The science behind the report:

Prepare images in Kubernetes for machine learning faster with a Dell EMC cluster powered by AMD EPYC 7543 processors

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 Prepare images in Kubernetes for machine learning faster with a Dell EMC cluster powered by AMD EPYC 7543 processors.

We concluded our hands-on testing on April 26, 2021. 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 April 23, 2021 or earlier. Unavoidably, these configurations may not represent the latest versions available when this report appears.

Our results

Table 1: Results of our testing

	Dell EMC™ PowerEdge™ R7525 cluster with AMD EPYC™ 7532 processors	Dell EMC PowerEdge R7525 cluster with AMD EPYC 7543 processors
Time to complete the workload (seconds)	2,557	2,201
Percentage less time	-	13.92%
Frames per second (FPS)	8,263	9,349
Percentage more FPS	-	13.14%
Average CPU utilization during testing	70.1%	69.7%

Table 2: Storage performance metrics from VMware® ESXTOP data

	Dell EMC PowerEdge R7525 cluster with AMD EPYC 7532 processors	Dell EMC PowerEdge R7525 cluster with AMD EPYC 7543 processors
Average read latency (ms)	0.12	0.12
Average write latency (ms)	0.32	0.3
Average read input/output operations per second (IOPS)	12	25
Average write IOPS	12,816	13,439
Average read bandwidth (MB/s)	0.43	0.52
Average write bandwidth (MB/s)	51	53

CPU utilization charts

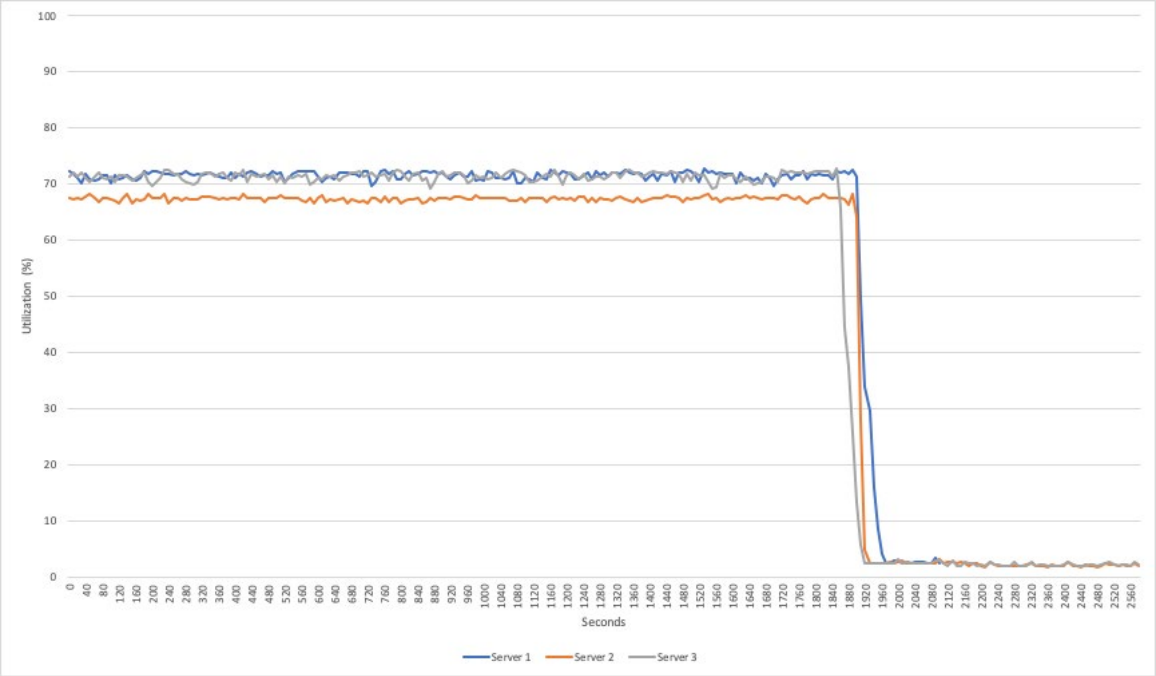

Figure 1: Average server CPU usage for individual Dell EMC PowerEdge R7525 servers with AMD EPYC 7532 processors. Source: Principled Technologies.

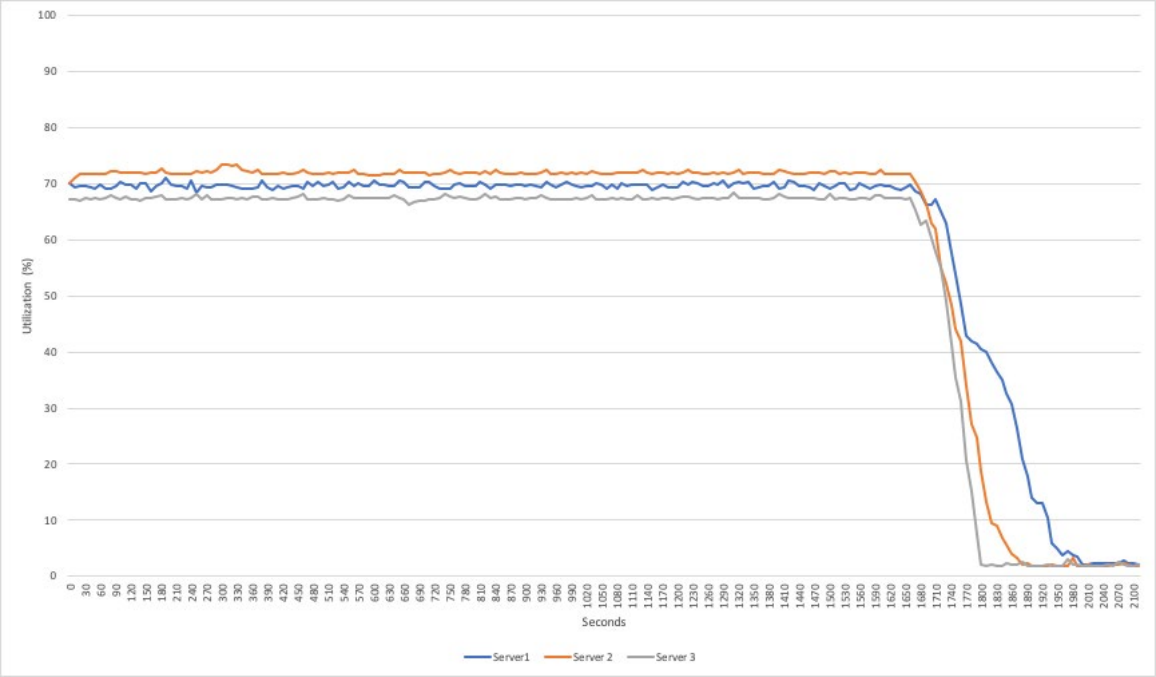

Figure 2: Average server CPU usage for individual Dell EMC PowerEdge R7525 servers with AMD EPYC 7543 processors. Source: Principled Technologies.

System configuration information

Table 3: Detailed information on the servers we tested

Server configuration information	3 x Dell EMC PowerEdge R7525	
BIOS name and version	Dell 2.0.3	
Non-default BIOS settings	N/A	
Operating system name and version/build number	VMware ESXi™, 7.0.2, 17630552	
Date of last OS updates/patches applied	2/8/21	
Power management policy	Maximum Performance	
Processor		
Number of processors	2	
Vendor and model	AMD EPYC 7543	AMD EPYC 7532
Core count (per processor)	32	32
Core frequency (GHz)	2.80	2.40
Memory module(s)		
Total memory in system (GB)	512	
Number of memory modules	16	
Vendor and model	Hynix HMA84GR7CJR4N-XN	
Size (GB)	32	
Type	PC4-3200	
Speed (MHz)	3,200	
Speed running in the server (MHz)	3,200	
Storage controller		
Vendor and model	Dell BOSS-S1 Adapter	
Firmware version	2.5.13.3024	
Local storage		
Number of drives	2	
Drive vendor and model	Micron MTFDDAV480TDS	
Drive information (speed, interface, type)	6Gbps SATA M.2 SSD	
Drive size (GB)	480	
Network adapter		
Vendor and model	Broadcom® BCM5720	
Number and type of ports	2 x 1Gb Ethernet Adapter	
Driver version	21.60.16	

Server configuration information	3 x Dell EMC PowerEdge R7525
Fibre Channel adapter	
Vendor and model	Emulex LightPulse
Number of type of ports	2 x 16Gb/s FC
Firmware version	12.6.281.13
Cooling fans	
Vendor and model	Foxconn PIE060M12M
Number of cooling fans	6
Power supplies	
Vendor and model	Dell 064JDMA01
Number of power supplies	2
Wattage of each (W)	1,400

Table 4: Detailed information on the storage appliance we tested

Storage configuration information	Dell EMC PowerStore 5000T
Drive vendor and model	Samsung® MZ-WLL1T9C
Number of storage controllers	2
Controller firmware version	1.0.2.1.3.273
Number of storage shelves	1
Number of drives per shelf	21
Drive size (TB)	1.92
Drive type	NVMe™ SSD
Number of LUNs	1
LUN size (TB)	2

How we tested

We installed and configured the latest available version of VMware vSphere® 7.0 Update 2 on three Dell EMC PowerEdge R7525 servers. We installed the hypervisor on internal SATA M.2 SSDs on a BOSS controller card. We configured and created a 2TB volume on a Dell EMC PowerStore 5000T storage system, and then mapped the volume to all three PowerEdge R7525 servers as a shared datastore for Tanzu Kubernetes Grid (TKG) deployment.

We used a Dell Networking 1Gb X1052 switch for the VM network, vMotion, and Management Network for Tanzu. We used a 1Gb switch for Workload Network for Tanzu. We isolated the Workload Network behind a NAT gateway and utilized private addresses for all connectivity. We configured the Workload Network port group on a Distributed vSwitch, which is required for Tanzu Kubernetes deployment. We connected the vCenter Server® and ESXi hosts to the Management Network, and we connected the TKG workload clusters to the Workload Network. The Tanzu Supervisor Cluster, Load Balancer, NAT gateway, and TKG CLI VM were dual-networked with a virtual NIC attached to both the Management Network and the Workload Network. We deployed the four infrastructure VMs (vCenter, Load Balancer, NAT gateway, and TKG CLI VM) on a PowerEdge R6525 server. We deployed a TKG cluster with 44 guaranteed-large worker nodes (each worker node had 4 CPUs and 16 GB of memory fully reserved). We then deployed a pod on each node with 3 CPU threads and 10 GB of memory reserved.

We compared the servers using a purpose-built preprocessing workload we wrote in Python using publicly available open-source libraries. We define this workload in the Workload description section. We can provide the code on request. Please contact info@principledtechnologies.com for more information.

Workload description

Our preparation workload emulates a simple image-processing workload by distributing dataset preparation tasks among M processes running on N nodes, the exact number of which is up to the user. Each process produces a single shard of the final data set by taking input images, performing simple conversions, encoding the resulting image, and appending it to the shard file.

This Python workload uses Pillow (https://python-pillow.org/) to perform image manipulations. The output format is a file with one Base64-encoded image per line.

We designed this application to operate in single-node mode or clustered mode, and added built-in logic for discovering cluster members when clustered. For this study, we used the application in single-node mode. We intended this preparation-stage application to be as computationally lightweight as possible. However, the pipeline is still compute-limited, even when using relatively slow storage.

This workload is particularly well suited to CPU comparisons with large thread/core-count disparity. Each thread operates independently, performs the same work, and is CPU-bound. Additional threads allow the server to complete more work per unit time, showing clear differentiation for higher core/thread count CPUs.

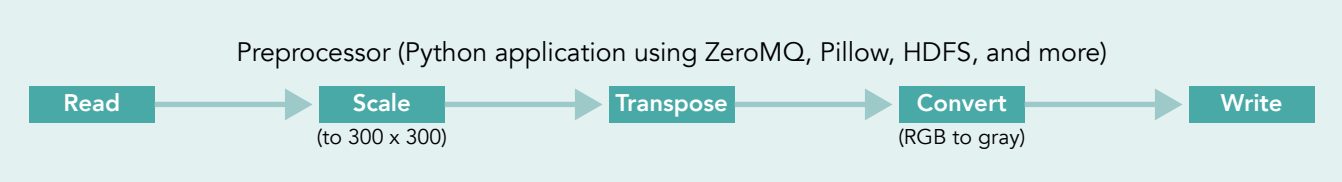

Figure 1: Pipeline data flow and operations. Source: Principled Technologies.

Figure 1 shows how the system reads images from storage, scales the images to a Machine-learning-friendly size of 300 x 300 pixels, transposes the images, and finally, converts the images to grayscale. The system then encodes the transformed image into JPG, and then converts the JPG-encoded to Base64 and writes it to the next line of the shard file. File systems for image-reading and shard-writing may be local storage or HDFS, though we used local storage for testing. In addition to the shard files, the workload also outputs frame count, byte count, frames per second, input and output bytes per second, and total runtime. In clustered mode, the workload also computes statistics across the cluster.

Application pseudocode

```
Let D = a directory containing JPG/PNG images
Let V = be the desired volume of data to process
Let N = number of nodes
Let M = number of processes per node

While not cluster achieved quorum: // (clustered mode only)
    Sleep // (clustered mode only)

V'=V/(M*N)

{launch M threads}:
Let selected = []
Let B_selected = 0 // number of bytes to write
For image in D:
    Selected.append(image)
    B_selected = B_selected + size(image)
    If B_selected > V'
            break
Let shard = open(output_file_name)
    For image in selected:
            Let image_resized = resize( image, [300,300] )
            Let image_transposed = transpose( image_resized )
            Let image_gray = rgb2gray( image_transposed )
            Let data = base64.encode( image_gray )
            shard.write(data+"\n")
    {compute thread statistics …}
{compute node statistics …}
Share statistics with cluster members…// (clustered mode only)
{compute cluster statistics …}// (clustered mode only)
```

Installing vSphere 7.0 Update 2 on a Dell EMC PowerEdge R7525

1. Download the Dell EMC Custom Image for ESXi 7.0 Update 2 from the following link: https://my.vmware.com/group/vmware/evalcenter?p=vsphere-eval-7#tab_download.
2. Open a new browser tab, and connect to the IP address of the Dell EMC PowerEdge R7525 server iDRAC.
3. Log in with the iDRAC credentials. We used `root/calvin`.
4. In the lower left of the screen, click Launch Virtual Console.
5. In the console menu bar, 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, and click Close.
8. On the console menu bar, click Boot, and select Virtual CD/DVD/ISO. To confirm, click Yes.
9. On the console menu bar, click Power. Select Power On System. To confirm, click Yes.
10. The system will boot to the mounted image and the Loading ESXi installer screen will appear. When prompted, press Enter to continue.
11. To Accept the EULA and Continue, press F11.
12. Select the storage device to target for installation. We selected the internal BOSS card with SATA M.2 SSDs. To continue, 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. To continue, press Enter.
16. To install, press F11.
17. Upon completion, reboot the server by pressing Enter.

Installing vCenter Server Appliance 7.0 Update 2

1. Download VMware vCenter 7.0 Update 2 from the VMware support portal: http://my.vmware.com.
2. Mount the image on your local system, and browse to the vcsa-ui-installer folder. Expand the folder for your OS, and launch the installer if it doesn't automatically begin.
3. When 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 R7525 servers with ESXi 7.0 Update 2. Provide 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 provide and confirm the root password for the VCSA. Click Next.
9. Set the size for the environment you're planning to deploy. We selected Medium. Click Next.
10. Select the datastore on which to install vCenter. 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. When Stage 1 has completed, click Close. To confirm, click Yes.
14. Open a browser window, and connect to https://vcenter.FQDN:5480/.
15. On the Getting Started - vCenter Server page, click Set up.
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 vSphere 7.0 Update 2

1. Open a browser, and enter the address of the vCenter server you deployed. For example: `https://[vcenter.FQDN]/ui`.
2. In the left panel, select the vCenter server, right-click, and select New Datacenter.
3. Provide a name for the new data center, and click OK.
4. Select the data center you just created, right-click, and select New Cluster.
5. Give a name to the cluster, and enable vSphere DRS. Click OK.
6. In the cluster configuration panel, under Add hosts, 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. Check the box beside Hostname/IP Address to select all hosts. Click Ok.
9. Click Next.
10. Click Finish.

Configuring the Dell EMC PowerStore 5000T

1. Connect a workstation or laptop to the service port on node A of the PowerStore enclosure.
2. In a web browser, go to `https://[Management IP of the PowerStore array]`.
3. Log onto PowerStore Manager, and begin the initial configuration process using the following default credentials:
   - Username: `admin`
   - Default Password: `Password123#`
4. Set the admin password, and click Update.
5. On the EULA, accept the agreement.
6. On Cluster Details, enter a cluster name, leave the Storage Configuration to the default selection, and click Next.
7. On the Networks page, select Configure Now.
8. Enter your desired Cluster IP configuration for the Management Network. Enter at least three additional IPs for the Management Network IPs, and click Continue.
9. For the Storage Network, select Configure later, and check the boxes to continue.
10. In Infrastructure Services, enter the DNS Server and the NTP Server, and click Next.
11. On Cluster Configuration, copy down the Cluster IP, and click Validate. Once complete, click Configure.
12. Once configuration is complete, click Next.

Configuring hosts on the Dell EMC PowerStore

1. From the PowerStore Management interface, select the Compute drop-down menu, and click Hosts & Host Groups.
2. Under Hosts & Host Groups, click Add Host.
3. For Host Details, provide a name for the host, select ESXi, and click Next.
4. For Host Protocol, keep the default selected (Fibre Channel), and click Next.
5. For Host Initiators, select the two identifiers associated with that host (WMM), and click Next.
6. For Summary, review the details, and click Add Host.
7. Complete steps 2 through 6 for the other hosts.

Creating a data volume on the Dell EMC PowerStore

1. From the PowerStore Management interface, select the Storage drop-down menu, and click Volumes.
2. Under Volumes, click Create.
3. For Properties, provide a name for the volume, set the size (2TB), and Volume Performance Policy. Click Next. This volume will be shared storage for Tanzu.
4. For Host Mappings, select all the hosts to which you would like to map the volume, and click Next.
5. Review the Summary, and click Next.

Creating a distributed vSwitch and port group

1. From vSphere client, click Home→Networking.
2. Select your Datacenter.
3. On the right panel, in the Actions drop-down menu, select Distributed vSwitch→New Distributed vSwitch.
4. Give your vSwitch a name, or accept the default. Click Next.
5. Select 7.0.2 - ESXi 7.0.2 and later as the version, and click Next.
6. Select the number of uplinks per ESXi host you'll give to the vSwitch. We selected 1. Click Next.
7. Click Finish.
8. Right-click the new DvSwitch, and select Add and Manage Hosts.
9. Leave Add hosts selected, and click Next.
10. To add new hosts, click the + sign.
11. To select all the hosts in your target cluster, check the box beside Host. Click OK. Click Next.
12. Select the NIC you want to use for this DvSwitch, and click Assign Uplink.
13. At the top of the panel, accept the defaults, but check the box for Apply this uplink assignment to the rest of the hosts. Click OK. Click Next.
14. Do not assign vmkernel adapters at this time. Click Next.
15. Do not migrate any VM networking at this time. Click Next.
16. Click Finish.
17. Right-click the DvSwitch, and select Distributed Port Group→New Distributed Port Group.
18. Name it `Workload Network`, and click Next.
19. Change the VLAN type to VLAN, and set the VLAN ID to VLAN 2. Click Next.
20. Click Finish.

Creating a DevOps user

1. From vCenter 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. Provide 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 DevOps user you just created, and add them to the administrators group. Click Save.

Creating the HAProxy content library

1. Download the vSphere-compatible HAproxy ovf (v0.1.8) from https://github.com/haproxytech/vmware-haproxy.
2. From vSphere client, in the left menu pane, 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, 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 drop-down menu, and select Import Item.
10. Change the selection to Local file, and click Upload files.
11. Browse to the location of the ovf file you downloaded in step 1, and click Open.
12. Click Import.

Creating the TKG content library

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

Creating the storage tag

1. From the vSphere client, select Menu→Storage.
2. From the left pane, select the shared storage you created on PowerStore 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. Use the Category drop-down menu to select Tanzu Storage, and click Create.
8. Check the box beside the newly created tag, and click Assign.

Creating the VM storage policy

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-clusters`, and click Next.
5. Check the box for Enable tag based placement rules, and click Next.
6. Use the Tag Category drop-down menu, and select the Tanzu Storage policy you created. Click Browse Tags.
7. Click the Tanzu checkbox, and click OK.
8. Click Next.
9. Review the compatible storage, making sure your storage target is marked as compatible, and click Next.
10. Click Finish.

Deploying 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 in the panel below. Select New VM from This Template…
4. Provide a simple name—we used `HAproxy`—and select the Datacenter 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 details, and click Next.

7. Check the box to 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 Frontend network, and click Next.
11. Customize the template. We used the following:
    - Appliance Configuration
        - For the root password, we used `Password1!`
        - Check the box for Permit Root Login
        - Leave the TLS CA blank
    - Network Configuration
        - We left the default haproxy.local
        - For local DNS server, we used `10.41.0.10`
        - For management IP, we used `10.218.201.200/16`
        - For management gateway, we used `10.218.0.1`
            - Note: The description asks for the workload network gateway address. You should enter the management gateway address instead.
        - For Workload IP, we used `192.168.1.2/24`
        - For Workload gateway, we used `192.168.1.1`
    - Load Balancing
        - For load balancer IP ranges, we used `192.168.1.240/29`
        - Accept the default management port
        - For HAProxy User ID, we used `admin`
        - For the HAProxy password, we used `Password1!`
12. Click Next.
13. Review the summary, and click Finish. The deployment will take a few minutes to completely deploy and configure.
14. Power on the HAProxy VM.

Configuring Workload Management

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 you wish to use for the control plane nodes. We chose tkg-clusters. Click Next.
7. Configure the Load Balancer section with the following:
    a. Name: `haproxy`
    b. Type: `HA proxy`
    c. Data plane API Addresses: `10.218.201.200:5556`
    d. User name: `admin`
    e. Password: `Password1!`
    f. IP Address Ranges for Virtual Servers: `192.168.1.240-192.168.1.247`
    g. Server Certificate Authority: [copy and pasted from the instructions below]
8. Open an SSH session to the HAProxy management address, and connect using `root` and `Password1!`.
9. Type `cat /etc/haproxy/ca.crt`.
10. Copy the entire output (including the first and last lines) and paste the contents into the Server Certificate Authority box in configuration setting g from step 7.

11. Close the SSH session.
12. Click Next.
13. Configure Workload Management with the following:
    a. Network: `VM Network`
    b. Starting IP Address: `10.218.201.201`
    c. Subnet Mask: `255.255.0.0`
    d. Gateway: `10.218.0.1`
    e. DNS Server: `10.41.0.10`
    f. NTP Server: `10.40.0.1`
14. Click Next.
15. Configure Workload Network with the following:
    a. Leave the default for Services addresses.
    b. Configure the DNS server with appropriate settings for the environment.
    c. Under Workload Network, click Add.
    d. Accept default for network-1.
    e. Port Group: `Workload Network`
    f. Gateway: `192.168.1.1`
    g. Subnet: `255.255.255.0`
    h. IP Address Ranges: `192.168.1.65-192.168.1.126`
16. Click Save.
17. For TKG Configuration, use the following:
    a. Beside Add Content Library, click Add.
    b. Select the TKG-cl library, and click OK.
18. Click Next.
19. Click Finish. The workload management cluster will deploy and configure. You may see apparent errors during configuration, but these will resolve upon successful completion.

Configuring Kubernetes namespace for service deployment

1. In Workload Management, click Namespaces.
2. Click Create Namespace.
3. Select the target cluster, and provide a name. We used `tanzu-ns`. Click Create.
4. Click the Permissions tab, and click Add.
5. Choose vSphere.local for the identity source. Search for the DevOps user you created, select the "can edit" role, and 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 deploying containers.

Installing and configuring Ubuntu VM for Tanzu Kubernetes Grid CLI

1. Log into vCenter, and from the Menu drop-down 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. From the Guest OS Family drop-down menu, select Linux.
11. From the Guest OS version drop-down menu, select Ubuntu Linux (64 bit), and click Next.
12. Assign the VM 2 vCPUs, 8 GB of memory, and a 40GB thick-provisioned VMDK.
13. From the New CD/DVD Drive dropdown, select Datastore ISO File, and select the Ubuntu ISO you uploaded to the datastore previously. Ensure Connect At Power On is checked, and click Next.
14. Click Finish.
15. Power on the VM, and click Launch Remote Console.
16. Click Install Ubuntu.

17. Click Continue.
18. Select Minimal installation, and click Continue.
19. Click Install now.
20. Click Continue.
21. Click Continue.
22. Enter your desired full name, computer name, username, and password, and click Continue.
23. Click Restart Now.
24. Press Enter.
25. Enter your password, and click Sign In.
26. When prompted by the Software Updater, install OS and software updates.
27. Click Restart Later, and power off the VM.
28. Right-click the VM in vCenter, and click Edit Settings.
29. Click Add New Device, and select VMXNET3.
30. From the New Network drop-down menu, select Browse.
31. Click Workload Network, and click OK.
32. Click OK.
33. Power on the VM, and click Launch Remote Console.
34. Enter your password, and click Sign In.
35. Click the network icon in the top right, and click Ethernet (ens192).
36. Click Wired Settings.
37. Next to Ethernet (ens192) click the gear icon.
38. Click IPv4.
39. Change IPv4 Method to Manual, and enter the following settings:
    - Address: `192.168.1.9`
    - Netmask: `255.255.255.0`
40. Click Apply, and close the settings window.
41. Open a browser on the VM, and navigate to http://www.vmware.com/go/get-tkg.
42. Click Go to downloads.
43. Next to VMware Tanzu Kubernetes Grid CLI for Linux, click Download Now.
44. Log in with your VMware credentials.
45. Scroll to the bottom of the EULA window, and check the box to agree to the EULA.
46. Click Accept.
47. Open a terminal, and navigate to the Downloads folder.
48. Unzip the archive:

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

49. 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
```

50. Initialize TKG CLI for the first time:

```
tkg get management-cluster
```

Installing Kubectl Plugin for vSphere

1. Open a browser on the VM, and navigate to the IP of one of the three Supervisor VMs. In our environment, the first control plane VM IP was 10.218.201.201.
2. Click Advanced, and click Accept the Risk and Continue to bypass the certificate warning.
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/
sudo mv kubectl /usr/local/bin/
```

Creating the image processing workload

1. From the Ubuntu VM, log into the Supervisor Cluster:

```
kubectl vsphere login --insecure-skip-tls-verify --server=https://10.218.201.201--vsphere-username
administrator@vsphere.local
```

2. Create a yaml file for a cluster of nine worker nodes with CPU and Memory fully reserved:

```
    cat cluster.yaml
  apiVersion: run.tanzu.vmware.com/v1alpha1
  kind: TanzuKubernetesCluster
      metadata:
name: ai-cluster
  namespace: tanzu-ns
      spec:
 topology:
   controlPlane:
     count: 1
     class: guaranteed-large
     storageClass: tkg-clusters
     workers:
       count: 44
       class: guaranteed-large
       storageClass: tkg-clusters
  distribution:
    version: v1.18
    settings:
      network:
        cni:
          name: antrea
        services:
          cidrBlocks: ["10.96.1.0/24"]
        pods:
          cidrBlocks: ["172.16.0.0/16"]
```

3. Create the cluster:

```
    Kubectl apply -f cluster.yaml
```

4. Log into the new cluster:

```
kubectl vsphere login --insecure-skip-tls-verify --vsphere-username administrator@vsphere.local
--server=https://10.218.201.201 --tanzu-kubernetes-cluster-name ai-cluster --tanzu-kubernetes-
cluster-namespace tanzu-ns
```

5. Create a yaml file for three PersistentVolumeClaims:

```
    cat pvc.yaml
      apiVersion: v1
      kind: PersistentVolumeClaim
      metadata:
name: ai-pvc-data
      spec:
accessModes:
  - ReadWriteOnce
storageClassName: tkg-clusters
resources:
  requests:
    storage: 2Gi
     ---
     apiVersion: v1
     kind: PersistentVolumeClaim
     metadata:
       name: ai-pvc-app
     spec:
       accessModes:
  - ReadWriteOnce
```

```
      storageClassName: tkg-clusters
      resources:
  requests:
storage: 1Gi
     ---
     apiVersion: v1
     kind: PersistentVolumeClaim
     metadata:
       name: ai-pvc-out
     spec:
       accessModes:
         - ReadWriteOnce
       storageClassName: tkg-clusters
       resources:
         requests:
storage: 2Gi
```

6. Create 44 sets of PVCs (one set for each pod) in the cluster:

```
Kubectl apply -f pvc.yaml
```

7. Create a yaml file for the image processing pod with the following content:

```
    cat pod.yaml
  apiVersion: v1
  kind: Pod
  metadata:
    name: ai-pod
  spec:
    containers:
    - name: ai
  image: ptuser/preparation:latest
      ports:
        - containerPort: 80
      resources:
        requests:
          memory: "10Gi"
          cpu: "3"
        limits:
          memory: "10Gi"
          cpu: "3"
      volumeMounts:
        - mountPath: "/app"
          name: ai-app
        - mountPath: "/data"
          name: ai-data
        - mountPath: "/out"
          name: ai-out
    volumes:
      - name: ai-app
        persistentVolumeClaim:
          claimName: ai-pvc-app
      - name: ai-data
        persistentVolumeClaim:
          claimName: ai-pvc-data
      - name: ai-out
        persistentVolumeClaim:
          claimName: ai-pvc-out
```

8. Create 44 pods, and mount the PVCs in the cluster:

```
Kubectl apply -f pod.yaml
```

9. To run the benchmark, navigate to the test code GIT repository and follow the instructions in the testing/README.md file. For this study, we used the application in single-node mode. See the Workload description section for more information.

We ran the image processing workload three times on each server. For each configuration of the cluster, we report the total FPS rate of all pods from the median run. Additionally, for each configuration, we report the time of the pod that took the most time to complete the workload.

Read the report at http://facts.pt/ObCtwcb ▶

This project was commissioned by Dell Technologies.