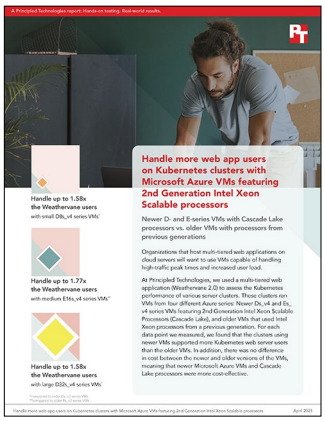

The science behind the report:

Handle more web app users on Kubernetes clusters with Microsoft Azure VMs featuring 2nd Generation Intel Xeon Scalable 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 Handle more web app users on Kubernetes clusters with Microsoft Azure VMs featuring 2nd Generation Intel Xeon Scalable processors.

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

Our results

To learn more about how we have calculated the wins in this report, go to http://facts.pt/calculating-and-highlighting-wins. Unless we state otherwise, we have followed the rules and principles we outline in that document.

Table 1: Weathervane 2.0 results for Microsoft Azure D-series VMs

VM name	Weathervane users
8vCPU VMs	
D8s_v4	24,700
D8s_v3	15,550
32vCPU results	
D32s_v4	75,800
D32s_v3	47,775

Table 2: Weathervane 2.0 results for Microsoft Azure E-series VMs

VM name	Weathervane users
8vCPU VMs	
E8s_v4	30,200
E8s_v3	20,300
16vCPU VMs	
E16s_v4	50,950
E16s_v3	28,725

System configuration information

Note: due to the fact that Azure Dsv3 series VMs run on a variety of processor gens (Haswell, Broadwell, Skylake, and Cascade Lake), in combination with the fact that a user deploying an AKS cluster is unable to specify which generation of processors the underlying nodes will use, we ended up with a variety of processors in each AKS cluster. Each Dsv3 series cluster had one Haswell node, one Broadwell node, and one Skylake node. The three different processor configurations are reflected in the Processor section of the table below.

Table 3: Detailed information on the D-series VMs we tested

Cluster configuration information	3x 8vCPU Older Gen VM	3x 8vCPU Current Gen VM	3x 32vCPU Older Gen VM	32x 32vCPU Current Gen VM
Tested by	Principled Technologies	Principled Technologies	Principled Technologies	Principled Technologies
Test date	03/22/2021	03/26/2021	03/24/2021	03/25/2021
CSP / Region	Microsoft Azure East US (Zone 1)	Microsoft Azure East US (Zone 1)	Microsoft Azure East US (Zone 1)	Microsoft Azure East US (Zone 1)
Workload & version	Weathervane 2.0.9	Weathervane 2.0.9	Weathervane 2.0.9	Weathervane 2.0.9
WL specific parameters	2x small2 app instances, useAppServerLimits : false	2x small2 app instances, useAppServerLimits : false	8x small2 app instances, useAppServerLimits : false	8x small2 app instances, useAppServerLimits : false
Iterations and result choice	3 runs, median	3 runs, median	3 runs, median	3 runs, median
Server platform	3x D8s_v3	3x D8s_v4	3x D32s_v3	3x D32s_v4
BIOS name and version	American Megatrends Inc. Version 090008, 12/07/2018	American Megatrends Inc. Version 090008, 12/07/2018	American Megatrends Inc. Version 090008, 12/07/2018	American Megatrends Inc. Version 090008, 12/07/2018
Operating system name and version/build number	Ubuntu 18.0.4 LTS	Ubuntu 18.0.4 LTS	Ubuntu 18.0.4 LTS	Ubuntu 18.0.4 LTS
Date of last OS updates/ patches applied	03/22/2021	03/26/2021	03/24/2021	03/25/2021
Processor				
Number of processors	1	1	1	1
Vendor and model	Intel® Xeon® E5-2673 v3 Intel Xeon E5-2673 v4 Intel Xeon 8171M	Intel Xeon® Platinum 8272CL	Intel Xeon E5-2673 v3 Intel Xeon E5-2673 v4 Intel Xeon 8171M	Intel Xeon Platinum 8272CL
Core count (per processor)	12 20 26	26	12 20 26	26
Core frequency (GHz)	2.40 2.30 2.60	2.60	2.40 2.30 2.60	2.60
Stepping	1 1 1	7	1 1 1	7
Hyper-Threading	Yes	Yes	Yes	Yes
Turbo	Yes	Yes	Yes	Yes
Number of vCPU per VM	8	8	32	32
Memory module(s)				
Total memory in system (GB)	32	32	128	128

Cluster configuration information	3x 8vCPU Older Gen VM	3x 8vCPU Current Gen VM	3x 32vCPU Older Gen VM	32x 32vCPU Current Gen VM
NVMe memory present?	No	No	No	No
Total memory (DDR+NVMe RAM)	32	32	128	128
General hardware				
Storage: NW or Direct Att / Instance	NW Att	NW Att	NW Att	NW Att
Network BW / Instance	N/A	N/A	N/A	N/A
Storage BW / Instance	N/A	N/A	N/A	N/A
Local storage				
OS				
Number of drives	1	1	1	1
Drive size (GB)	127	127	127	127
Drive information (speed, interface, type)	Standard HDD	Standard HDD	Standard HDD	Standard HDD
Data drive				
Number of drives	8	8	32	32
Drive size (GB)	2x 60GB, 2x 10GB, 4x 5GB	2x 60GB, 2x 10GB, 4x 5GB	8x 60GB, 8x 10GB, 16x 5GB	8x 60GB, 8x 10GB, 16x 5GB
Drive information (speed, interface, type)	Premium SSD	Premium SSD	Premium SSD	Premium SSD
Temporary drive				
Number of drives	1	0	1	0
Drive size (GB)	200	N/A	800	N/A
Network adapter				

Cluster configuration information	3x 8vCPU Older Gen VM	3x 8vCPU Current Gen VM	3x 32vCPU Older Gen VM	32x 32vCPU Current Gen VM
Vendor and model	Mellanox ConnectX-3	Mellanox ConnectX-4	Mellanox ConnectX-3	Mellanox ConnectX-4
Number and type of ports	1x 40Gb	1x 50Gb	1x 40Gb	1x 50Gb

Note: Due to the fact that Azure Es_v3 series VMs run on a variety of processor gens (Haswell, Broadwell, Skylake, and Cascade Lake), in combination with the fact that a user deploying an AKS cluster is unable to specify which generation of processors the underlying nodes will use, we ended up with a variety of processors in each AKS cluster. The three different processor configurations are reflected in the Processor section of the table below.

Table 4: Detailed information on the E-series VMs we tested

Cluster configuration information	3x 8vCPU Older Gen VM	3x 8vCPU Current Gen VM	3x 16vCPU Older Gen VM	3x 16vCPU Current Gen VM
Tested by	Principled Technologies	Principled Technologies	Principled Technologies	Principled Technologies
Test date	03/11/2021	03/10/2021	03/12/2021	03/16/2021
CSP / Region	Microsoft Azure East US (Zone 1)	Microsoft Azure East US (Zone 1)	Microsoft Azure East US (Zone 1)	Microsoft Azure East US (Zone 1)
Workload & version	Weathervane 2.0.9	Weathervane 2.0.9	Weathervane 2.0.9	Weathervane 2.0.9
WL specific parameters	3x small2 app instances, useAppServerLimits : false	3x small2 app instances, useAppServerLimits : false	7x small2 app instances, useAppServerLimits : false	7x small2 app instances, useAppServerLimits : false
Iterations and result choice	3 runs, median	3 runs, median	3 runs, median	3 runs, median
Server platform	3x E8s_v3	3x E8s_v4	3x E16s_v3	3x E16s_v4
BIOS name and version	American Megatrends Inc. Version 090008, 12/07/2018	American Megatrends Inc. Version 090008, 12/07/2018	American Megatrends Inc. Version 090008, 12/07/2018	American Megatrends Inc. Version 090008, 12/07/2018
Operating system name and version/build number	Ubuntu 18.0.4 LTS	Ubuntu 18.0.4 LTS	Ubuntu 18.0.4 LTS	Ubuntu 18.0.4 LTS
Date of last OS updates/ patches applied	03/11/2021	03/10/2021	03/12/2021	03/16/2021
Processor				
Number of processors	1	1	1	1
Vendor and model	Intel Xeon E5-2673 v3 Intel Xeon E5-2673 v4 Intel Xeon 8171M	Intel® Xeon® Platinum 8272CL	Intel Xeon E5-2673 v3 Intel Xeon E5-2673 v4 Intel Xeon 8171M	Intel Xeon Platinum 8272CL
Core count (per processor)	12 20 26	26	12 20 26	26
Core frequency (GHz)	2.40 2.30 2.60	2.60	2.40 2.30 2.60	2.60
Stepping	1 1 1	7	1 1 1	7

Cluster configuration information	3x 8vCPU Older Gen VM	3x 8vCPU Current Gen VM	3x 16vCPU Older Gen VM	3x 16vCPU Current Gen VM
Hyper-Threading	Yes	Yes	Yes	Yes
Turbo	Yes	Yes	Yes	Yes
Number of vCPU per VM	8	8	16	16
Memory module(s)				
Total memory in system (GB)	64	64	128	128
NVMe memory present?	No	No	No	No
Total memory (DDR+NVMe RAM)	64	64	128	128
General hardware				
Storage: NW or Direct Att / Instance	NW Att	NW Att	NW Att	NW Att
Network BW / Instance	N/A	N/A	N/A	N/A
Storage BW / Instance	N/A	N/A	N/A	N/A
Local storage				
OS				
Number of drives	1	1	1	1
Drive size (GB)	127	127	127	127
Drive information (speed, interface, type)	Standard HDD	Standard HDD	Standard HDD	Standard HDD
Data drive				
Number of drives	12	12	28	28
Drive size (GB)	3x 60GB, 3x 10GB, 6x 5GB	3x 60GB, 3x 10GB, 6x 5GB	7x 60GB, 7x 10GB, 14x 5GB	7x 60GB, 7x 10GB, 14x 5GB
Drive information (speed, interface, type)	Premium SSD	Premium SSD	Premium SSD	Premium SSD
Temporary drive				
Number of drives	0	0	0	0
Drive size (GB)	N/A	N/A	N/A	N/A
Network adapter				
Vendor and model	Mellanox ConnectX-4	Mellanox ConnectX-4	Mellanox ConnectX-4	Mellanox ConnectX-4
Number and type of ports	1x 50Gb	1x 50Gb	1x 50Gb	1x 50Gb

How we tested

Testing overview

For this project, we will test Azure Kubernetes Service (AKS) cluster featuring older Intel Xeon processors vs. clusters using Cascade Lake processors. We will run the VMware Weathervane 2.0 benchmark on the Azure instances to show the increased cluster capabilities that customers can expect to see using the newer instance series compared to the previous generation.

Creating the run-harness VM

This section contains the steps we took to create our run-harness VM.

Creating the run-harness VM

1. Log into the Azure Portal, and click Virtual machines.
2. Click Add Virtual machine.
3. On the Basics tab, set the following:
    a. Choose your Subscription.
    b. Choose your Resource group.
    c. Name the VM.
    d. Select your desired Region.
    e. Leave Availability options default.
    f. Choose your base image. We used Ubuntu 20.04 LTS.
    g. Leave Azure Spot instance unchecked.
    h. Choose your instance size. We used D2s v3.
    i. Leave the rest of the options default.
    j. Click Next: Disks.
4. On the Disks tab, set the following:
    a. Under OS disk type, choose Standard SSD.
    b. Leave the rest of the default options, and click Next: Networking.
5. On the Networking tabs, leave all of the default options, and click Next: Management.
6. On the Management tab, leave all of the default options, and click Next: Advanced.
7. On the Advanced tab, leave all of the default options, and click Next: Tags.
8. On Tags tab, add any tags you wish to use for the VM. Click Next: Review + create.
9. On the Review + create tab, ensure your options are correct, and click Create.

Configuring the run-harness VM

1. Using the ssh command in the Connect tab for the VM, log into the run-harness VM.
2. Install latest updates:

```
sudo apt update
sudo apt-get upgrade -y
sudo reboot
```

3. Install unzip:

```
sudo apt-get unzip
```

4. Set up the prerequisites for Docker repository access:

```
sudo apt-get install \
  apt-transport-https \
  ca-certificates \
  curl \
  gnupg
```

5. Add the official Docker PGP key:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/
    docker-archive-keyring.gpg
```

6. Set up the stable Docker repository:

```
echo \
"deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.
    com/linux/ubuntu \
$(lsb_release -cd) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

7. Install Docker Engine:

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

8. Install git, if not already installed:

```
sudo apt-get install git
```

9. Clone the Weathervane repository to the run-harness VM:

```
git clone https://github.com/vmware/weathervane
```

10. Create the Weathervane Docker images (If you do not have a Docker Hub account, you will need to create one to host the Docker images for the benchmark):

```
cd weathervane
./buildDockerImages.pl –username [Docker Hub username]
```

11. When prompted, input your Docker Hub password.
12. Once the build has finished, create a copy of the Weathervane configuration file:

```
cp weathervane.config.k8s.micro weathervane.config.k8s.quickstart
```

13. The following section shows how to edit the configuration files.

Creating and accessing the Azure Kubernetes Service (AKS) cluster

This section contains the steps we took to create our Kubernetes cluster.

Creating the Kubernetes cluster

1. Log into the Azure Portal, and click Create a resource.
2. Select Containers, and click Kubernetes Service.
3. On the Basics tab, set the following:
    a. Choose your Subscription.
    b. Choose your Resource group.
    c. Name the Kubernetes cluster.
    d. Choose your Region.
    e. Leave the Availability options set to default.
    f. For Kubernetes version, choose 1.20.2.
    g. Select the node size and count you wish to use for you systems under test (SUT). We used 3.
    h. Click Next: Node pools.
4. On the Node pools tab, to create an additional node pool for a driver node, do the following:
    a. Click Add node pool.
    b. Name the node pool.
    c. Leave Mode and OS type as the defaults.
    d. Choose matching Availability zones to the agent node pool created in the previous step.
    e. Select a node size.
    f. Set node count to 1.
    g. Click Add.
    h. Review the details of the new node pool, and click Next: Authentication.
5. On the Authentication tab, set the following:
    a. Under Authentication method, choose Service principal.
    b. Leave the rest of the settings at defaults.
    c. Click Next: Networking.

6. On the Networking tab, leave all defaults.
7. On the Integrations tab, leave all defaults.
8. On the Tags tab, add any tags you wish to use.
9. On the Review + create tab, review your settings, and click Create

Setting up the Kubernetes cluster.

1. In the Azure Portal, open an Azure CLI shell.
2. Create a config file for your cluster:

```
az aks get-credentials --resource-group [resource group] --name [cluster name]
```

3. Get names and internal IPs of the cluster nodes:

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

4. Using the names of the nodes retrieved above, label the agentpool nodes as SUTs and the driverpool node as a driver:

```
kubectl label node [driver node name] wvrole=driver
kubectl label node [SUT node name] wvrole=sut
```

5. We used a custom StorageClass (included below) so that we could use Premium SSDs with a volumeBindingMode of Immediate:

```
kubectl apply -f gurira-prem.yaml
```

6. View the contents of your kubeconfig file:

```
cat ~/.kube/config
```

7. Copy the output to a text file on your local computer, and copy the file to the run-harness VM.
8. Create a .kube folder on the run harness, and copy the config file to it while removing any file extension:

```
mv config.txt .kube/config
```

9. Test that you can successfully access the Kubernetes cluster from the run-harness:

```
kubectl get nodes
```

SSH-ing into the AKS cluster

1. In the Azure Portal, open an Azure CLI shell.
2. Create a variable for your newly created AKS resource group:

```
CLUSTER_RESOURCE_GROUP=$(az aks show --resource-group [resource group] --name [cluster name] --query
    nodeResourceGroup -o tsv)
```

3. If there is not currently an SSH key associated with your Azure CLI instance, create one:

```
ssh-keygen
```

4. In the Azure Portal, navigate to Resource Groups.
5. Select the newly created Resource Group for your cluster (it should begin with MC, and have the name of the cluster within it).
6. Note the full name of the agentpool VMSS (virtual machine scale set), and copy it for the next steps.
7. Create an extension set pointing to the public SSH key for the VMSS:

```
az vmss extension set  \
--resource-group $CLUSTER_RESOURCE_GROUP \
--vmss-name [agentpool vmss name] \
--name VMAccessForLinux \
--publisher Microsoft.OSTCExtensions \
--version 1.4 \
--protected-settings "{\"username\":\"azureuser\", \"ssh_key\":\"$(cat ~/.ssh/id_rsa.pub)\"}"
```

8. Update the instances in the agentpool VMSS:

```
az vmss update-instances --instance-ids '*' \
--resource-group $CLUSTER_RESOURCE_GROUP \
--name [agentpool vmss name]
```

9. Create a jumpbox between the CLI instance and the AKS nodes:

```
kubectl run -it --rm aks-ssh --image=debian
```

10. Update the jumpbox, and install OpenSSH:

```
apt-get update && apt-get install openssh-client -y
```

11. Open a new window on the Azure Portal, and start a new CLI instance.
12. From the new CLI instance, copy over the public SSH key:

```
kubectl cp ~/.ssh/id_rsa $(kubectl get pod -l run=aks-ssh -o jsonpath='{.items[0].metadata.
    name}'):/id_rsa
```

13. Back in the jumpbox terminal, set permissions for the key:

```
chmod 0600 id_rsa
```

14. You can now SSH to your AKS nodes from the jumpbox:

```
ssh -i id_rsa azureuser@[node internal ip]
```

15. Outside of the jumpbox, use the following command to find AKS node internal IPs:

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

16. Once inside the nodes, you can check the processor type, and run performance monitoring applications such as iostat or nmon.

Running the tests

In this section, we list the steps to run the VMware Weathervane benchmark on the clusters under test. As each cluster had different hardware specs, please refer to Table 3 and Table 4 to see the number of applications to run on each cluster. We used the findMaximumSingleRun test type in order to determine the largest number of wvusers that each cluster could sustain while still maintaining the QoS standards required to pass the benchmark.

Running the tests

1. Log into the run-harness instance.
2. Navigate to the weathervane folder.
3. Open the run configuration file:

```
sudo vim weathervane.config.k8s.quickstart
```

4. Make sure the following information is correct:
    a. description: enter a description of the run
    b. configurationSize: small2
    c. runStrategy: findMaxSingleRun
    d. kubeconfigFile: [kube config file location]
    e. kubeconfigContext: [k8s cluster name]
    f. appIngressMethod: clusterip
    g. StorageClasses: gurira-prem
    h. numAppInstance: [# of app instances]
    i. useAppServerLimits: false
5. Exit the text editor.
6. Run the test:

```
sudo ./runWeathervane.pl –configFile=weathervane.config.k8s.quickstart
```

7. Once the test completes, reboot both the agentpool and driverpool vmss.
8. Repeat the test two more times for a total of three runs, and report the median run.

gurira-prem.yaml

```
allowVolumeExpansion: true
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
```

```
  creationTimestamp: "2021-02-25T13:13:34Z"
  labels:
    addonmanager.kubernetes.io/mode: EnsureExists
    kubernetes.io/cluster-service: "true"
  managedFields:
  - apiVersion: storage.k8s.io/v1beta1
    fieldsType: FieldsV1
    fieldsV1:
      f:allowVolumeExpansion: {}
      f:metadata:
        f:labels:
          .: {}
          f:addonmanager.kubernetes.io/mode: {}
          f:kubernetes.io/cluster-service: {}
      f:parameters:
        .: {}
        f:cachingmode: {}
        f:kind: {}
        f:storageaccounttype: {}
      f:provisioner: {}
      f:reclaimPolicy: {}
      f:volumeBindingMode: {}
    manager: kubectl-create
    operation: Update
    time: "2021-02-25T13:13:34Z"
  name: gurira-prem
  resourceVersion: "342"
  #uid: d42cd257-6985-47bf-aefa-53922ecfea49
parameters:
  cachingmode: ReadOnly
  kind: Managed
  storageaccounttype: Premium_LRS
provisioner: kubernetes.io/azure-disk
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

Read the report at http://facts.pt/tjJuw4u ▶

This project was commissioned by Intel.