The science behind the report:

Make Kubernetes containers on Dell EMC PowerEdge R740xd servers easier to manage with VMware Tanzu

This document describes what we tested, how we tested, and what we found. To learn how these facts translate into real-world benefits, read the report Make Kubernetes containers on Dell EMC PowerEdge R740xd servers easier to manage with VMware Tanzu.

We concluded our hands-on testing on November 11, 2020. During testing, we determined the appropriate hardware and software configurations and applied updates as they became available. The results in this report reflect configurations that we finalized on November 11, 2020 or earlier. Unavoidably, these configurations may not represent the latest versions available when this report appears.

System configuration information

Table 1: Detailed information on the system we tested.

Server configuration information	3x Dell EMC™ PowerEdge™ R740xd
BIOS name and version	Dell 2.8.2
Non-default BIOS settings	N/A
Operating system name and version/build number	VMware ESXi™, 7.0.1, 16850804
Power management policy	Performance
Processor	
Number of processors	2
Vendor and model	Intel® Xeon® Gold 6240 CPU @ 2.60GHz Model 85
Core count (per processor)	18
Core frequency (GHz)	2.60
Stepping	7

Server configuration information	3x Dell EMC™ PowerEdge™ R740xd
Memory	
Total memory in system (GB)	192
Number of memory modules	12
Vendor and model	Hynix® HMA82GR7AFR8N-VK
Size (GB)	16
Type	DDR4
Speed (MHz)	2,666
Storage controller	
Vendor and model	Dell PERC H730P Mini
Cache size (MB)	2
Firmware version	25.5.6.0009
Driver version	7.712.51.00
Local storage	
Number of drives	1
Drive vendor and model	Toshiba PX02SMF020
Drive size (GB)	186
Drive information (interface, type)	SAS SSD
Network adapter #1	
Vendor and model	QLogic Corporation QLogic 57800 10 Gigabit Ethernet
Number and type of ports	4 x 10GbE
Network adapter #2	
Vendor and model	Broadcom P225p NetXtreme dual port 10/25 Gigabit Ethernet
Number and type of ports	2 x 10/25GbE
Cooling fans	
Vendor and model	Nidec® UltraFlo 4VXP3-X30
Number of cooling fans	6
Power supplies	
Vendor and model	Dell 0PJMDN
Number of power supplies	2
Wattage of each (W)	750

How we tested

Installing VMware vSphere® 7.0 Update 1 on a Dell EMC PowerEdge R740xd

1. Download the Dell EMC Custom Image for VMware ESXi™ 7.0 U1 from the following link: https://my.vmware.com/en/group/vmware/downloads/details?downloadGroup=OEM-ESXI70U1-DELLEMC&productId=974.
2. Open a new browser tab, and connect to the IP address of the Dell EMC PowerEdge R740xd server iDRAC.
3. Log in with the iDRAC credentials (login and password). We used `root` / `calvin`.
4. In the lower left of the screen, click Launch Virtual Console.
5. On the console menu bar in the console, click Connect Virtual Media.
6. Under Map CD/DVD, click Browse…, and select the image you downloaded in step 1. Click Open.
7. Click Map Device. Click Close.
8. On the console menu bar, click Boot. Select Virtual CD/DVD/ISO, and click Yes.
9. On the console menu bar, click Power. Select Power On System, and click Yes.
10. The system will boot to the mounted image and the Loading ESXi installer screen will appear. When prompted, press Enter.
11. To accept the EULA and continue, press F11.
12. Select the storage device to target for installation. We selected the internal SD card. Press Enter.
13. To confirm the storage target, press Enter.
14. Select the keyboard layout, and press Enter.
15. Provide a root password, and confirm the password. Press Enter.
16. To install, press F11.
17. To reboot the server, press Enter.

Installing VMware vCenter Server® Appliance™ 7.0 Update 1

1. Download the VMware vCenter® 7.0 Update 1 from the VMware support portal: https://my.vmware.com.
2. Mount the image on your local system, and browse to the `vcsa-ui-installer` folder. Expand the folder, and launch the installer if it doesn't do so automatically.
3. After the vCenter Server Installer wizard opens, click install.
4. To begin installation of the new vCenter Server Appliance, click Next.
5. Check the box to accept the license agreement, and click Next.
6. Enter the IP address of one of your newly deployed Dell EMC PowerEdge R740xd servers with ESXi 7.0 Update 1. Enter the root password, and click Next.
7. To accept the SHA1 thumbprint of the server's certificate, click Yes.
8. Accept the VM name, and enter and confirm the root password for the VCSA. Click Next.
9. Select the size of the environment you're planning to deploy. We selected medium. Click Next.
10. Select the datastore on which to install the vCenter Server Appliance. Accept the datastore defaults, and click Next.
11. Enter the FQDN, IP address information, and DNS servers you want to use for the vCenter Server Appliance. Click Next.
12. To begin deployment, click Finish.
13. Once Stage 1 completes, click Close, and click Yes.
14. Open a browser window, and navigate to https://[vcenter.FQDN:5480/.
15. Click Set up on the Getting Started - vCenter Server.
16. Enter the root password, and click Log in.
17. Click Next.
18. Enable SSH access, and click Next.
19. To confirm the changes, click OK.
20. For the Single Sign-On domain name, type `vsphere.local`. Enter a password for the administrator account, confirm it, and click Next.
21. Click Next.
22. Click Finish.

Creating a cluster in VMware vSphere 7.0 Update 1

1. Open a browser, and enter the address of the VMware vCenter Server you deployed (https://[vcenter.FQDN]/ui).
2. Select the vCenter Server in the left panel, right-click, and select New Datacenter.
3. Enter a name for the new datacenter, and click OK.
4. Select the datacenter you just created, right-click, and select New Cluster.
5. Give a name to the cluster, and enable vSphere DRS. Click OK.
6. Under Add hosts in the cluster configuration panel, click Add.
7. Check the box for Use the same credentials for all hosts. Enter the IP Address and root credentials for the first host, and the IP addresses of all remaining hosts. Click Next.
8. To select all hosts, check the box for Hostname/IP Address. Click Ok.
9. Click Next.
10. Click Finish.

Configuring VMware Tanzu Kubernetes

This section presents the steps for setting up our environment prerequisites and deploying VMware® Tanzu™ Kubernetes in VMware vSphere 7.0 Update 1. Our environment contained a three-node vSphere cluster running VMware ESXi 7.0 Update 1 and VMware vCenter 7.0 Update 1 and sharing datastores housed on generic NFS storage. We configured upstream networking switches to accommodate the VLANs and routing for the environment.

Defining networks and IP addresses (prerequisite)

Management network (VM network)

This network can interact with all vSphere components and ESXi hosts.

We used our VM network on a standard (not Distributed) vSwitch. We used a 10.206.0.0/16 network scope with a gateway of 10.206.0.1. We did not assign VLAN tagging for this network (all ports were "untagged" or access).

Workload network

We isolated this network behind a NAT gateway and used private addresses for all connectivity. We defined the Workload Network portgroup on a Distributed vSwitch, which Tanzu Kubernetes deployment requires. We used a 192.168.1.0/24 network scope for the workload network with a NAT gateway of 192.168.1.1. We assigned the network as VLAN 11.

Note: We recommend ensuring the gateway functions properly. Deployments without gateways might appear to complete successfully but will not function correctly after deployment.

Required network addresses

1. HAProxy management address: 10.206.201.200/16, gateway 10.206.0.1
2. HAProxy workload address: 192.168.1.2/24, gateway 192.168.1.1
3. Cluster management addresses 10.206.201.201 through 10.206.201.205
   - The Supervisor Control Plane VMs use these addresses to manage the namespace.
   - These must be five contiguous IP addresses.
     - You will only provide the starting IP of 10.206.201.201, which the shared cluster IP use.
     - The next three addresses will be individual VM IP addresses.
     - One will be for upgrade purposes.
4. Worker VM addresses: 192.168.1.64/26 (192.168.1.65 - 192.168.1.126)
   - The workload network uses these addresses to provision deployed services, and the Supervisor Control Plane VMs use them to provide connectivity to those services.
   - In this case, the CIDR notation defines a scope of addresses. When inputting the address range, exclude the first (.64) and last (.127) addresses.
5. Load balancer addresses: 192.168.1.240/29 (192.168.1.240 - 192.168.1.247)
   - The HAProxy on the workload network uses these addresses to provide load balancing for deployed services. These are separate from the HAProxy management address and must not overlap any other network or VM (especially gateways) because the HAProxy will bound and respond to all the addresses.
   - In this case, the CIDR notation includes the first and last addresses. You will need to provide all eight addresses to the wizard.

Creating a Distributed vSwitch and workload port group

1. From the vSphere client, click Home→Networking.
2. Select your datastore, and in the Actions pull-down menu on the right panel, select Distributed vSwitch→New Distributed vSwitch.

3. Either enter a name for your vSwitch or accept the default. Click Next.
4. Select 7.0.0 - ESXi 7.0 and later as the version, and click Next.
5. Select the number of ESXi hosts you'll give the vSwitch. We selected one. Click Next.
6. Click Finish.
7. Right-click the new Distributed vSwitch, and select Add and Manage Hosts.
8. Leave Add hosts selected, and click Next.
9. Click the + to add new hosts.
10. To select all the hosts in your target cluster, check the box beside "Host." Click OK, and click Next.
11. Select the NIC for this Distributed vSwitch, and click Assign Uplink.
12. Accept the defaults at the top of the panel, but check "Apply this uplink assignment to the rest of the hosts." Click OK, and click Next.
13. Do not assign vmkernel adapters at this time. Click Next.
14. Do not migrate any VM networking at this time. Click Next.
15. Click Finish.
16. Right-click the Distributed vSwitch, and select Distributed Port Group→New Distributed Port Group.
17. Type the name `Workload Network`, and click Next.
18. Change the VLAN type to VLAN, and set the VLAN ID to VLAN 11. Click Next.
19. Click Finish.

Creating a DevOps user

Create a dedicated vSphere user account to access Kubernetes namespaces.

1. From the vSphere client, click Home→Administration.
2. In the left panel, click Users and Groups.
3. In the right panel, click Users, select the vsphere.local domain, and click Add.
4. Enter a username and password, and click Add.
5. For simplicity, we added the DevOps user to a group with Administrator privileges. Click Groups, and select the Administrators group. Click Edit.
6. Under Add Members, search for the new DevOps user, and add them to the administrators group. Click Save.

Creating the HAProxy content library

1. Click the following link to download the vSphere compatible HAProxy ovf (v0.1.8): https://github.com/haproxytech/vmware-haproxy.
2. In left menu pane of the vSphere client, click Content Libraries.
3. In the Content Libraries panel on the right, click Create.
4. Name the content library `HAProxy-cl`, and click Next.
5. Accept the default, and click Next.
6. Choose the storage location for the content library, and click Next.
7. Review the settings, and click Finish.
8. Click the newly created HAProxy-cl content library.
9. In the upper portion of the right-side panel for HAProxy-cl, click the Actions pull-down menu, and select Import Item.
10. Change the selection to local file, and click Upload files.
11. Navigate to the location of the ovf file you downloaded in step 1, and click Open.
12. Click Import.

Creating the Tanzu Kubernetes Grid (TKG) content library

1. In left menu pane of the vSphere client, click Content Libraries.
2. In the Content Libraries panel on the right, click Create.
3. Name the content library `TKG-cl`, and click Next.
4. Select Subscribed content library, and use https://wp-content.vmware.com/v2/latest/lib.json for the subscription URL. Click Next.
5. Click Yes.
6. Choose the storage location for the content library, and click Next.
7. Review the settings, and click Finish.

Creating a storage tag

1. From the vSphere client, select Menu→Storage.
2. From the left pane, select the storage to tag for Tanzu.
3. Under the Summary tab, locate the Tags panel, and click Assign.
4. Click Add Tag.
5. Name the tag `Tanzu`. Click Create New Category.
6. Name the category `Tanzu Storage`. Clear all object types except Datastore, and click Create.
7. To select Tanzu Storage, use the Category pull-down menu, and click Create.
8. Check the box for the newly created tag, and click Assign.

Creating VM Storage policies

1. From the vSphere client, click Menu→Policies and Profiles.
2. On the left panel, click VM Storage Policies.
3. Click Create.
4. Create a new VM Storage policy, name it `TKG-cl`, and click Next.
5. Check the box to Enable tag-based placement rules, and click Next.
6. To select the Tanzu Storage policy you created, use the Tag Category pull-down menu. Click Browse Tags.
7. Select Tanzu, and click OK.
8. Click Next.
9. To make sure your storage target is marked as compatible, review the compatible storage, and click Next.
10. Click Finish.

Deploying the HAProxy

1. From the vSphere client, click Menu→Content Libraries.
2. Click the HAProxy-cl library.
3. In the left panel, click OVF & OVA Templates, and right-click the HAProxy template that appears. Select New VM from This Template….
4. Enter a simple name (we used `HAProxy`), and select the data center or folder to which you want to deploy. Click Next.
5. Select the cluster or compute resource where you want to deploy the HAProxy VM, and click Next.
6. Review the details, and click Next.
7. Select I accept all license agreements, and click Next.
8. Accept the default configuration, and click Next.
9. Select the target storage for the VM, and click Next.
10. Select VM Network for the management network, and choose a network for the workload network. Choose the same network for the front-end network, and click Next.

11. Customize the template with the following:
    a. Appliance configuration
        i. For the root password, we used `Password1!`.
        ii. Select Permit Root Login.
        iii. Leave the TLS CA blank.
    b. Network configuration
        i. We left the default `haproxy.local`.
        ii. Configure a local DNS server.
        iii. For management IP, we used `10.206.201.200/16`.
        iv. For management gateway, we used `10.206.0.1`. Note: The description will ask for the workload network gateway address, but enter the management gateway address instead.
        v. For Workload IP, we used `192.168.1.2/24`.
        vi. For Workload gateway, we used `192.168.1.1`.
    c. Load balancing
        i. For load balancer IP ranges, we used `192.168.1.240/29`.
        ii. Accept the default management port.
        iii. For HAProxy User ID, we used `admin`.
        iv. For the HAProxy password, we used `Password1!`.
    d. Click Next.
12. Review the summary, and click Finish. The deployment will take a few minutes to complete.
13. Power on the HAProxy VM.

Configuring workload management

This task will configure the vSphere environment for Kubernetes.

1. From the vSphere client, click Menu→Workload Management.
2. Click Get Started.
3. Review the messages and warnings regarding supported configurations. Click Next.
4. Select the Cluster on which you want to enable workload management, and click Next.
5. Choose the capacity for the control plane VMs. We chose Small. Click Next.
6. Choose the storage policy for the control plane nodes. We chose tkg-clusters. Click Next.
7. Configure the load balancer section with the following:
    - Name: `haproxy`
    - Type: `HAProxy`
    - Data plane API Addresses: `10.206.201.200:5556`
    - User name: `admin`
    - Password: `Password1!`
    - IP Address Ranges for Virtual Servers: `192.168.1.240-192.168.1.247`
    - Server Certificate Authority: [copy and pasted from the instructions below]
        - Open an SSH session to the HAProxy management address, and connect using `root` and `Password1!`.
        - Type the following: `cat /etc/haproxy/ca.crt`
        - Copy the entire output (including the first and last lines), and paste the contents into the Server Certificate Authority box for the Server Certificate Authority.
        - Close the SSH session.
8. Click Next
9. Configure workload management with the following:
    - Network: `VM Network`
    - Starting IP Address: `10.206.201.201`
    - Subnet Mask: `255.255.0.0`
    - Gateway: `10.206.0.1`
    - DNS Server: `10.41.0.10`
    - NTP Server: `10.40.0.1`
10. Click Next.

11. Configure the workload network with the following:
    a. For Services addresses, leave the default.
        i. DNS Servers: `10.41.0.10`
    b. Under Workload Network, click Add.
    c. Accept the default for network-1.
        i. Port Group: `Workload Network`
        ii. Gateway: `192.168.1.1.`
        iii. Subnet: `255.255.255.0`
        iv. IP Address Ranges: `192.168.1.65-192.168.1.126`
    d. Click Save.
12. For TKG configuration, use the following:
    a. Beside Add Content Library, click Add.
    b. Select the TKG-cl library, and click OK.
    c. Click Next.
13. Click Finish. The workload management cluster will deploy and configure. If you see apparent errors during configuration, they will resolve upon successful completion.

Configuring namespaces

1. In Workload Management, click Namespaces.
2. Click Create Namespace.
3. Select the target cluster, and enter a name. We used `tanzu-ns`. Click Create.
4. After Tanzu creates the new namespace, click the Permissions tab, and click Add.
5. For the identity source, choose vSphere.local. Search for the DevOps user you created. Select the "can edit" role. Click OK.
6. Click the Storage tab.
7. In the Storage Policies section, click Edit.
8. Select the tkg-clusters policy, and click Ok. The environment is ready for connection and deployment of containers.

Creating a bootstrap Ubuntu 18.04.5 VM for Tanzu Kubernetes Grid CLI

1. Log into vCenter, and from the dropdown menu, click Storage.
2. Select datastore1, and click Files.
3. Click Upload Files, and upload the Ubuntu 18.04.5 ISO image.
4. Right-click the cluster, and click New Virtual Machine.
5. Click Next.
6. Enter a name for the VM, and click next.
7. Click Next.
8. Select datastore1, and click Next.
9. Click Next.
10. Select Linux from the Guest OS Family dropdown menu, select Ubuntu Linux (64 bit) from the guest OS version dropdown menu, and click Next.
11. Assign the VM two vCPUs, 6GB of memory, and a 16GB hard disk.
12. Select Datastore ISO File from the New CD/DVD Drive dropdown menu, and select the Ubuntu ISO uploaded to the datastore previously. Ensure Connect At Power On is checked, and click Next.
13. Click Finish.
14. Power on the VM, and click Launch Remote Console.
15. Click Install Ubuntu.
16. Click Continue.
17. Select Minimal installation, and click Continue.
18. Click Install now.
19. Click Continue.
20. Click Continue.
21. Enter your desired full name, computer name, username, and password, and click Continue.
22. Click Restart Now.
23. Press Enter.
24. Enter your password, and click Sign In.
25. When prompted by the Software Updater, install OS and software updates.

26. Click Restart Later, and power off the VM.
27. Right-click the VM in vCenter, and click Edit Settings.
28. Click Add New Device, and select Network Adapter.
29. From the New Network dropdown menu, select Browse.
30. Click workload network, and click OK.
31. Click OK.
32. Power on the VM, and click Launch Remote Console.
33. Enter your password, and click Sign In.
34. In the top right, click the network icon, and click Ethernet (ens192).
35. Click Wired Settings.
36. Click the gear icon next to Ethernet (ens192).
37. Click IPv4.
38. Change IPv4 Method to Manual, and enter the following settings:
    - Address: `192.168.1.5`
    - Netmask: `255.255.255.0`
39. Click Apply, and close the settings window.

Installing kubectl on the bootstrap VM

1. On the VM, open a terminal.
2. Install curl:

```
sudo apt-get install curl
```

3. When prompted to verify the install, type `y`.
4. Install kubectl:

```
curl -LO https://storage.googleapis.com/kubernetes-release/release/`curl -s https://storage.
googleapis.com/kubernetes-release/release/stable.txt`/bin/linux/amd64/kubectl
```

5. Make kubectl executable:

```
chmod +x ./kubectl
```

6. Move kubectl to your PATH:

```
sudo mv ./kubectl /usr/local/bin/kubectl
```

Installing docker on the bootstrap VM

1. On the VM, open a terminal.
2. Install prerequisite packages:

```
sudo apt-get install apt-transport-https ca-certificates gnupg-agent software-properties-common
```

3. When prompted to install, type `y`.
4. Add the official Docker GPG key:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

5. Add the Docker repository:

```
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable"
```

6. Update the package list, and install docker:

```
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io
```

Installing Tanzu Kubernetes Grid 1.2.0 CLI on the bootstrap VM

1. Open a browser on the VM, and navigate to https://www.vmware.com/go/get-tkg.
2. Click Go to downloads.
3. Next to VMware Tanzu Kubernetes Grid CLI for Linux, click Download Now.
4. Log in with your VMware credentials.
5. Scroll to the bottom of the EULA window, and check the box to agree to the EULA.
6. Click accept.
7. Open a terminal, and navigate to the Downloads folder:

```
cd ~/Downloads
```

8. Unzip the archive:

```
tar -xvf tkg-linux-amd64-v1.2.0-vmware.1.tar.gz
```

9. Make the TKG binary executable, and add it to PATH:

```
sudo mv tkg/tkg-linux-amd64-v1.2.0+vmware.1 /usr/local/bin/tkg
chmod +x /usr/local/bin/tkg
```

10. Initialize TKG CLI for the first time:

```
tkg get management-cluster
```

Installing vSphere for kubectl plugin

1. Open a browser on the VM, and navigate to the IP of one of the three Supervisor Control Plane VMs. In our environment, the first control plane VM was at 10.206.201.204.
2. Click Advanced, and to bypass the certificate warning, click Accept the Risk and Continue.
3. Click Download CLI Plugin Linux.
4. Select Save File, and click OK.
5. Open the Files app, and navigate to the Downloads folder.
6. Right-click `vsphere-plugin.zip`, and select Extract Here.
7. Open a terminal, and navigate to the vsphere-plugin binary within the extracted folder:

```
cd Downloads/vsphere-plugin/bin
```

8. Make the vsphere-plugin binary executable, and add it to PATH:

```
sudo mv kubectl-vsphere /usr/local/bin
chmod +x /usr/local/bin/kubectl-vsphere
```

Using the Tanzu Kubernetes Grid CLI to deploy a WordPress workload to vSphere with Kubernetes

1. On the VM, open a terminal.
2. Create an SSH key pair to authenticate with Tanzu Kubernetes Grid:

```
ssh-keygen -t rsa -b 4096 -C administrator@vsphere.local
```

3. Press Enter, then enter your desired password twice to confirm. Add the private key to the VM's SSH agent:

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

4. Enter the password created in step 1.
5. Log into the cluster with the kubectl vSphere plugin by connecting to the management cluster IP:

```
kubectl vsphere login --insecure-skip-tls-verify --vsphere-username administrator@vsphere.local
--server=https://10.206.201.201
```

6. When prompted, enter your password, and press Enter.
7. Switch to the tanzu-ns namespace context, and add the management cluster:

```
kubectl config use-context tanzu-ns
tkg add management-cluster
```

8. Find the management cluster IP:

```
tkg get management-cluster
```

9. Set the management cluster to the IP from step 8:

```
tkg set management-cluster 192.168.1.240
```

   a. Open the TKG config file by running `sudo nano ~/.tkg/config`, and pasting the following:

```
CONTROL_PLANE_STORAGE_CLASS=tkg-clusters
WORKER_STORAGE_CLASS=tkg-clusters
DEFAULT_STORAGE_CLASS=tkg-clusters
STORAGE_CLASSES=""
SERVICE_DOMAIN=new-tkg-cluster.local
CONTROL_PLANE_VM_CLASS=guaranteed-small
WORKER_VM_CLASS=guaranteed-small
SERVICE_CIDR=100.64.0.0/13
CLUSTER_CIDR=100.96.0.0/11
```

10. To save and quit, press Ctrl+X. When prompted, type `Y`.
11. Create a new YAML file to define the new cluster configuration:

```
sudo nano new-tkg-cluster.yaml
```

   a. Paste the following into the file:

```
apiVersion: run.tanzu.vmware.com/v1alpha1
kind: TanzuKubernetesCluster
metadata:
  name: new-tkg-cluster
spec:
  topology:
    controlPlane:
      count: 1
      class: guaranteed-small
      storageClass: tkg-clusters
    workers:
      count: 3
      class: guaranteed-small
      storageClass: tkg-clusters
  distribution:
    version: v1.18.5
  settings:
    network:
      cni:
        name: calico
      services:
        cidrBlocks: ["172.16.0.0/24"]
      pods:
        cidrBlocks: ["172.16.1.0/24"]
```

12. Press Ctrl+X, and to save, type `Y`.
13. Create the cluster:

```
kubectl apply -f new-tkg-cluster.yaml
```

14. Log out:

```
kubectl vsphere logout
```

15. Once it has fully deployed, log back into the new cluster's context:

```
kubectl vsphere login --insecure-skip-tls-verify --vsphere-username administrator@vsphere.local
--server=https://10.206.201.201 --tanzu
kubernetes-cluster-name new-tkg-cluster --tanzu-kubernetes-cluster-namespace tanzu-ns
```

16. Enter your password, and when prompted, press Enter.
17. Set the context to the new cluster:

```
kubectl config use-context new-tkg-cluster
```

18. Create a new directory from which to deploy WordPress, and enter it:

```
mkdir wp
cd wp
```

19. In the previously created directory, use a text editor to create a new file called `kustomization.yaml` with the following contents:

```
secretGenerator:
- name: mysql-pass
  literals:
  - password=Password1
resources:
  - wordpress-rb.yaml
  - mysql-deployment.yaml
  - wordpress-deployment.yaml
```

20. In the previously created directory, use a text editor to create a new file called `wordpress-deployment.yaml` with the following contents:

```
apiVersion: v1
kind: Service
metadata:
  name: wordpress
  labels:
    app: wordpress
spec:
  ports:
    - port: 80
  selector:
    app: wordpress
    tier: frontend
  type: LoadBalancer
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: wp-pv-claim
  labels:
    app: wordpress
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: tkg-clusters
  resources:
    requests:
      storage: 20Gi
---
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
kind: Deployment
metadata:
  name: wordpress
  labels:
    app: wordpress
spec:
  selector:
    matchLabels:
      app: wordpress
      tier: frontend
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: wordpress
        tier: frontend
    spec:
      containers:
      - image: wordpress:4.8-apache
        name: wordpress
        env:
        - name: WORDPRESS_DB_HOST
          value: wordpress-mysql
        - name: WORDPRESS_DB_PASSWORD
```

```
          valueFrom:
            secretKeyRef:
              name: mysql-pass
              key: password
        ports:
        - containerPort: 80
          name: wordpress
        volumeMounts:
        - name: wordpress-persistent-storage
          mountPath: /var/www/html
      volumes:
      - name: wordpress-persistent-storage
        persistentVolumeClaim:
          claimName: wp-pv-claim
```

21. In the previously created directory, use a text editor to create a new file called `mysql-deployment.yaml` with the following contents:

```
apiVersion: v1
kind: Service
metadata:
  name: wordpress-mysql
  labels:
    app: wordpress
spec:
  ports:
    - port: 3306
  selector:
    app: wordpress
    tier: mysql
  clusterIP: None
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-pv-claim
  labels:
    app: wordpress
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: tkg-clusters
  resources:
    requests:
      storage: 20Gi
---
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
kind: Deployment
metadata:
  name: wordpress-mysql
  labels:
    app: wordpress
spec:
  selector:
    matchLabels:
      app: wordpress
      tier: mysql
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: wordpress
        tier: mysql
    spec:
      containers:
      - image: mysql:5.6
        name: mysql
```

```
    env:
    - name: MYSQL_ROOT_PASSWORD
      valueFrom:
        secretKeyRef:
          name: mysql-pass
          key: password
    ports:
    - containerPort: 3306
      name: mysql
    volumeMounts:
    - name: mysql-persistent-storage
      mountPath: /var/lib/mysql
  volumes:
  - name: mysql-persistent-storage
    persistentVolumeClaim:
      claimName: mysql-pv-claim
```

22. In the previously created directory, use a text editor to create a new file in the directory called `wordpress-rb.yaml` with the following contents:

```
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: rolebinding-default-privileged-sa-ns_default
roleRef:
  kind: ClusterRole
  name: psp:vmware-system-privileged
  apiGroup: rbac.authorization.k8s.io
subjects:
- kind: Group
  apiGroup: rbac.authorization.k8s.io
  name: system:serviceaccounts
```

23. Deploy the workload to the cluster:

```
kubectl apply -k ./
```

24. Find the external IP address of the newly created WordPress LoadBalancer service:

```
kubectl get services
```

25. Open a web browser, and navigate to the External IP listed by the command in step 24. You will see the initial language selection screen of a fresh WordPress installation.

This project was commissioned by Dell EMC.