to the TKGI Control Plane VM.
2. Change directory to `/var/vcap/jobs/pks-nsx-t-osb-proxy`.
3. Run the following command:

```
time ./bin/ncp_cleanup test-read ROUTER-ID
```

Where `ROUTER-ID` is your NSX-T Tier-0 Router ID.

For example:

```
pivotal-container-service/88d4bf76-d3967d53b4c4:/var/vcap/jobs/pks-nsx-t-osb-proxy# time ./bin/ncp_cleanup test-read 8dc31113-64e8-40bb-83fb-1af75857d5ae real 1m28.057s user 0m13.121s sys 0m0.629s
```

4. Collect the returned `real` value.
5. Add 30 seconds to the `real` value and convert the sum from minutes-seconds to seconds, rounding up. For example, sum, convert, and round `1m28.057s` to `120`.
6. Convert the summed value to milliseconds. This is your calculated Operation Timeout value.
7. Configure the **TKGI Operation Timeout** field on the TKGI Tile with your calculated Operation Timeout value. For more information on configuring the **TKGI Operation Timeout** field, see [Networking in Installing TKGI on vSphere with NSX-T](#).

## Cannot Re-Create a Cluster that Failed to Deploy

### Symptom

After cluster creation fails, you cannot re-run `tkgi create-cluster` to attempt creating the cluster again.

### Explanation

Tanzu Kubernetes Grid Integrated Edition does not automatically clean up the failed BOSH deployment. Running `tkgi create-cluster` using the same cluster name creates a name clash error in BOSH.

### Solution

Log in to the BOSH Director and delete the BOSH deployment manually, then retry the `tkgi delete-cluster` operation. After cluster deletion succeeds, re-create the cluster.

1. Log in to the BOSH Director and obtain the deployment name for cluster you want to delete.

For instructions, see [Using BOSH Diagnostic Commands in Tanzu Kubernetes Grid Integrated Edition](#).

- Run the following BOSH command:

```
bosh -e MY-ENVIRONMENT delete-deployment -d DEPLOYMENT-NAME
```

Where:

- `MY-ENVIRONMENT` is the name of your BOSH environment.
- `DEPLOYMENT-NAME` is the name of your BOSH deployment.



**Note:** If necessary, you can append the `-force` flag to delete the deployment.

- Run the following TKGI command:

```
tkgi delete-cluster CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of your Tanzu Kubernetes Grid Integrated Edition cluster.



**Note:** Use only lowercase characters in your TKGI-provisioned Kubernetes cluster names if you manage your clusters with Tanzu Mission Control (TMC). Clusters with names that include an uppercase character cannot be attached to TMC.

- To re-create the cluster, run the following TKGI command:

```
tkgi create-cluster CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of your Tanzu Kubernetes Grid Integrated Edition cluster.



**Note:** Use only lowercase characters when naming your cluster if you manage your clusters with Tanzu Mission Control (TMC). Clusters with names that include an uppercase character cannot be attached to TMC.

## Windows Stemcell for vSphere Creation Fails with Login Issue

### Symptom

The `stembuild construct` command fails with error: `Cannot complete login due to an incorrect user name or password.`

### Explanation

Your vCenter login contains special characters, or you have `GOVC` environment variables set locally.

### Solution

For special characters, see [Authentication Error with Special Characters in stembuild Commands](#), in the TAS for VMs [Windows] documentation.

For `GOVC` variables, follow the steps to unset the variables in [Step 4: Construct the BOSH Stemcell](#), in the TAS for VMs [Windows] documentation.

## Cannot Access Add-On Features or Functions

### Symptom

You cannot access a feature or function provided by a Kubernetes add-on.

For example, pods cannot resolve DNS names, and error messages report the service `CoreDNS` is invalid. If `CoreDNS` is not deployed, the cluster typically fails to start.

### Explanation

Kubernetes features and functions are provided by Tanzu Kubernetes Grid Integrated Edition add-ons. DNS resolution, for example, is provided by the `CoreDNS` service.

To activate these add-ons, Ops Manager must run scripts after deploying Tanzu Kubernetes Grid Integrated Edition. You must configure Ops Manager to automatically run these post-deploy scripts.

### Solution

Perform the following steps to configure Ops Manager to run post-deploy scripts to deploy the missing add-ons to your cluster.

1. Navigate to <https://YOUR-OPS-MANAGER-FQDN/> in a browser to log in to the Ops Manager Installation Dashboard.
2. Click the **BOSH Director** tile.
3. Select **Director Config**.
4. Select **Enable Post Deploy Scripts**.



**Note:** This setting activates post-deploy scripts for all tiles in your Ops Manager installation.

5. Click **Save**.
6. Click the **Installation Dashboard** link to return to the Installation Dashboard.
7. Click **Review Pending Changes**. Review the changes that you made. For more information, see [Reviewing Pending Product Changes](#).
8. Click **Apply Changes**.
9. After Ops Manager finishes applying changes, enter `tkgi delete-cluster` on the command line to delete the cluster. For more information, see [Deleting Clusters](#).
10. On the command line, enter `tkgi create-cluster` to recreate the cluster. For more information, see [Creating Clusters](#).

## Resurrecting VMs Causes Incorrect Permissions in vSphere HA

### Symptoms

Output resulting from the `bosh vms` command alternates between showing that the VMs are `failing` and showing that the VMs are `running`. The operator must run the `bosh vms` command multiple times to see this cycle.

### Explanation

The VMs' permissions are altered during the restarting of the VM so operators have to reset permissions every time the VM reboots or is redeployed.

VMs cannot be successfully resurrected if the resurrection state of your VM is set to `off` or if the vSphere HA restarts the VM before BOSH is aware that the VM is down. For more information about VM resurrection, see [Resurrection](#) in the BOSH documentation.

### Solution

Run the following command on all of your control plane and worker VMs:

```
bosh -environment BOSH-DIRECTOR-NAME -deployment DEPLOYMENT-NAME ssh INSTANCE-GROUP-NAME -c "sudo /var/vcap/jobs/kube-controller-manager/bin/pre-start; sudo /var/vcap/jobs/kube-apiserver/bin/post-start"
```

Where:

- `BOSH-DIRECTOR-NAME` is your BOSH Director name.
- `DEPLOYMENT-NAME` is the name of your BOSH deployment.
- `INSTANCE-GROUP-NAME` is the name of the BOSH instance group you are referencing.

The above command, when applied to each VM, gives your VMs the correct permissions.

## Worker Node Hangs Indefinitely

### Symptoms

After making your selection in the **Upgrade all clusters errand** section, the worker node might hang indefinitely. For more information about monitoring the **Upgrade all clusters errand** using the BOSH CLI, see [Upgrade the TKGI Tile in Upgrading Tanzu Kubernetes Grid Integrated Edition \(Flannel Networking\)](#).

### Explanation

During the Tanzu Kubernetes Grid Integrated Edition tile upgrade process, worker nodes are cordoned and drained. This drain is dependent on Kubernetes being able to unschedule all pods. If Kubernetes is unable to unschedule a pod, then the drain hangs indefinitely. Kubernetes might be unable to unschedule the node if the `PodDisruptionBudget` object has been configured to permit zero disruptions and only a single instance of the pod has been scheduled.

In your spec file, the `.spec.replicas` configuration sets the total amount of replicas that are available in your app. `PodDisruptionBudget` objects specify the amount of replicas, proportional to the total, that must be available in your app, regardless of downtime. Operators can configure `PodDisruptionBudget` objects for each app using their spec file.

Some apps deployed using Helm charts might have a default `PodDisruptionBudget` set. For more information on configuring `PodDisruptionBudget` objects using a spec file, see [Specifying a PodDisruptionBudget](#) in the Kubernetes documentation.

If `.spec.replicas` is configured correctly, you can also configure the default node drain behavior to prevent cluster upgrades from hanging or failing.

## Solution

To resolve this issue, do one of the following:

- Configure `.spec.replicas` to be greater than the `PodDisruptionBudget` object.

When the number of replicas configured in `.spec.replicas` is greater than the number of replicas set in the `PodDisruptionBudget` object, disruptions can occur.

For more information, see [How Disruption Budgets Work](#) in the Kubernetes documentation.

For more information about workload capacity and uptime requirements in Tanzu

Kubernetes Grid Integrated Edition, see [Prepare to Upgrade](#) in *Upgrading Tanzu Kubernetes Grid Integrated Edition (Antrea and Flannel Networking)*.

- Configure the default node drain behavior by doing the following:

1. Navigate to **Ops Manager Installation > Tanzu Kubernetes Grid Integrated Edition > Plans**.
2. Set the default node drain behavior by configuring the following fields:

| Field                                                                                                                                     | Instructions                                                                                                                                                                                                                                                                      |
|-------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Node Drain Timeout</b>                                                                                                                 | Enter a timeout in minutes for the node to drain pods. You must enter a valid integer between <code>0</code> and <code>1440</code> . If you set this value to <code>0</code> , the node drain does not terminate.                                                                 |
| <b>Pod Shutdown Grace</b>                                                                                                                 | Enter a timeout in seconds for the node to wait before it forces the pod to terminate. You must enter a valid integer between <code>-1</code> and <code>86400</code> . If you set this value to <code>-1</code> , the timeout is set to the default timeout specified by the pod. |
| <b>Force node to drain even if it has running pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or StatefulSet.</b> | If you activate this configuration, the node still drains when pods are not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or StatefulSet.                                                                                                                        |
| <b>Force node to drain even if it has running DaemonSet-managed pods.</b>                                                                 | If you activate this configuration, the node still drains when pods are managed by a DaemonSet.                                                                                                                                                                                   |
| <b>Force node to drain even if it has running pods using emptyDir.</b>                                                                    | If you activate this configuration, the node still drains when pods are using an emptyDir volume.                                                                                                                                                                                 |
| <b>Force node to drain even if pods are still running after timeout.</b>                                                                  | If you activate this configuration and then during the timeout pods fail to drain on the worker node, the node forces running pods to terminate and the upgrade or scale continues.                                                                                               |



**Warning:** If you select **Force node to drain even if pods are still running after timeout**, the node halts all running workloads on pods. Before enabling this configuration, set **Node Drain Timeout** to

greater than 0.



**Warning:** If you deselect **Force node to drain even if it has running DaemonSet-managed pods with Enable Metric Sink Resources, Enable Log Sink Resources, or Enable Node Exporter** selected, the upgrade will fail as all options deploy a DaemonSet in the `pks-system` namespace.

3. Navigate to **Ops Manager Installation Dashboard > Review Pending Changes**, select **Upgrade all clusters errand**, and **Apply Changes**. The new behavior takes effect during the next upgrade, not immediately after applying your changes.



**Note:** You can also use the TKGI CLI to configure node drain behavior. To configure the default node drain behavior with the TKGI CLI, run `tkgi update-cluster` with an action flag. You can view the current node drain behavior with `tkgi cluster -details`. For more information, see [Configure Node Drain Behavior in Upgrade Preparation Checklist for Tanzu Kubernetes Grid Integrated Edition v1.9](#). **Warning:** Do not use `tkgi update-cluster` on clusters configured with a network profile CNI configuration.

## Cannot Authenticate to an OpenID Connect-Enabled Cluster

### Symptom

When you authenticate to an OpenID Connect-enabled cluster using an existing kubeconfig file, you see an authentication or authorization error.

### Explanation

`users.user.auth-provider.config.id-token` and `users.user.auth-provider.config.refresh-token` contained in the kubeconfig file for the cluster might have expired.

### Solution

1. Upgrade the TKGI CLI to v1.2.0 or later.

To download the TKGI CLI, navigate to [VMware Tanzu Network](#). For more information, see [Installing the TKGI CLI](#).

2. Obtain a kubeconfig file that contains the new tokens by running the following command:

```
tkgi get-credentials CLUSTER-NAME
```

Where `CLUSTER-NAME` is the name of your cluster.

For example:

```
$ tkgi get-credentials tkgi-example-cluster
```

```
Fetching credentials for cluster tkgi-example-cluster. Context set for
cluster tkgi-example-cluster.
```

```
You can now switch between clusters by using: $kubectl config use-cont
ext <cluster-name>
```



**Note:** If your operator has configured Tanzu Kubernetes Grid Integrated Edition to use a SAML identity provider, you must include an additional SSO flag to use the above command. For information about the SSO flags, see the section for the above command in [TKGI CLI](#). For information about configuring SAML, see [Connecting Tanzu Kubernetes Grid Integrated Edition to a SAML Identity Provider](#)

3. Connect to the cluster using kubectl.

If you continue to see an authentication or authorization error, verify that you have sufficient access permissions for the cluster.

## Cannot Access Apps Deployed to Clusters That Utilize Websocket

### Symptom

Your NSX-T LB disconnects the sessions for your apps deployed to clusters utilizing websocket. These apps are inaccessible or non-functional.

### Explanation

Tanzu Kubernetes Grid Integrated Edition on vSphere with NSX-T fully supports websocket. The most likely cause for this behavior is a connectivity issue specific to supporting websocket.

### Solution

Review your configuration for a source for the connectivity issues:

1. Review the connectivity to the NSX-T LB instance.
2. Confirm the devices between your NSX-T LB and app are not blocking websocket.

## Login Failed Error: Credentials were rejected

### Symptom

TKGI login command fails with an error “Credentials were rejected, please try again.”

### Explanation

You might experience this issue when a large number of pods are running continuously in your Tanzu Kubernetes Grid Integrated Edition deployment. As a result, the persistent disk on the TKGI Database VM runs out of space.

### Solution

1. Check the total number of pods in your Tanzu Kubernetes Grid Integrated Edition

deployments.

2. If there are a large number of pods such as over 1,000 pods, then check the amount of available persistent disk space on the TKGI Database VM.
3. If available disk space is low, increase the amount of persistent disk storage on the TKGI Database VM depending on the number of pods in your Tanzu Kubernetes Grid Integrated Edition deployment. Refer to the table in the following section.

## Storage Requirements for Large Numbers of Pods

If you expect the cluster workload to run a large number of pods continuously, then increase the size of persistent disk storage allocated to the TKGI Database VM as follows:

| Number of Pods | Persistent Disk Requirements (GB) |
|----------------|-----------------------------------|
| 1,000 pods     | 20                                |
| 5,000 pods     | 100                               |
| 10,000 pods    | 200                               |
| 50,000 pods    | 1,000                             |

## Login Failed Errors Due to Server State

### Symptom

You encounter an error similar to one of the following when running a `kubectl` or `cluster` command:

- “Error: You must be logged in to the server (Unauthorized)”
- “Error: You are not currently authenticated. Please log in to continue”

### Explanation

You might experience this issue when your authentication server or a host has the incorrect time.

### Workaround

1. To refresh your credentials, run the following:

```
pks get-credentials
```

### Solution

1. To resolve the problem permanently, correct the time on the server with the incorrect time.

## Error: Failed Jobs

### Symptom

In stdout or log files, you see an error message referencing `post-start scripts failed` or `Failed Jobs`.

### Explanation

After deploying Tanzu Kubernetes Grid Integrated Edition, Ops Manager runs scripts to start a number of jobs. You must configure Ops Manager to automatically run these post-deploy scripts.

## Solution

Perform the following steps to configure Ops Manager to run post-deploy scripts.

1. Navigate to <https://YOUR-OPS-MANAGER-FQDN/> in a browser to log in to the Ops Manager Installation Dashboard.
2. Click the **BOSH Director** tile.
3. Select **Director Config**.
4. Select **Enable Post Deploy Scripts**.



**Note:** This setting activates post-deploy scripts for all tiles in your Ops Manager installation.

5. Click **Save**.
6. Click the **Installation Dashboard** link to return to the Installation Dashboard.
7. Click **Review Pending Changes**. Review the changes that you made. For more information, see [Reviewing Pending Product Changes](#).
8. Click **Apply Changes**.
9. (Optional) If it is a new deployment of Tanzu Kubernetes Grid Integrated Edition, follow the steps below:
  1. On the command line, enter `tkgi delete-cluster` to delete the cluster. For more information, see [Deleting Clusters](#).
  2. Enter `tkgi create-cluster` to recreate the cluster. For more information, see [Creating Clusters](#).

## Error: No Such Host

### Symptom

In stdio or log files, you see an error message that includes `lookup vm-WORKER-NODE-GUID on IP-ADDRESS: no such host`.

### Explanation

This error occurs on GCP when the Ops Manager Director tile uses 8.8.8.8 as the DNS server. When this IP range is in use, the control plane node cannot locate the route to the worker nodes.

## Solution

Use the Google internal DNS range, 169.254.169.254, as the DNS server.

## Error: FailedMount

### Symptom

In Kubernetes log files, you see a `Warning` event from kubelet with `FailedMount` as the reason.

## Explanation

A persistent volume fails to connect to the Kubernetes cluster worker VM.

## Diagnostics

- In your cloud provider console, verify that volumes are being created and attached to nodes.
- From the Kubernetes cluster control plane node, check the controller manager logs for errors attaching persistent volumes.
- From the Kubernetes cluster worker node, check kubelet for errors attaching persistent volumes.

## Error: Plan Not Found

### Symptom

Plan not found error when an active plan is deactivated.

### Explanation

You might receive the error “plan UUID not found” if, after creating a cluster using a plan (such as Plan 1), you then deactivate the plan (Plan 1) from the TKGI Tile in Ops Manager and then **Save** and **Apply Changes** with the **Upgrade all clusters errand** selected.

Ops Manager does not have capability to check clusters that are using a particular plan. Only when user saves the plan, the deployment process will check whether a plan can be deactivated. The error message "plan is displayed in the Ops Manager logs.

### Solution

1. Do not deactivate a plan that is in use by or more clusters.
2. Run the command `tkgi cluster my-cluster --details` to view what plan the cluster is using.

## Using BOSH Diagnostic Commands in Tanzu Kubernetes Grid Integrated Edition

This topic describes how to access information about your VMware Tanzu Kubernetes Grid Integrated Edition deployment by using the BOSH Command Line Interface (BOSH CLI).

## Overview

BOSH diagnostic commands such as `bosh ssh` and `bosh vms` enable you to access information about your Tanzu Kubernetes Grid Integrated Edition deployment. For example, you can access Tanzu Kubernetes Grid Integrated Edition log files after SSHing into the TKGI API or a Kubernetes cluster VM:

1. [SSH into the TKGI API VM or SSH into a Kubernetes Cluster VM](#)
2. [View Log Files](#)

## Log in to the BOSH Director VM

To set a BOSH alias for your Tanzu Kubernetes Grid Integrated Edition environment and log in to the BOSH Director VM, follow the steps below:

1. Gather your credential and IP address information for the BOSH Director and SSH into the Ops Manager VM. For instructions, see [Advanced Troubleshooting with the BOSH CLI](#).
2. To create a BOSH alias for your Tanzu Kubernetes Grid Integrated Edition environment, run the following command:

```
bosh alias-env ENVIRONMENT \
-e BOSH-DIRECTOR-IP \
--ca-cert /var/tempest/workspaces/default/root_ca_certificate
```

Where:

- **ENVIRONMENT** is an alias of your choice. For example, `tkgi`.
- **BOSH-DIRECTOR-IP** is the BOSH Director IP address you located in the first step. For example, `10.0.0.3`.

For example:

```
$ bosh alias-env tkgi -e 10.0.0.3
--ca-cert /var/tempest/workspaces/default/root_ca_certificate
```

3. To log in to the BOSH Director using the alias you set, run the following command:

```
bosh -e ENVIRONMENT login
```

For example:

```
$ bosh -e tkgi login
```

Alternatively, you can set the BOSH environment variables on the Ops Manager VM to authenticate with the BOSH Director VM. For more information, see [Authenticate with the BOSH Director VM](#) in *Advanced Troubleshooting with the BOSH CLI* in the Ops Manager documentation.

## SSH into the TKGI API VM

To SSH into the TKGI API VM using the BOSH CLI, follow the steps below:

1. Log in to the BOSH Director. For instructions, see [Log in to the BOSH Director VM](#).
2. To identify your TKGI deployment name, run the following command:

```
bosh -e ENVIRONMENT deployments
```

Where **ENVIRONMENT** is your BOSH environment alias.

For example:

```
$ bosh -e tkgi deployments
```

Your TKGI deployment name begins with `pivotal-container-service` and includes a BOSH-generated identifier.

- To identify your TKGI API VM name, run the following command:

```
bosh -e ENVIRONMENT -d DEPLOYMENT vms
```

Where:

- `ENVIRONMENT` is the BOSH environment alias.
- `DEPLOYMENT` is your TKGI deployment name.

For example:

```
$ bosh -e tkgi -d pivotal-container-service-a1b2c333d444e5f66a77 vms
```

Your TKGI API VM name begins with `pivotal-container-service` and includes a BOSH-generated identifier.



**Note:** The TKGI API VM identifier is different from the identifier in your TKGI deployment name.

- To SSH into the TKGI API VM:

```
bosh -e ENVIRONMENT -d DEPLOYMENT ssh TKGI-API-VM
```

Where:

- `ENVIRONMENT` is the BOSH environment alias.
- `DEPLOYMENT` is your TKGI deployment name.
- `TKGI-API-VM` is your TKGI API VM name.

For example:

```
$ bosh -e tkgi
-d pivotal-container-service-a1b2c333d444e5f66a77
ssh pivotal-container-service/000a1111-222b-3333-4cc5-de66f7a8899b
```

## SSH into the TKGI Database VM

To SSH into a TKGI Database VM using the BOSH CLI, follow the steps below:

- Log in to the BOSH Director. For instructions, see [Log in to the BOSH Director VM](#).
- To identify your TKGI deployment name:

```
bosh -e ENVIRONMENT deployments
```

Where **ENVIRONMENT** is your BOSH environment alias.

For example:

```
$ bosh -e tkgi deployments
```

Your TKGI deployment name begins with **pivotal-container-service** and includes a BOSH-generated identifier.

- To identify your TKGI Database VM names:

```
bosh -e ENVIRONMENT -d DEPLOYMENT vms
```

Where:

- ENVIRONMENT** is the BOSH environment alias.
- DEPLOYMENT** is your TKGI deployment name.

For example:

```
$ bosh -e tkgi -d pivotal-container-service-a1b2c333d444e5f66a77 vms
```

Your TKGI Database VM names begin with **pks-db** and include a BOSH-generated identifier.

- Choose one of the returned TKGI Database VMs as the database VM to SSH into.
- To SSH into the selected TKGI Database VM, run the following command:

```
bosh -e ENVIRONMENT -d DEPLOYMENT ssh TKGI-DB-VM
```

Where:

- ENVIRONMENT** is the BOSH environment alias.
- DEPLOYMENT** is your TKGI deployment name.
- TKGI-DB-VM** is the name of the TKGI Database VM to SSH into.

For example:

```
$ bosh -e tkgi
-d pivotal-container-service-a1b2c333d444e5f66a77
ssh pks-db/000a4444-555b-6666-4cc5-de66f8a9900b
```

## SSH into a Kubernetes Cluster VM

Each Kubernetes cluster corresponds to a BOSH deployment. To SSH into a TKGI-provisioned Kubernetes cluster VM using the BOSH CLI, follow the steps below:

- Log in to the BOSH Director. For instructions, see [Log in to the BOSH Director VM](#).
- To identify your Kubernetes cluster deployment name, run the following command:

```
bosh -e ENVIRONMENT deployments
```

Where `ENVIRONMENT` is your BOSH environment alias.

For example:

```
$ bosh -e tkgi deployments
```

Kubernetes cluster deployment names begin with `service-instance` and include a BOSH-generated identifier.

- To identify your Kubernetes cluster VM name, run the following command:

```
bosh -e ENVIRONMENT -d DEPLOYMENT vms
```

Where:

- `ENVIRONMENT` is the BOSH environment alias.
- `DEPLOYMENT` is your Kubernetes cluster deployment name.

For example:

```
$ bosh -e tkgi -d service-instance_ae681cd1-7ff4-4661-b12c-49a5b543f1
6f vms
```

Each Kubernetes cluster VM name begins with `master` or `worker` and includes a BOSH-generated identifier.

- To SSH into your Kubernetes cluster VM, run the following command:

```
bosh -e ENVIRONMENT -d DEPLOYMENT ssh CLUSTER-VM
```

Where:

- `ENVIRONMENT` is the BOSH environment alias.
- `DEPLOYMENT` is your Kubernetes cluster deployment name.
- `CLUSTER-VM` is your Kubernetes cluster VM name, either `master/VM-ID` or `worker/VM-ID`.

For example:

```
$ bosh -e tkgi -d service-instance_ae681cd1-7ff4-4661-b12c-49a5b543f1
6f ssh master/000a1111-222b-3333-4cc5-de66f7a8899b
```

## View Log Files

Log files contain error messages and other information you can use to diagnose issues with your Tanzu Kubernetes Grid Integrated Edition deployment. To access Tanzu Kubernetes Grid Integrated Edition log files, SSH into the TKGI API VM, or a Kubernetes cluster VM, and then follow the steps below:

1. To act as super user on your VM, run the following command:

```
sudo su
```

2. Navigate to the `/var/vcap/sys/log` log file directory:

```
cd /var/vcap/sys/log
```

3. Examine the contents of the `/var/vcap/sys/log` directory. For example, when diagnosing issues with a Kubernetes cluster VM, you might want to review the following log files:

- On a control plane VM, examine the `kube-apiserver` subdirectory.
- On a worker VM, examine the `kubelet` subdirectory.

## Verifying Deployment Health

This topic describes how to check the health of your Tanzu Kubernetes Grid Integrated Edition deployment and the nodes, pods, and clusters that it hosts.

### Verify Kubernetes Node and Pod Health

Verify the health of your Kubernetes nodes and pods by following the steps below:

1. From the Ops Manager VM, run the following command:

```
bosh -e ENVIRONMENT login
```

Where `ENVIRONMENT` is the alias you set for your BOSH Director. For more information, see [Using BOSH Diagnostic Commands in Tanzu Kubernetes Grid Integrated Edition](#).

For example:

```
$ bosh -e tkgi login
```

2. To verify that all nodes are in a ready state, run the following command for all Kubernetes contexts:

```
kubectl get nodes
```

3. To verify that all pods are running, run the following command for all Kubernetes contexts:

```
kubectl get pods --all-namespaces
```

### Verify Kubernetes Cluster Health

Verify the health of your Kubernetes clusters by following the steps below:

1. From the Ops Manager VM, run the following command:

```
bosh -e ENVIRONMENT login
```

Where `ENVIRONMENT` is the alias you set for your BOSH Director. For more information, see [Using BOSH Diagnostic Commands in Tanzu Kubernetes Grid Integrated Edition](#).

For example:

```
$ bosh -e tkgi login
```

2. To get the deployment name of a target Kubernetes cluster, run the following command:

```
bosh deployments
```

For example:

```
$ bosh deployments
Using environment '30.0.0.10' as client 'ops_manager'
Name Release(s) Team(s)
Name Release(s) Team(s)
Stemcell(s) -
harbor-container-registry-b4023f6857207b237399 bosh-dns/1.10.0
 bosh-vsphere-esxi-ubuntu-jammy-go_agent/170.15 -
 harbor-container-registr
y/1.7.3-build.2
pivotal-container-service-7e64d53fc570503b5690 backup-and-restore-sdk/1
.9.0 bosh-vsphere-esxi-ubuntu-jammy-go_agent/170.15 -
 bosh-dns/1.10.0
 bpm/0.13.0
 cf-mysql/36.14.0
 cfcr-etcd/1.8.0
 harbor-container-registr
y/1.7.3-build.2
 kubo/0.25.9
 kubo-service-adapter/1.3
.3-build.1
0
 nsx-cf-cni/2.3.1.1069341
 on-demand-service-broker
/0.24.0
 pks-api/1.3.3-build.1
 pks-helpers/50.0.0
 pks-nsx-t/1.19.0
 pks-telemetry/2.0.0-buil
d.113
 pks-vrli/0.7.0
 sink-resources-release/0
.1.15
 syslog/11.4.0
 uaa/64.0
 wavefront-proxy/0.9.0
service-instance_8de000ff-a87a-4930-81ba-106d42c2471e bosh-dns/1.10.0
 bosh-vsphere-esxi-ubuntu-jammy-go_agent/170.15 pivotal-contai
ner-service-7e64d53fc570503b5690
 bpm/0.13.0
 cfcr-etcd/1.8.0
 harbor-container-registr
y/1.7.3-build.2
 kubo/0.25.9
 nsx-cf-cni/2.3.1.1069341
0
```

```

d.113 pks-helpers/50.0.0
 pks-nsx-t/1.19.0
 pks-telemetry/2.0.0-buil

.1.15 pks-vrli/0.7.0
 sink-resources-release/0

 syslog/11.4.0
 wavefront-proxy/0.9.0

3 deployments

```

In the example above, `service-instance_8de000ff-a87a-4930-81ba-106d42c2471e` is the Kubernetes cluster deployment name.



**Note:** If you have deployed multiple Kubernetes clusters, determine the UUID using `tkgi clusters` and then match that UUID with the Kubernetes cluster deployment you are targeting.

- With each cluster in a deployment, or any specific cluster, check the status of the cluster's VMs by running the following command:

```
bosh -d K8S-DEPLOYMENT vms
```

Where `K8S-DEPLOYMENT` is the name of your Kubernetes cluster deployment. Kubernetes cluster deployment names begin with `service-instance` and include a unique BOSH-generated identifier.

This command returns the name of each VM comprising the Kubernetes cluster, including each control plane and worker node.

For example:

```

$ bosh -d service-instance•8de000ff-a87a-4930-81ba-106d42c2471e vms
Using environment '30.0.0.10' as client 'ops_manager'
Task 677. Done
Deployment 'service-instance•8de000ff-a87a-4930-81ba-106d42c2471e'
Instance Process State AZ IPs V
M CID VM Type Active
master/b6d3c263-1682-4c79-a9ab-35939127dedb running AZ-K8S 40.0.2.2 v
m-60dbc68-5538-4c4e-8c00-61edc003bb54 medium.disk true
worker/d450548a-2b0c-4494-8144-cf9b7ef9c825 running AZ-K8S 40.0.2.4 v
m-1bfdde6d-ce1d-4cdf-90d9-32bba260358f medium.disk true
worker/d7f882f0-33dd-43d3-ab5d-058bcc505088 running AZ-K8S 40.0.2.3 v
m-822cb573-411f-4c44-a32b-34e79520a7a6 medium.disk true
worker/e5e25ffe-f448-4d19-990b-89546118c502 running AZ-K8S 40.0.2.5 v
m-c6748604-8440-4b27-9cf4-10a70a02da24 medium.disk true

4 vms

Succeeded

```

- With each cluster in a deployment, or any specific cluster, check the status of the cluster's

processes by running the following command:

```
bosh -d K8S-DEPLOYMENT instances --ps
```

Where `K8S-DEPLOYMENT` is the name of your Kubernetes cluster deployment. Kubernetes cluster deployment names begin with `service-instance` and include a unique BOSH-generated identifier. This command returns status information for the processes on each Kubernetes cluster VM, including each control plane and worker node.

For example:

|                                             |         |        |          |                             |
|---------------------------------------------|---------|--------|----------|-----------------------------|
|                                             |         |        |          | kube-proxy                  |
| ~                                           | running | -      | -        | kubelet                     |
| ~                                           | running | -      | -        | nsx-kube-proxy              |
| ~                                           | running | -      | -        | nsx-node-agent              |
| ~                                           | running | -      | -        | ovs-vswitchd                |
| ~                                           | running | -      | -        | ovsdb-server                |
| ~                                           | running | -      | -        | pks-helpers-bosh-dns-resolv |
| onf                                         | running | -      | -        | -                           |
| worker/d7f882f0-33dd-43d3-ab5d-058bcc505088 | running | AZ-K8S | 40.0.2.3 |                             |
| ~                                           | running | -      | -        | blackbox                    |
| ~                                           | running | -      | -        | bosh-dns                    |
| ~                                           | running | -      | -        | bosh-dns-healthcheck        |
| ~                                           | running | -      | -        | bosh-dns-resolvconf         |
| ~                                           | running | -      | -        | containerd                  |
| ~                                           | running | -      | -        | kube-proxy                  |
| ~                                           | running | -      | -        | kubelet                     |
| ~                                           | running | -      | -        | nsx-kube-proxy              |
| ~                                           | running | -      | -        | nsx-node-agent              |
| ~                                           | running | -      | -        | ovs-vswitchd                |
| ~                                           | running | -      | -        | ovsdb-server                |
| ~                                           | running | -      | -        | pks-helpers-bosh-dns-resolv |
| onf                                         | running | -      | -        | -                           |
| worker/e5e25ffe-f448-4d19-990b-89546118c502 | running | AZ-K8S | 40.0.2.5 |                             |
| ~                                           | running | -      | -        | blackbox                    |
| ~                                           | running | -      | -        | bosh-dns                    |
| ~                                           | running | -      | -        | bosh-dns-healthcheck        |
| ~                                           | running | -      | -        | bosh-dns-resolvconf         |
| ~                                           | running | -      | -        | containerd                  |
| ~                                           | running | -      | -        | kube-proxy                  |
| ~                                           | running | -      | -        | kubelet                     |
| ~                                           | running | -      | -        | nsx-kube-proxy              |
| ~                                           | running | -      | -        | nsx-node-agent              |

|              |         |   |   |
|--------------|---------|---|---|
| running      | -       | - |   |
| ~            | running | - | - |
| ~            | running | - | - |
| ~            | running | - | - |
| onf          | running | - | - |
| 51 instances |         |   |   |

## Retrieve Cluster Upgrade Task ID

To retrieve the BOSH task ID for a recent cluster upgrade:

1. (Optional) To retrieve a list of clusters, run the `tkgi clusters` command.

For example:

```
$ tkgi clusters
Upgrade is available to TKGI Version: 1.9.0-build.1
TKGI Version Name k8s Version Plan Name UUID
 Status Action
1.9.0-build.5 Sample5 1.18.8 small 04b9e9c3-ed99-4a24-b30d-b5c74b
23d3b0 succeeded UPGRADE
1.9.0-build.5 Sample6 1.18.8 small 0bb39685-73b9-458c-a52c-e79295
a153b4 in progress UPGRADE
1.8.0-build.2 Sample2 1.17.5 small 26e80c96-e65c-47e7-be78-3c4361
4f0712 succeeded CREATE
1.8.0-build.2 Sample3 1.17.5 small 39667c39-b498-4065-b6fe-7119d5
79554b succeeded CREATE
1.8.0-build.2 Sample4 1.17.5 small 3ce19c9d-3e4c-48f3-8a4e-cd6d6a
124617 succeeded CREATE
```

2. Run the `tkgi tasks` command.

3. In the `tkgi tasks` output, find the cluster's `UPGRADE` task. The BOSH task ID is listed in the `ID` column.

For example, the upgrade task ID here for the cluster `Sample5` is `4925355e-44ed-4aea-abae-9f8e8d58c4d5`:

```
$ tkgi tasks
ID Type Status StartTime
 EndTime Clusters
4925355e-44ed-4aea-abae-9f8e8d58c4d5 UPGRADE done Sun, 28 Jun 52048 0
5:00:00 PST Tue, 30 Jun 52048 23:56:40 PST Sample5
0ba8cb6b-e136-466f-99a3-8071bc4d7741 UPGRADE in progress Tue, 30 Jun 52048 2
3:40:00 PST Tue, 30 Jun 52048 22:33:20 PST Sample6
```

- Use the `-1` flag to limit the number of tasks returned by the `tkgi tasks` command.  
For more information, see `tkgi tasks` in the *TKGI CLI* topic.

## Verify NCP Health (NSX-T Only)

NSX Container Plugin (NCP) runs as a BOSH host process. Each Kubernetes control plane node VM

has one running NCP process. If your cluster has multiple control plane nodes, one NCP process is active while the others are on standby.

To verify the `ncp` process is running, do the following:

1. Run `bosh instances` in your Tanzu Kubernetes Grid Integrated Edition environment:

```
bosh -e ENVIRONMENT -d K8S-DEPLOYMENT instances --ps
```

Where:

- `ENVIRONMENT` is the alias you set for your BOSH Director.
- `K8S-DEPLOYMENT` is the name of your Kubernetes cluster deployment. Kubernetes cluster deployment names begin with `service-instance` and include a unique BOSH-generated identifier.

For example:

```
$ bosh -e tkgi -d service-instance_8de000ff-a87a-4930-81ba-106d42c2471e instances --ps
Using environment '30.0.0.10' as client 'ops_manager'
Task 678. Done
Deployment 'service-instance_8de000ff-a87a-4930-81ba-106d42c2471e'
Instance Process
 Process State AZ IPs
apply-addons/ef0d09ae-d3ed-4832-8d3f-431d99730c26 -
 - AZ-K8S -
master/b6d3c263-1682-4c79-a9ab-35939127dedb -
 running AZ-K8S 40.0.2.2
~ - - blackbox
~ running - -
~ running - - bosh-dns
~ running - - bosh-dns-healthcheck
~ running - - bosh-dns-resolvconf
~ running - - etcd
~ running - - kube-apiserver
~ running - - kube-controller-manager
~ running - - kube-scheduler
~ running - - ncp
~ running - - pks-helpers-bosh-dns-resolv
onf running - - -
```

## Alternatively:

1. SSH into your target Kubernetes control plane node VM:

```
bosh -e ENVIRONMENT -d K8S-DEPLOYMENT ssh master/VM-ID
```

Where:

- ◊ `ENVIRONMENT` is the alias you set for your BOSH Director.
- ◊ `K8S-DEPLOYMENT` is the name of your Kubernetes cluster deployment. Kubernetes cluster deployment names begin with `service-instance` and include a unique BOSH-generated identifier.
- ◊ `VM-ID` is your Kubernetes control plane node VM ID. This is a unique BOSH-generated identifier.

For example:

```
$ bosh -e tkgi -d service-instance_8de000ff-a87a-4930-81ba-106d42c2471e ssh master/b6d3c263-1682-4c79-a9ab-35939127dedb
```

2. From the control plane node VM, run `monit summary`.
3. (Optional) To check if the `ncp` process on your target control plane node is active or on standby, run `/var/vcap/jobs/ncp/bin/nsxcli -c get ncp-master status`. This applies only to multi-control plane node clusters.

For information about troubleshooting NCP, see [NSX-T NCP troubleshooting and debug logging](#) in the VMware Knowledge Base.

## Service Interruptions

This topic describes events in the lifecycle of a Kubernetes cluster deployed by VMware Tanzu Kubernetes Grid Integrated Edition that can cause temporary service interruptions.

### Stemcell or Service Update

An operator performs a stemcell version update or Tanzu Kubernetes Grid Integrated Edition version update.

#### Impact

- **Workload:** If you use the recommended configuration, no workload downtime is expected since the VMs are upgraded one at a time. For more information, see [Maintaining Workload Uptime](#).
- **Kubernetes control plane:** The Kubernetes control plane VM is recreated during the upgrade, so `kubectl` and the Kubernetes control plane experience a short downtime.

#### Required Actions

None. If the update deploys successfully, the Kubernetes control plane recovers automatically.

### VM Process Failure on a Cluster Control Plane

A process, such as the scheduler or the Kubernetes API server, crashes on the cluster control plane VM.

#### Impact

- **Workload:** If the scheduler crashes, workloads that are in the process of being rescheduled might experience up to 120 seconds of downtime.
- **Kubernetes control plane:** Depending on the process and what it was doing when it crashed, the Kubernetes control plane might experience 60-120 seconds of downtime. Until the process resumes, the following can occur:
  - Developers might be unable to deploy workloads
  - Metrics or logging might stop
  - Other features might be interrupted

## Required Actions

None. BOSH brings the process back automatically using `monit`. If the process resumes cleanly and without manual intervention, the Kubernetes control plane recovers automatically.

## VM Process Failure on a Cluster Worker

A process, such as Docker or `kube-proxy`, crashes on a cluster worker VM.

### Impact

- **Workload:** If the cluster and workloads follow the recommended configuration for the number of workers, replica sets, and pod anti-affinity rules, workloads should not experience downtime. The Kubernetes scheduler reschedules the affected pods on other workers. For more information, see [Maintaining Workload Uptime](#).

## Required Actions

None. BOSH brings the process back automatically using `monit`. If the process resumes cleanly and without manual intervention, the worker recovers automatically, and the scheduler resumes scheduling new pods on this worker.

## VM Process Failure on the TKG API VM

A process, such as the TKG API server, crashes on the pivotal-container-service VM.

### Impact

- **TKGI control plane:** Depending on the process and what it was doing, the TKG control plane might experience 60-120 seconds of downtime. Until the process resumes, the following can occur:
  - The TKG API or UAA might be inaccessible
  - Use of the TKG CLI is interrupted
  - Metrics or logging might stop
  - Other features might be interrupted

## Required Actions

None. BOSH brings the process back automatically using `monit`. If the process resumes cleanly, the TKGI control plane recovers automatically and the TKGI CLI resumes working.

## VM Failure

An Tanzu Kubernetes Grid Integrated Edition VM fails and goes offline due to either a virtualization problem or a host hardware problem.

### Impact

- **If the BOSH Resurrector is enabled,** BOSH detects the failure, recreates the VM, and reattaches the same persistent disk and IP address. Downtime depends on which VM goes offline, how quickly the BOSH Resurrector notices, and how long it takes the IaaS to create a replacement VM. The BOSH Resurrector usually notices an offline VM within one to two minutes. For more information about the BOSH Resurrector, see the [BOSH documentation](#).
- **If the BOSH Resurrector is not enabled,** some cloud providers, such as vSphere, have similar resurrection or high availability (HA) features. Depending on the VM, the impact can be similar to a key process on that VM going down as described in the previous sections, but the recovery time is longer while the replacement VM is created. See the documentation for process failures in the [cluster worker](#), [cluster control plane](#), and [TKGI API VM](#) sections for more information.

### Required Actions

When the VM comes back online, no further action is required for the developer to continue operations.

## AZ Failure

An availability zone (AZ) goes offline entirely or loses connectivity to other AZs (net split).

### Impact

The control plane and clusters are inaccessible. The extent of the downtime is unknown.

### Required Actions

When the AZ comes back online, the control plane recovers in one of the following ways:

- **If BOSH is in a different AZ,** BOSH recreates the VMs with the last known persistent disks and IPs. If the persistent disks are gone, the disks can be restored from your last backup and reattached. VMware recommends manually checking the state of VMs and databases.
- **If BOSH is in the same AZ,** follow the directions for [region failure](#).

## Region Failure

An entire region fails, bringing all Tanzu Kubernetes Grid Integrated Edition components offline.

### Impact

The entire Tanzu Kubernetes Grid Integrated Edition deployment and all services are unavailable. The extent of the downtime is unknown.

## Required Actions

The TKGI control plane can be restored using BOSH Backup and Restore (BBR). Each cluster might need to be restored manually from backups.

For more information, see [Restore Tanzu Kubernetes Grid Integrated Edition Control Plane](#) in [Restoring Tanzu Kubernetes Grid Integrated Edition](#).

## Troubleshooting Tanzu Kubernetes Grid Integrated Edition Management Console

The following sections describe how to troubleshoot failures to deploy of the VMware Tanzu Kubernetes Grid Integrated Edition Management Console and of Tanzu Kubernetes Grid Integrated Edition instances from the management console.

For information about how to deploy the management console and install Tanzu Kubernetes Grid Integrated Edition, see [Install on vSphere with the Management Console](#).

---

## Deployment of the Tanzu Kubernetes Grid Integrated Edition Management Console Fails

### Problem

Tanzu Kubernetes Grid Integrated Edition Management Console VM fails to deploy from the OVA template.

### Solution

1. Use SSH to log in to the Tanzu Kubernetes Grid Integrated Edition Management Console VM as `root` user.  
Use the password that you specified when you deployed the OVA.
2. Run the following command to obtain the server logs:

```
journalctl -u pks-mgmt-server > server.log
```

3. If the logs do not provide the solution, delete the management console VM from vCenter Server and attempt to deploy it again.

---

## Deployment of Tanzu Kubernetes Grid Integrated Edition from the Management Console Fails

### Problem

Tanzu Kubernetes Grid Integrated Edition fails to deploy from the management console.

### Solution

1. Follow the procedure in [Delete Your Tanzu Kubernetes Grid Integrated Edition Deployment](#)

to cleanly remove all Tanzu Kubernetes Grid Integrated Edition components from vSphere and to clean up related objects in the management console VM.

2. Attempt to deploy Tanzu Kubernetes Grid Integrated Edition again.

## Tanzu Kubernetes Grid Integrated Edition Management Console Cannot Retrieve Cluster Data in a Multi-Tier0 Topology

### Problem

In a deployment to a multiple-tier0 topology, Tanzu Kubernetes Grid Integrated Edition Management Console cannot display cluster information when you go to **TKG Integrated Edition > Clusters** and select a cluster. You see errors of the following type:

```
Failed to retrieve current K8s Cluster summary. cannot get cluster details: cannot get
cluster namespaces: Get https://<address>:8443/api/v1/namespaces: dial tcp <address>:
8443: i/o timeout
Failed to retrieve current K8s Cluster Volumes. cannot get namespaces of cluster 01166
63b-f27b-4026-87e3-cddd01af41f2: Get https://<address>:8443/api/v1/namespaces: dial tc
p <address>:8443: i/o timeout
```

### Cause

In a single tier0 topology, Tanzu Kubernetes Grid Integrated Edition Management Console is deployed to the same infrastructure network as vSphere and NSX-T Data Center. In a multiple-tier0 topology, due to tenant isolation, the infrastructure network is not routable to tenant tier0 uplink networks. In a multiple-tier0 topology, data from the Kubernetes API is exposed by floating IP addresses on tenant tier0 routers. Consequently, the management console cannot retrieve cluster data from the Kubernetes API because it is not on the same network as the tenants.

### Solution

Make sure that the Tanzu Kubernetes Grid Integrated Edition Management Console can connect to tenant floating IP addresses.

1. Connect to the management console VM by using `ssh`.
2. Configure a route on the management console VM. For example, run the following command:

```
route add -net <destination_subnet> gw <gateway_address>
```

- **Destination subnet:** The network CIDR of the tenant floating IP addresses.
- **Gateway:** A VM that can reach the tenant floating IP addresses and the management console.

Because the gateway can reach both the management console and the tenant floating IP addresses, the management console can reach the tenants and retrieve cluster data from the Kubernetes API.

## TKGI MC is Unable to Upgrade the TKGI Control Plane After Migrating from N-VDS to VDS

If you have an existing TKGI installation on a vSphere N-VDS network and migrate your network to

VDS, TKGI MC will no longer be able to upgrade TKGI to a newer version.

## Explanation

TKGI MC uses network resource MOIDs to manage network resources, allowing the Management Console to manage network resources with identical names.

When converting a network from N-VDS to VDS, network resources are assigned new MOIDs.

Although your existing TKGI Kubernetes cluster workloads continue to function, the Management Console configuration maintains stale network resource MOIDs and cannot upgrade the TKGI control plane.

## Workaround

To upgrade TKGI MC after converting a network from N-VDS to VDS:

1. Log in to the VM of your existing TKGI MC installation.
2. To collect your network's current configuration:
  1. Export the `pks-management-server` container's IP as an environment variable:

```
export MGMT_IP=`docker inspect pks-mgmt-server --format='{{.NetworkSettings.Networks.pks.IPAddress}}'`
```

2. Query and save your datacenter list as a json file named `datacenter.json`:

```
curl -u root http://{$MGMT_IP}:8080/api/v1/inventory/vcenter/datacenter
-X GET -k -H \
"Content-type:application/json" > datacenter.json
```

3. Review the exported `datacenter.json` file and note the datacenter MOID.
4. Query and save your network list as a json file named `networks.json`:

```
curl -u root http://{$MGMT_IP}:8080/api/v1/inventory/vcenter/network?dc=
DC-MOID -X GET -k -H \
"Content-type:application/json" > networks.json
```

Where `DC-MOID` is the datacenter MOID you noted in the previous step.

3. To reconfigure TKGI MC with the current network resources MOIDs, do one of the following:
  - ❖ Configure TKGI MC in the TKGI MC YAML Editor, replacing the old MOIDs in your configuration with the current MOIDs.
  - ❖ Manually configure TKGI MC using a manifest file:
    1. Manually create a TKGI MC manifest file with the current network resources MOIDs:

1. Export the TKGI MC manifest as a manifest file named `manifest.json`:

```
curl -u root:Admin\ADMIN-PASSWORD https://localhost/api/v1
/deployment/manifest -X GET -k -H "Content-type:application/json" > manifest.json
```

Where `ADMIN-PASSWORD` is the password for the admin account used in

- the command.
2. Back up the exported manifest file.
  3. Review the `networks.json` file you exported above and note the `moid` and `type` values.
  4. Open the exported manifest file in an editor and change the `dep_network_moid` and `dep_network_type` values to the `moid` and `type` values you collected from the `networks.json` file.
  5. Save the revised manifest file.
2. To restore your TKGI MC's ability to manage the TKGI network resources:
1. Update TKGI MC with the revised manifest:
- ```
curl -u root:ADMIN-PASSWORD http://{$MGMT_IP}:8080/api/v1/deployment -X POST -d @manifest.json -k -H "Content-Type:application/json"
```
- Where `ADMIN-PASSWORD` is the password for the admin account used in the command.
4. To validate your TKGI MC's configuration:
1. Open the TKGI MC UI.
 2. Review **Configuration > Deployment** and confirm that all statuses are either `SUCCESS` or `SKIPPED`.
 3. Review the **TKGI MC Wizard** and confirm the displayed **Network Information** is correct.
5. To upgrade TKGI MC, deploy the TKGI MC OVA for the desired TKGI MC version.
-

Obtain the vRealize Log Insight Agent ID for Tanzu Kubernetes Grid Integrated Edition Management Console

If you enabled integration with VMware vRealize Log Insight, Tanzu Kubernetes Grid Integrated Edition Management Console generates a unique vRealize Log Insight agent ID for the management console VM. You must provide this agent ID to vRealize Log Insight so that it can pull the appropriate logs from the management console.

You obtain the vRealize Log Insight agent ID as follows:

1. Use SSH to log in to the Tanzu Kubernetes Grid Integrated Edition Management Console VM as `root` user.
2. Run the following command to obtain the ID:

```
grep LOGIN_INSIGHT_ID /etc/vmware/environment | cut -d= -f2
```

The resulting ID will be similar to `59debec7-daba-4770-9d21-226ffd743843`.

3. Log in to the vRealize Log Insight Web user interface as administrator and add the agent ID to your list of agents.

Connect to Operations Manager

When you use Tanzu Kubernetes Grid Integrated Edition Management Console to deploy Tanzu Kubernetes Grid Integrated Edition on vSphere, it deploys Operations Manager. The **Deployment Metadata** view of the management console displays the credentials that you need to log in to the deployed Operations Manager instance.

Connect to Operations Manager with SSH

Tanzu Kubernetes Grid Integrated Edition Management Console generates an SSH private key to control SSH access to the Operations Manager VM when you deploy Tanzu Kubernetes Grid Integrated Edition.

1. Go to **Deployment Metadata** in the management console.
2. Click the clipboard icon at the end of the **Ops Manager VM SSH Private Key** row to copy its contents.
3. Paste the contents of the SSH key into a text file, for example `~/tkgi_om.key`.
4. Go to the **Summary** tab of the **TKG Integrated Edition** view in the management console.
5. Copy the **Ops Manager IP Address**.
6. In a terminal run the following command to use SSH to connect to the Operations Manager VM:

```
ssh -i ~/pm_om.key ubuntu@END-POINT
```

Where `END-POINT` is the Ops Manager endpoint address.

Log In to the Operations Manager UI

Tanzu Kubernetes Grid Integrated Edition Management Console generates a random password for the Operations Manager admin account when you deploy Tanzu Kubernetes Grid Integrated Edition.

1. Go to **Deployment Metadata** in the management console.
2. Click the clipboard icon at the end of the **Ops Manager VM Password** row to copy the password.
3. Go to the **Summary** tab of the **TKG Integrated Edition** view in the management console.
4. Click the **Ops Manager IP Address** to open the Operations Manager UI.
5. Log in to Operations Manager with user name `admin` and the password that you copied from the Deployment Metadata view.

Using the BOSH CLI

After you deploy Tanzu Kubernetes Grid Integrated Edition from Tanzu Kubernetes Grid Integrated Edition Management Console on vSphere, you can use the BOSH CLI from both the management console VM and the Ops Manager VM.

Using the BOSH CLI from the Tanzu Kubernetes Grid Integrated Edition Management Console VM

You can use the BOSH CLI from the Tanzu Kubernetes Grid Integrated Edition Management Console VM.

1. In Tanzu Kubernetes Grid Integrated Edition Management Console, go to the **Deployment Metadata** view.
2. Expand the row for **BOSH CLI invocation from console appliance**.
3. Click the clipboard icon at the end of the row to copy the BOSH CLI invocation information.
4. Connect to the management console VM by using SSH.

```
ssh ADDRESS
```

Where `ADDRESS` is the management console VM address.

5. Export the value that you copied from Deployment Metadata view to use BOSH CLI from the management console VM.

```
export BOSH-VALUE
```

Where `BOSH-VALUE` is the BOSH CLI invocation value.

Using BOSH SSH

The management console VM does not support the use of the `bosh ssh` command to connect to the BOSH VM from the management console VM. To connect to the BOSH VM by using `bosh ssh`, you must use the BOSH CLI from the Ops Manager VM.

1. In Tanzu Kubernetes Grid Integrated Edition Management Console, go to the **Deployment Metadata** view.
2. Expand the row for **BOSH CLI invocation from Ops Manager**.
3. Click the clipboard icon at the end of the row to copy the BOSH CLI invocation command.
4. Connect to the Ops Manager VM by using SSH.

For information about how to connect to the Ops Manager VM, see [Connect to Operations Manager with SSH](#).

5. Export the value that you copied from Deployment Metadata view to use BOSH CLI from Ops Manager.

```
export BOSH-VALUE
```

Where `BOSH-VALUE` is the BOSH CLI invocation value.

Solution Guides for Tanzu Kubernetes Grid Integrated Edition

This topic lists the solution guides you can use with VMware Tanzu Kubernetes Grid Integrated Edition.

Solution Guides

- [Solution Guide for Enabling Highly Resilient Kubernetes Workloads Using vSAN Stretched Clusters](#)

Solution Guides for Tanzu Kubernetes Grid Integrated Edition

This topic lists the solution guides you can use with VMware Tanzu Kubernetes Grid Integrated Edition.

Solution Guides

- [Solution Guide for Enabling Highly Resilient Kubernetes Workloads Using vSAN Stretched Clusters](#)

TKGI CLI

This topic describes how to use the VMware Tanzu Kubernetes Grid Integrated Edition Command Line Interface (TKGI CLI) to interact with the TKGI API.

Overview

The [TKGI CLI](#) is a command-line tool to manage Tanzu Kubernetes Grid Integrated Edition provisioned Kubernetes clusters. Use the TKGI CLI to create, manage, and delete Kubernetes clusters.

To deploy workloads to a Kubernetes cluster, use `kubectl`, the Kubernetes CLI.

The [TKGI CLI](#) was previously named the [PKS CLI](#), and both CLIs accept the same commands and arguments.

This version of Tanzu Kubernetes Grid Integrated Edition is compatible with both the TKGI and the PKS CLIs. Enterprise PKS v1.7 and earlier versions are compatible with only the [PKS CLI](#).

If you are using the [PKS CLI](#):

- When using the reference below, substitute `pks` where the commands below use `tkgi`.
- Please consider revising your command line scripts to use the `tkgi` CLI:
 - New commands have been added to the `tkgi` CLI.
 - The `pks` CLI might eventually be deprecated.

TKGI CLI Commands

Current Version: 1.16.0-build.133

tkgi cancel-task

Cancel a task.

```
tkgi cancel-task TASK [flags]
```

Where `TASK` is the ID of the task to cancel.

Synopsis

Cancels a task.

Examples

```
tkgi cancel-task 0941fc83-b254-41a0-a505-14b04919e2cd
```

Options

```
-h, --help    help for cancel-task
```

tkgi certificates

List a Kubernetes cluster's certificates.

```
tkgi certificates CLUSTER-NAME [flags]
```

Where **CLUSTER-NAME** is the name of your cluster.

Synopsis

Lists the certificates for a specific cluster. Requires a target cluster name.

Examples

```
tkgi certificates my-cluster -d 730
```

Options

-d, --days int32	Action flag, Show certificates expire within days (default 180)
-h, --help	help for certificates
--json	Return the TKGI-API output as json
--non-interactive	Don't ask for user input
--wait	Wait for the operation to finish

tkgi cluster

List a cluster's details.

```
tkgi cluster CLUSTER-NAME [flags]
```

Where **CLUSTER-NAME** is the name of your cluster.

Synopsis

Returns the details about a cluster, including name, host, port, ID, number of worker nodes, and last operation.

Examples

```
tkgi cluster my-cluster
```

Options

```
--details  Show details  
-h, --help   help for cluster  
--json     Return the TKGI-API output as json
```

tkgi clusters

List clusters.

```
tkgi clusters [flags]
```

Synopsis

Lists and describes the Kubernetes clusters created using TKGI. Includes the last actions taken on the clusters.

Examples

```
tkgi clusters
```

Options

```
-h, --help   help for clusters  
--json     Return the TKGI-API output as json
```

tkgi compute-profile

Describe a compute profile.

```
tkgi compute-profile PROFILE [flags]
```

Where `PROFILE` is the name of the profile to describe.

Synopsis

Returns the configuration of a saved compute profile.

Examples

```
tkgi computer-profile custom-profile-1
```

Options

```
-h, --help    help for compute-profile
--json      Return the TKGI-API output as json
```

tkgi compute-profiles

List compute profiles.

```
tkgi compute-profiles [flags]
```

Synopsis

Lists and describes compute profiles.

Examples

```
tkgi compute-profiles
```

Options

```
-h, --help    help for compute-profiles
--json      Return the TKGI-API output as json
```

tkgi create-cluster

Create a Kubernetes cluster.

```
tkgi create-cluster CLUSTER-NAME [flags]
```

Where `CLUSTER-NAME` is the name of your cluster.



Note: Use only lowercase characters when naming your cluster if you manage your clusters with Tanzu Mission Control (TMC). Clusters with names that include an uppercase character cannot be attached to TMC.

Synopsis

Creates a Kubernetes cluster. `create-cluster` requires a cluster name, as well as an external hostname and plan. The external hostname can be the loadbalancer from which you access your Kubernetes API (aka, your cluster control plane).

Examples

```
tkgi create-cluster my-cluster --external-hostname example.hostname --plan productio
n
```

Options

<code>--compute-profile</code> string	Optional, compute profile name
<code>--config-file</code> string	Optional, path to the config file, supported form at json/yaml, identified by file extension
<code>-e, --external-hostname</code> string	Address from which to access Kubernetes API
<code>-h, --help</code>	help for create-cluster
<code>--json</code>	Return the TKGI-API output as json
<code>--kubernetes-profile</code> string	Optional, Kubernetes profile name
<code>--network-profile</code> string	Optional, network profile name (NSX-T only)
<code>--node-pool-instances</code> string	Optional, node-pool-instances
<code>--non-interactive</code>	Don't ask for user input
<code>-n, --num-nodes</code> string	Number of worker nodes
<code>-p, --plan</code> string	Preconfigured plans. Run "tkgi plans" for more de tails
<code>--tags []ClusterTag</code>	Optional, Add Tags for VMs as a list of key value pairs (eg. "key1:val1,key2:val2,keyWithoutVal")
<code>--wait</code>	Wait for the operation to finish

tkgi create-compute-profile

Create a compute profile.

```
tkgi create-compute-profile PROFILE-PATH [flags]
```

Where `PROFILE-PATH` is the JSON file describing the compute profile.

Synopsis

Creates a compute profile. Requires a path to the profile JSON file.

Examples

```
tkgi create-compute-profile my-profile.json
```

Options

```
-h, --help    help for create-compute-profile
```

tkgi create-kubernetes-profile

Create a Kubernetes profile.

```
tkgi create-kubernetes-profile PROFILE-PATH [flags]
```

Where `PROFILE-PATH` is the JSON file describing the Kubernetes profile.

Synopsis

Creates a Kubernetes profile. Requires a path to the profile JSON file.

Examples

```
tkgi create-kubernetes-profile my-profile.json
```

Options

```
-h, --help    help for create-kubernetes-profile
```

tkgi create-network-profile

Create a network profile.

```
tkgi create-network-profile PROFILE-PATH [flags]
```

Where `PROFILE-PATH` is the JSON file describing the network profile.

Synopsis

Creates a network profile. Requires a path to the profile JSON file. (Only applicable for NSX-T.)

Examples

```
tkgi create-network-profile my-network-profile.json
```

Options

```
-h, --help      help for create-network-profile
```

tkgi delete-cluster

Delete a Kubernetes cluster.

```
tkgi delete-cluster CLUSTER-NAME [flags]
```

Where **CLUSTER-NAME** is the name of your cluster.

Synopsis

Deletes a Kubernetes cluster. Requires a cluster name.

Examples

```
tkgi delete-cluster my-cluster
```

Options

```
-h, --help      help for delete-cluster
--non-interactive  Don't ask for user input
--wait          Wait for the operation to finish
```

tkgi delete-compute-profile

Delete a compute profile.

```
tkgi delete-compute-profile PROFILE [flags]
```

Where **PROFILE** is the name of the profile to delete.

Synopsis

Deletes a compute profile. Requires a compute profile name. The profile cannot be deleted if it is in use.

Examples

```
tkgi delete-compute-profile my-k8s-profile
```

Options

```
-h, --help           help for delete-kubernetes-profile  
--non-interactive  Don't ask for user input
```

tkgi delete-kubernetes-profile

Delete a Kubernetes profile.

```
tkgi delete-kubernetes-profile PROFILE [flags]
```

Where `PROFILE` is the name of the profile to delete.

Synopsis

Deletes a Kubernetes profile. Requires a Kubernetes profile name. The profile cannot be deleted if it is in use.

Examples

```
tkgi delete-kubernetes-profile my-k8s-profile
```

Options

```
-h, --help           help for delete-kubernetes-profile  
--non-interactive  Don't ask for user input
```

tkgi delete-network-profile

Delete a network profile.

```
tkgi delete-network-profile PROFILE [flags]
```

Where `PROFILE` is the name of the profile to delete.

Synopsis

Deletes a network profile. Requires a network profile name. The profile cannot be deleted if it is in use. Only applicable for NSX-T.

Examples

```
tkgi delete-network-profile my-network-profile
```

Options

```
-h, --help           help for delete-network-profile
--non-interactive  Don't ask for user input
```

tkgi get-credentials

Allows you to connect to a cluster and use kubectl.

```
tkgi get-credentials CLUSTER-NAME [flags]
```

Where `CLUSTER-NAME` is the name of your cluster.

Synopsis

Run this command to update a kubeconfig file so that you can access the cluster through kubectl.

Use the `--sso` flag if the TKGI tile is configured with SAML.

If OIDC is enabled and is not SSO, the password could also be set through environment variable:

`PKS_USER_PASSWORD`.

Examples

```
tkgi get-credentials my-cluster
tkgi get-credentials my-cluster --sso
```

Options

```
-h, --help           help for get-credentials
--sso               Prompt for a one-time passcode to do Single sign-on
--sso-auto          Auto launch local browser to do Single sign-on
--sso-passcode string Single sign-on with one-time passcode
```

tkgi get-kubeconfig

Return the kubeconfig for your username.

```
tkgi get-kubeconfig CLUSTER-NAME -u USER -p PASSWORD -a API [flags]
```

Where:

- **CLUSTER-NAME** is the name of your cluster.
- **USER** is the account name to use for authentication.
- **PASSWORD** is the password to use for authentication.
- **API** is the IP Address for the API.

Synopsis

Run this command in order to get a kubeconfig file so you can access the cluster through kubectl. Typically your kubeconfig will need to be updated based on any new role bindings you have been granted.

Use the `--sso` flag if the TKGI tile is configured with SAML.

Examples

```
tkgi get-kubeconfig my-cluster -u username -p password -a 192.168.1.1
```

```
tkgi get-kubeconfig my-cluster --sso -a 192.168.1.1
```

Options

<code>-a, --api string</code>	API
<code>--ca-cert string</code>	Path to CA Cert for TKGI API
<code>-h, --help</code>	help for get-kubeconfig
<code>-p, --password string</code>	Password
<code>-k, --skip-ssl-validation</code>	Skip SSL Validation
<code>--sso</code>	Prompt for a one-time passcode to do Single sign-on
<code>--sso-auto</code>	Auto launch local browser to do Single sign-on
<code>--sso-passcode string</code>	Single sign-on with one-time passcode
<code>-u, --username string</code>	Username

tkgi kubernetes-profile

View a Kubernetes profile.

```
tkgi kubernetes-profile PROFILE [flags]
```

Where `PROFILE` is the name of the profile.

Synopsis

Lists the details of a saved Kubernetes profile configuration.

Examples

```
tkgi kubernetes-profile custom-profile-1
```

Options

```
-h, --help    help for kubernetes-profile
--json     Return the TKGI-API output as json
```

tkgi kubernetes-profiles

List Kubernetes profiles.

```
tkgi kubernetes-profiles [flags]
```

Synopsis

Lists the details of all saved Kubernetes profile configurations.

Examples

```
tkgi kubernetes-profiles
```

Options

```
-h, --help    help for kubernetes-profiles
--json     Return the TKGI-API output as json
```

tkgi login

Log in to TKGI.

```
tkgi login -u USER -p PASSWORD -a API [flags]
```

Where:

- **USER** is the account name to use for authentication.
- **PASSWORD** is the password to use for authentication.
- **API** is the IP Address for the TKGI API.

Synopsis

The login command requires the following parameters: **-a** to target the IP of your TKGI API, **-u** for

username, and `-p` for password.

Use the `--sso` flag if the TKGI tile is configured with SAML.

Examples

```
tkgi login -a <API> -u <USERNAME> -p <PASSWORD> [--ca-cert <PATH TO CERT> | -k]

tkgi login -a <API> --client-name <CLIENT NAME> --client-secret <CLIENT SECRET> [--ca-cert <PATH TO CERT> | -k]

tkgi login -a <API> --sso [--ca-cert <PATH TO CERT> | -k]

tkgi login -a <API> --sso-auto [--ca-cert <PATH TO CERT> | -k]

tkgi login -a <API> --sso-passcode <sso-passcode> [--ca-cert <PATH TO CERT> | -k]
```

Options

<code>-a, --api string</code>	The TKGI API server URI
<code>--ca-cert string</code>	Path to CA Cert for TKGI API
<code>--client-name string</code>	Client name
<code>--client-secret string</code>	Client secret
<code>-h, --help</code>	help for login
<code>-p, --password string</code>	Password
<code>-k, --skip-ssl-validation</code>	Skip SSL Validation
<code>--skip-ssl-verification</code>	Skip SSL Verification (DEPRECATED: use <code>--skip-ssl-validation</code>)
<code>--sso</code>	Prompt for a one-time passcode to do Single sign-on
<code>--sso-auto</code>	Auto launch local browser to do Single sign-on
<code>--sso-passcode string</code>	Single sign-on with one-time passcode
<code>--timeout int</code>	Timeout with tkgi-api endpoint in seconds (default 300)
<code>)</code>	
<code>-u, --username string</code>	Username

tkgi logout

Log out of TKGI.

```
tkgi logout [flags]
```

Synopsis

Log out of TKGI. Does not remove kubeconfig credentials or kubectl access.

Examples

```
tkgi logout
```

Options

```
-h, --help    help for logout
```

tkgi network-profile

View a network profile.

```
tkgi network-profile PROFILE [flags]
```

Where `PROFILE` is the name of the profile.

Synopsis

Returns the configuration of a saved network profile.

Examples

```
tkgi network-profile large-lb-profile
```

Options

```
-h, --help    help for network-profile
--json      Return the TKGI-API output as json
```

tkgi network-profiles

List network profiles.

```
tkgi network-profiles [flags]
```

Synopsis

Lists and describes all of the network profiles created with TKGI.

Examples

```
tkgi network-profiles
```

Options

```
-h, --help    help for network-profiles
```

```
--json      Return the TKGI-API output as json
```

tkgi plans

List plans.

```
tkgi plans [flags]
```

Synopsis

Lists and describes the available preconfigured plans.

Examples

```
tkgi plans
```

Options

```
-h, --help      help for plans
--json      Return the TKGI-API output as json
```

tkgi promote-cluster-to-policy

(Experimental) Promote a Kubernetes cluster to NSX Policy.

```
tkgi promote-cluster-to-policy CLUSTER-NAME [flags]
```

Where:

- **CLUSTER-NAME** is the name of your cluster.

Synopsis

Promotes a specific cluster to NSX Policy.

Examples

```
tkgi promote-cluster-to-policy my-cluster
```

Options

```
-h, --help      help for promote-cluster-to-policy
```

```
--json           Return the TKGI-API output as json
--non-interactive  Don't ask for user input
--wait          Wait for the operation to finish
```

tkgi resize

Change the number of worker nodes in a cluster.



Note: This command is deprecated as of TKGI v1.12. VMware recommends that you avoid using the `tkgi resize` command to perform resizing operations. Please use `tkgi update-cluster -num-nodes` instead. For more information about the `update-cluster` command, see `tkgi update-cluster` below.

```
tkgi resize CLUSTER-NAME [flags]
```

Where `CLUSTER-NAME` is the name of your cluster.



WARNING: Resize only TKGI clusters that have been upgraded to the current TKGI version.

Synopsis

This command is deprecated as of TKGI v1.12. Please use `tkgi update-cluster` instead.

Resize requires a cluster name, and the number of desired worker nodes. You can scale up clusters to the plan defined maximum number of worker nodes, or scale down clusters to one node.

Examples

```
tkgi resize my-cluster --num-nodes 5
```

Options

<code>-h, --help</code>	help for resize
<code>--json</code>	Return the TKGI-API output as json. Only applicab
le when used with <code>--wait</code> flag	
<code>--node-pool-instances string</code>	Number of instances for each node pool. e.g. node
<code>-pool1:2</code>	
<code>--non-interactive</code>	Don't ask for user input
<code>-n, --num-nodes int32</code>	Number of worker nodes
<code>--tags []ClusterTag</code>	Action flag, Add/Update/Delete Tags for VMs as a
list of key value pairs (eg. <code>--tags "key1:vall,key2:val2,keyWithoutVal"</code>). To delete al	
l tags, pass an empty string (eg. <code>--tags ""</code>)	
<code>--wait</code>	Wait for the operation to finish

tkgi rotate-certificates

Rotate certificates.

```
tkgi rotate-certificates CLUSTER-NAME [flags]
```

Where `CLUSTER-NAME` is the name of your cluster.

Synopsis

Rotates the certificates for a specific cluster. Requires a target cluster name.

Examples

```
tkgi rotate-certificates my-cluster
```

Options

--all	Rotate all certificates belong to one cluster.
-h, --help	help for rotate-certificates
--json	Return the PKS-API output as json
--non-interactive	Don't ask for user input
--only-nsx	Only rotate nsx certificates.
--skip-nsx	Skip nsx certificates when rotating certificates for the cluster.
--wait	Wait for the operation to finish

tkgi task

List a task.

```
tkgi task TASK [flags]
```

Where `TASK` is the ID of the task to describe.

Synopsis

List the status and details of a task.

Examples

```
tkgi task 0941fc83-b254-41a0-a505-14b04919e2cd
```

Options

```
-h, --help      help for task
--json        Return the TKGI-API output as json
```

tkgi tasks

List tasks.

```
tkgi tasks [flags]
```

Synopsis

Lists recent tasks. By default, it lists the ten most recent tasks.

Examples

```
tkgi tasks -l 10
```

Options

```
-h, --help      help for tasks
--json        Return the TKGI-API output as json
-l, --limit int32  Action flag, Show limit number of recent tasks (default 10)
```

tkgi update-cluster

Update a Kubernetes cluster's configuration.

```
tkgi update-cluster CLUSTER-NAME [flags]
```

Where `CLUSTER-NAME` is the name of your cluster.



WARNING: Resize only TKGI clusters that have been upgraded to the current TKGI version.

Synopsis

Updates the configuration of a specific Kubernetes cluster.

Requires a target cluster name and at least 1 valid action flag (e.g. `-num-nodes`). Updates the cluster settings based on the passed flag values. All updated values will persist through cluster upgrades.

Examples

```
tkgi update-cluster my-cluster --num-nodes 5
```

Options

--network-profile string	Action flag, Network profile name
--kubernetes-profile string	Optional, Kubernetes profile name
--compute-profile string	Optional, compute profile name
--num-nodes int32	Action flag, Number of worker nodes You can scale up clusters to the plan defined maximum number of worker nodes, or scale down clusters to one worker node
--kubelet-drain-timeout string	Action flag, The length of time in minutes for drain to wait before giving up.
--kubelet-drain-grace-period string	Action flag, Period of time in seconds given to each pod to terminate gracefully.
--kubelet-drain-force string	Action flag, Force drain even if there are pods not managed by a ReplicationController, ReplicaSet, Job, DaemonSet or StatefulSet.
--kubelet-drain-ignore-daemonsets string	Action flag, Ignore DaemonSet managed pods during drain.
--kubelet-drain-delete-local-data string	Action flag, Drain even if there are pods using emptyDir.
--kubelet-drain-force-node string	Action flag, Forcefully terminate pods which fail to drain. Use it with caution.
--node-pool-instances string	Specify how many instances each node pool should have. Applicable on when the cluster has a compute profile applied to it. e.g. node-pool1:2
--tags []ClusterTag	Action flag, Add/Update/Delete Tags for VMs as a list of key value pairs (eg. --tags "key1:vall,key2:val2,keyWithoutVal"). To delete all tags, pass an empty string (eg. --tags "")
--config-file string	Optional, path to the config file, supported format json/yaml, identified by file extension
--non-interactive	Don't ask for user input
--json	Return the TKGI-API output as json
--wait	Wait for the operation to finish
-h, --help	help for update-cluster

If you are updating a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.

tkgi upgrade-cluster

Upgrade a Kubernetes cluster.

```
tkgi upgrade-cluster CLUSTER-NAME [flags]
```

Where `CLUSTER-NAME` is the name of your cluster.

Synopsis

Upgrades the specified Kubernetes cluster to the current TKGI version. You must provide a single

cluster name.

Examples

```
tkgi upgrade-cluster <one-cluster>
```

Options

<code>-h, --help</code>	help for upgrade-cluster
<code>--json</code>	Return the TKGI-API output as json
<code>--non-interactive</code>	Don't ask for user input
<code>--wait</code>	Wait for the operation to finish



Note: The nodes in an upgrading cluster are processed serially.

If you are upgrading a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.

tkgi upgrade-clusters

Upgrade one or more Kubernetes clusters.

```
tkgi upgrade-clusters --clusters CLUSTER-NAME-1,CLUSTER-NAME-2 [flags]
```

Where:

- `CLUSTER-NAME-1` is the name of a cluster.
- `CLUSTER-NAME-2` is the name of a cluster.

Synopsis

Upgrades one or more Kubernetes clusters to the current TKGI version.

Examples

```
tkgi upgrade-clusters --clusters <cluster-1>,<cluster-2>,<cluster-3> --canaries <cluster-4>,<cluster-5> --max-in-flight 2
```

Options

<code>--canaries</code> string	Optional, list of clusters to be treated as canaries. Will upgrade sequentially before other clusters. Should be a comma separated list of names.
<code>-c, --clusters</code> string	List of clusters to be upgraded. Should be a comma separated list of names.
<code>-h, --help</code>	help for upgrade-clusters.
<code>--json</code>	Return the TKGI-API output as JSON.

```
--max-in-flight int32    Optional, number of clusters to be upgraded in parallel  
(default 1).  
                         The max-in-flight value cannot exceed the Worker VM Max  
in Flight setting defined for your TKGI environment.  
--non-interactive        Don't ask for user input.  
--wait                   Wait for the operation to finish.
```



Note: The nodes in an upgrading cluster are always processed serially.

If you are upgrading a cluster that uses a public cloud CSI driver, see [Limitations on Using a Public Cloud CSI Driver](#) in *Release Notes* for additional requirements.

For more information, see [Upgrade Multiple Clusters](#) in *Upgrading Clusters*.