Implementing WEKA with Azure AKS

This document will describe how to:

1. Deploy an AKS cluster in Azure
2. Deploy a WEKA cluster into the existing network created by AKS (via Terraform) - Using UDP Mode
3. Install the CSI plugin
4. Configure a daemonset to automatically install the WEKA client on new worker nodes (AKS autoscaling)
5. Deploy an example fio statefulset application

Note that each of these steps can be done individually if previous state has already been achieved (e.g. if an AKS and Weka cluster already exist you can install the CSI plugin and configure the daemon sets).

Additionally most of these steps will be performed only once (e.g. deploy the AKS cluster, Deploy the Weka cluster, Install CSI Plugin, Configure the Daemonset) following that every pod that will run on every AKS Worker node (autoscale or static) will be able to provision a PVC on the weka cluster and work with it.

This document will describe how to:

1. Deploy an AKS cluster in Azure
2. Deploy a WEKA cluster into the existing network created by AKS (via Terraform)
3. Install the CSI plugin
4. Configure a daemonset to automatically install the WEKA client on new worker nodes (AKS autoscaling)
5. Deploy an example fio statefulset application

Deploy an AKS cluster:

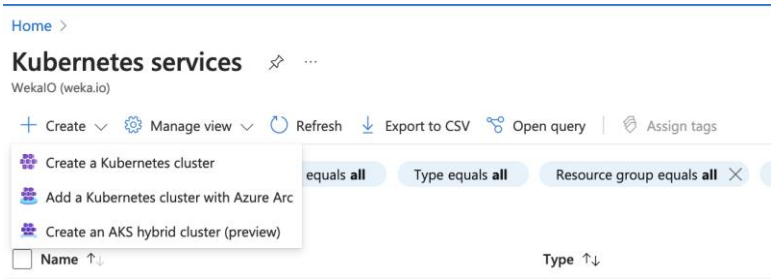
Resource group:

If you don't have one already, create a resource group. You can do this by browsing to portal.azure.com and using the cloudshell. For example:

```
az group create --name weka-alexh-rg --location eastus
```

AKS Deployment:

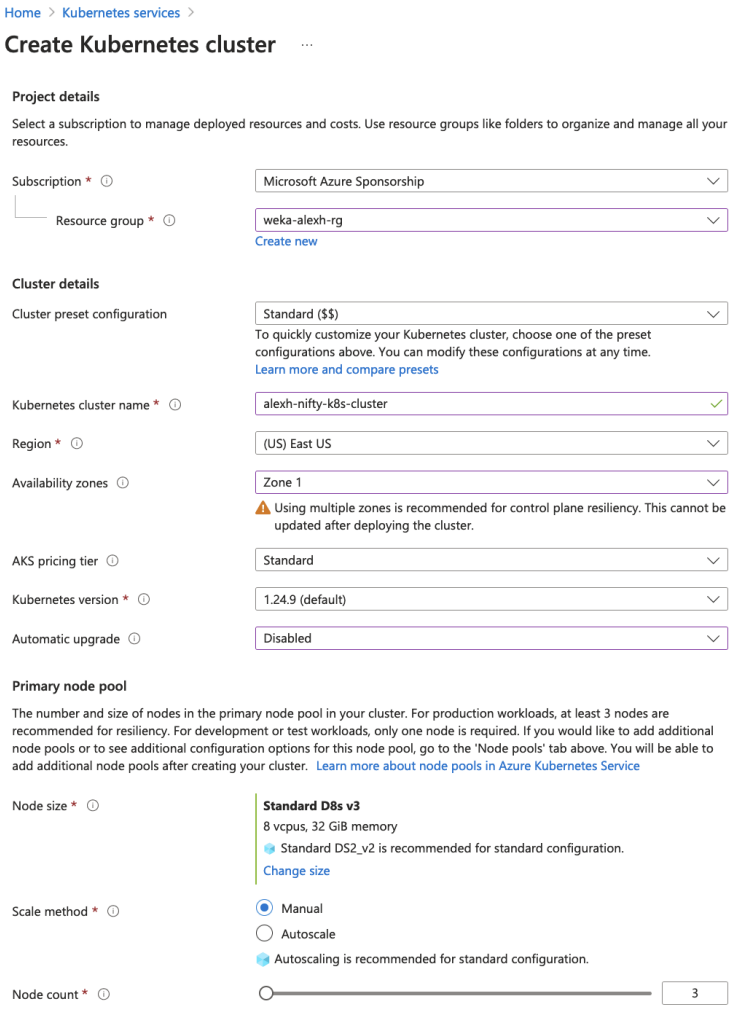
Go to portal.azure.com and search for 'Kubernetes Services'. Under the Create tab, select 'Create a Kubernetes cluster':



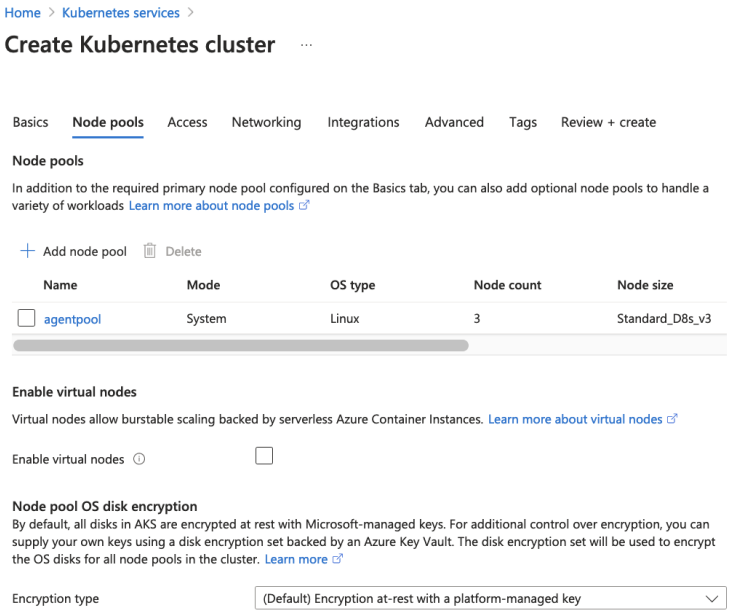
Under the basics tab, ensure the following:

1. Resource group - This should be an existing resource group or the one you created earlier. We will be deploying WEKA into this same resource group later on.
2. Kubernetes cluster name - pick something easy to remember, you'll need it later on
3. Availability zones - For the time being, pick a single zone (Zone 1)
4. Automatic upgrade - Disabled, for now
5. Node size - D8sv3 is alright for demos
6. Scale method - Set to Manual with node count = 3

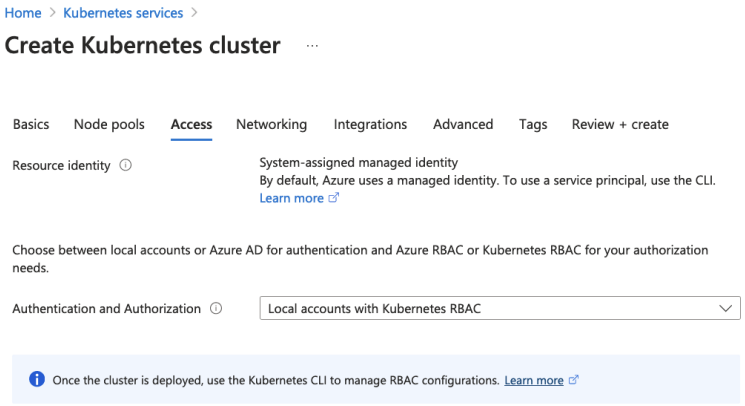
Example Basics tab below:



Under the node pools tab, just leave the defaults:



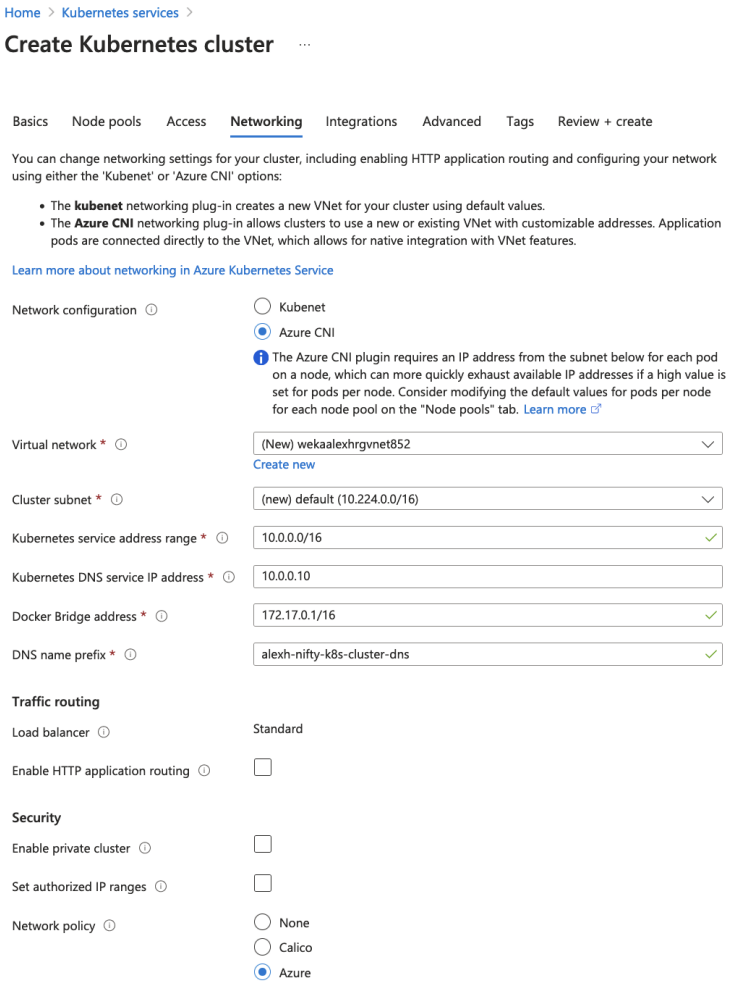
For the Access tab, just leave the defaults:



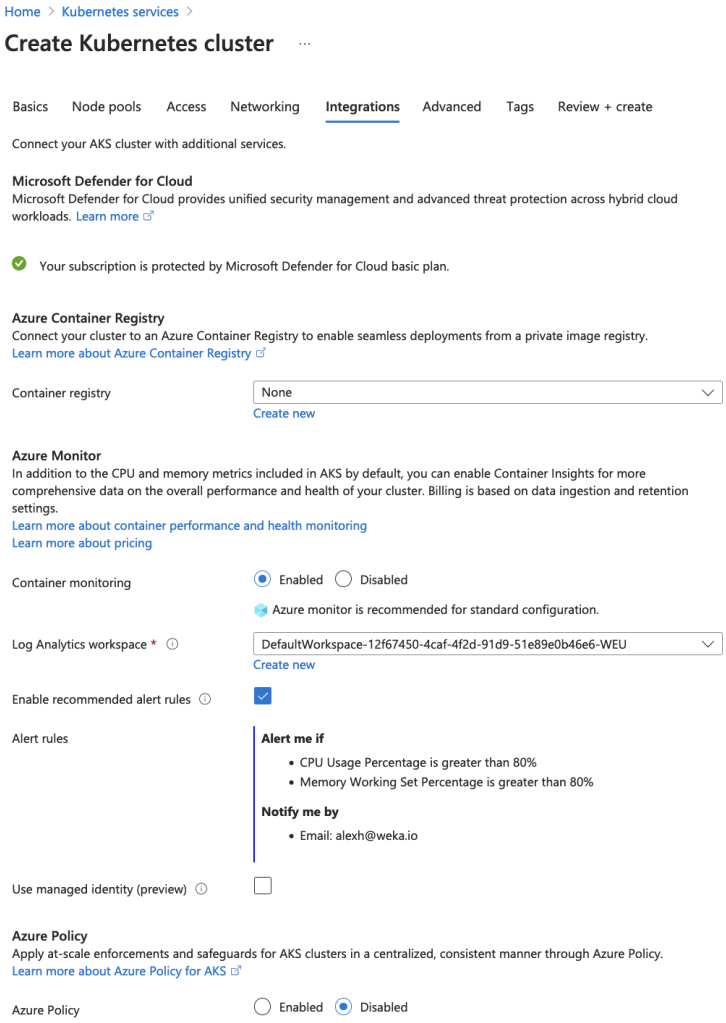
On the Networking tab, select the following options:

1. Network configuration - Azure CNI
2. Virtual network - Allow it to create a new one
3. Network policy - Azure
4. Rest of the settings, leave at defaults

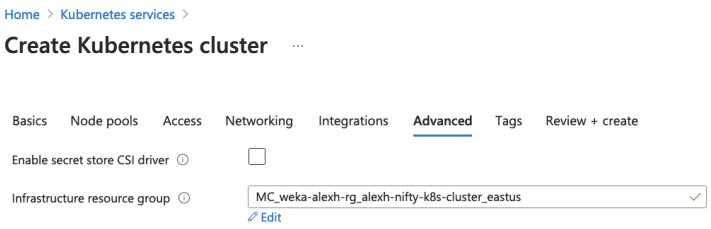
Example Networking tab:



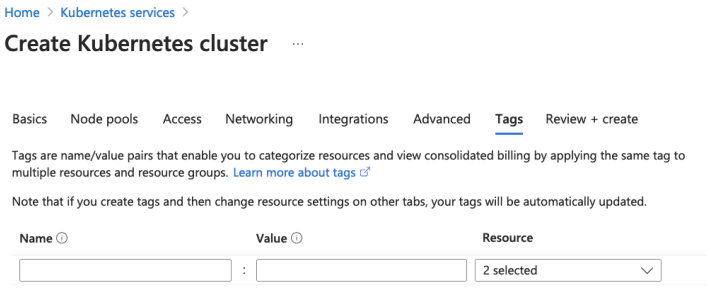
Integrations tab, just leave at the defaults:



Advanced tab, leave at the defaults:

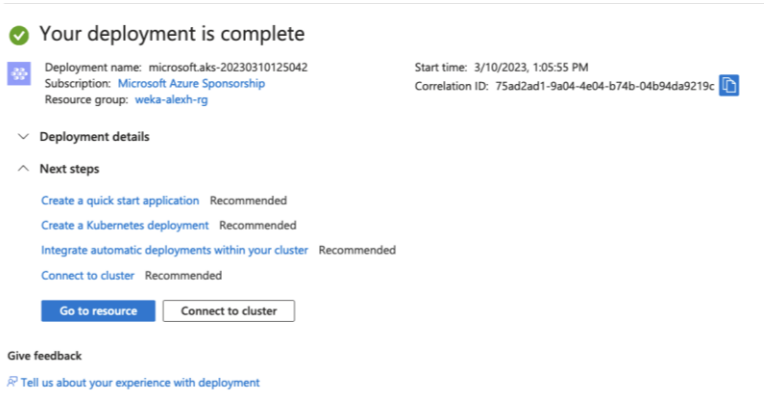


Tags tab, leave at the defaults:



Finally, review and create the cluster:

Once the AKS cluster is deployed, click "Connect to cluster":



There are several options listed. It is easiest to open up a cloud shell and run the following:

```
alex [ ~ ]$ az aks get-credentials --resource-group weka-alexh-rg --name alexh-nifty-k8s-cluster
The behavior of this command has been altered by the following extension: aks-preview
Merged "alexh-nifty-k8s-cluster" as current context in /home/alex/.kube/config
alex [ ~ ]$ kubectl get nodes
NAME                                STATUS   ROLES   AGE     VERSION
aks-agentpool-23409389-vmss000000   Ready    agent   9m35s   v1.24.9
aks-agentpool-23409389-vmss000001   Ready    agent   9m35s   v1.24.9
aks-agentpool-23409389-vmss000002   Ready    agent   9m37s   v1.24.9
```

As can see above, we were able to issue 'kubectl get nodes' and can see the 3 worker nodes. Now it is time to deploy the weka cluster.

Deploy WEKA via Terraform:

Using the Azure cloudshell, clone the WEKA Azure terraform repo:

```
alex [ ~ ]$ git clone [https://github.com/weka/terraform-azure-weka.git](https://github.com/weka/terraform-azu
re-weka.git)
Cloning into 'terraform-azure-weka'...
remote: Enumerating objects: 1224, done.
remote: Counting objects: 100% (586/586), done.
remote: Compressing objects: 100% (246/246), done.
remote: Total 1224 (delta 429), reused 348 (delta 336), pack-reused 638
Receiving objects: 100% (1224/1224), 266.58 KiB | 2.67 MiB/s, done.
Resolving deltas: 100% (756/756), done.
```

We will be using the 'public_network_existing_network' example, so cd into this directory:

```
alex [ ~ ]$ cd terraform-azure-weka/examples/public_network_existing_network/
alex [ ~/terraform-azure-weka/examples/public_network_existing_network ]$
```

There is only one file that needs to be modified, it is the tfvars.auto.vars file:

prefix = Should be a unique name (short)

rg_name = Should be the same resource group for which you deployed the AKS cluster into

vnet_name = Should be the name of of the vnet that was automatically generated in the Networking tab during the AKS cluster creation process. You can also find this by browsing to your resource group in the azure portal and looking at the vnets created there.

subnets_name_list = Likely there is only one subnet in this new vnet and by default it is called "default"

cluster_name = A simple, short cluster name

instance_type = Can be either Standard_L8s_v3 or Standard_L16s_v3

```
cat vars.auto.tfvars

prefix              = "alexz"
rg_name             = "weka-alexh-rg"
vnet_name           = "wekaalexhrgvnet852"
subnets_name_list   = ["default"]
cluster_name        = "alexz"
instance_type       = "Standard_L8s_v3"
set_obs_integration = true
tiering_ssd_percent = 20
cluster_size        = 6
```

Issue terraform init:

```
alex [ ~/terraform-azure-weka/examples/public_network_existing_network ]$ terraform init

Initializing the backend...
Initializing modules...

Initializing provider plugins...
- Finding latest version of hashicorp/null...
- Finding latest version of hashicorp/azuread...
- Finding latest version of hashicorp/tls...
- Finding latest version of hashicorp/local...
- Finding latest version of hashicorp/template...
- Finding latest version of hashicorp/archive...
- Finding hashicorp/azurerm versions matching "~> 3.26.0"...
- Installing hashicorp/template v2.2.0...
- Installed hashicorp/template v2.2.0 (signed by HashiCorp)
- Installing hashicorp/archive v2.3.0...
- Installed hashicorp/archive v2.3.0 (signed by HashiCorp)
- Installing hashicorp/azurerm v3.26.0...
- Installed hashicorp/azurerm v3.26.0 (signed by HashiCorp)
- Installing hashicorp/null v3.2.1...
- Installed hashicorp/null v3.2.1 (signed by HashiCorp)
- Installing hashicorp/azuread v2.36.0...
- Installed hashicorp/azuread v2.36.0 (signed by HashiCorp)
- Installing hashicorp/tls v4.0.4...
- Installed hashicorp/tls v4.0.4 (signed by HashiCorp)
- Installing hashicorp/local v2.4.0...
```

```
- Installed hashicorp/local v2.4.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

Issue 'terraform apply'. You will need to provide your get.weka.io token and your subscription ID from the portal.azure.com web page (Note: We issue ./terraform (dot followed by slash) in order to ensure we are running the new version of terraform that we downloaded and not the default system one):

```
terraform apply -var='get_weka_io_token=<your_get_weka_token>' -var='subscription_id=<your_subscription_id>'
```

Once the terraform apply is complete, you should see output close to:

```
Apply complete! Resources: 91 added, 0 changed, 0 destroyed.

Outputs:

DOWNLOAD-SSH-KEYS-COMMAND = <<EOT
############################## download ssh keys command from blob ##########################################
  CLUSTER: alexz
  az keyvault secret download --file private.pem --encoding utf-8 --vault-name  alexz-alexz-key-vault --name p
rivate-key --query "value"
  az keyvault secret download --file public.pub --encoding utf-8 --vault-name  alexz-alexz-key-vault --name pu
blic-key --query "value"

EOT
IPS = [
  "alexz-alexz-backend-0: 40.121.66.161",
  "alexz-alexz-backend-1: 40.76.35.144",
  "alexz-alexz-backend-2: 40.76.32.6",
  "alexz-alexz-backend-3: 40.76.34.164",
  "alexz-alexz-backend-4: 40.76.35.48",
  "alexz-alexz-backend-5: 40.76.34.126",
]
SSH-KEY-PATH = "/tmp/alexz-alexz-public-key.pub, /tmp/alexz-alexz-private-key.pem"
get-cluster-status = <<EOT
################################# get cluster status ############################################################
function_key=$(az functionapp keys list --name alexz-alexz-function-app --resource-group weka-alexh-rg --subsc
ription <subscription_id> --query functionKeys -o tsv)
curl https://alexz-alexz-function-app.azurewebsites.net/api/status?code=$function_key
```

It should take an additional 15 minutes or so for the cluster to come up in the background. Verify that the cluster is up by running the supplied cluster-status commands:

```
function_key=$(az functionapp keys list --name alexz-alexz-function-app --resource-group weka-alexh-rg --subsc
ription <subscription_id> --query functionKeys -o tsv)
curl https://alexz-alexz-function-app.azurewebsites.net/api/status?code=$function_key
```

```
EOT
➜  public_network_existing_network git:(main) ✗ function_key=$(az functionapp keys list --name alexz-alexz-fu
nction-app --resource-group weka-alexh-rg --subscription <subscription_id> --query functionKeys -o tsv)
➜  public_network_existing_network git:(main) ✗ curl https://alexz-alexz-function-app.azurewebsites.net/api/s
tatus\?code\=$function_key
{
  "initial_size": 6,
  "desired_size": 6,
  "instances": [
    "alexz-alexz-backend-4",
    "alexz-alexz-backend-0",
    "alexz-alexz-backend-3",
    "alexz-alexz-backend-1",
    "alexz-alexz-backend-5",
    "alexz-alexz-backend-2"
  ],
  "clusterized": false
}%
```

Using the addresses provided in the output, login to one of the cluster nodes using the private key that was created in /tmp:

```
➜  public_network_existing_network git:(main) ✗ ssh -i /tmp/alexz-alexz-private-key.pem weka@40.121.66.161

weka@alexz-alexz-backend-0:~$ weka cluster host
HOST ID  HOSTNAME              CONTAINER  IPS          STATUS  RELEASE   FAILURE DOMAIN  CORES  MEMORY   LAST FAILURE  UPTIME
0        alexz-alexz-backend-4  drives0    10.224.1.82  UP      4.1.0.71  AUTO            1      1.49 GB                0:07:04h
1        alexz-alexz-backend-0  drives0    10.224.1.79  UP      4.1.0.71  AUTO            1      1.49 GB                0:06:56h
2        alexz-alexz-backend-3  drives0    10.224.1.80  UP      4.1.0.71  AUTO            1      1.49 GB                0:07:06h
3        alexz-alexz-backend-1  drives0    10.224.1.77  UP      4.1.0.71  AUTO            1      1.49 GB                0:06:49h
4        alexz-alexz-backend-5  drives0    10.224.1.78  UP      4.1.0.71  AUTO            1      1.49 GB                0:06:40h
5        alexz-alexz-backend-2  drives0    10.224.1.81  UP      4.1.0.71  AUTO            1      1.49 GB                0:05:39h
6        alexz-alexz-backend-4  compute0   10.224.1.82  UP      4.1.0.71  AUTO            1      31 GB                  0:03:41h
7        alexz-alexz-backend-0  compute0   10.224.1.79  UP      4.1.0.71  AUTO            1      31 GB                  0:03:32h
8        alexz-alexz-backend-3  compute0   10.224.1.80  UP      4.1.0.71  AUTO            1      31 GB                  0:03:20h
9        alexz-alexz-backend-1  compute0   10.224.1.77  UP      4.1.0.71  AUTO            1      31 GB                  0:03:08h
10       alexz-alexz-backend-5  compute0   10.224.1.78  UP      4.1.0.71  AUTO            1      31 GB                  0:02:59h
11       alexz-alexz-backend-2  compute0   10.224.1.81  UP      4.1.0.71  AUTO            1      31 GB                  0:02:49h
12       alexz-alexz-backend-4  frontend0  10.224.1.82  UP      4.1.0.71  AUTO            1      1.47 GB                0:01:58h
13       alexz-alexz-backend-0  frontend0  10.224.1.79  UP      4.1.0.71  AUTO            1      1.47 GB                0:01:49h
14       alexz-alexz-backend-3  frontend0  10.224.1.80  UP      4.1.0.71  AUTO            1      1.47 GB                0:01:41h
15       alexz-alexz-backend-1  frontend0  10.224.1.77  UP      4.1.0.71  AUTO            1      1.47 GB                0:01:32h
16       alexz-alexz-backend-5  frontend0  10.224.1.78  UP      4.1.0.71  AUTO            1      1.47 GB                0:01:24h
17       alexz-alexz-backend-2  frontend0  10.224.1.81  UP      4.1.0.71  AUTO            1      1.47 GB                0:01:16h
```

Excellent. Now we have a running AKS cluster and a running weka cluster in the same vnet and subnet. Now we'll deploy our daemonset

Deploy a Daemonset:

NOTE - This daemonset is using an external docker container from github to allow the daemonsent operations on the AKS worker node - please refer for the following github link for more info https://github.com/patnaikshekhar/AKSNodeInstaller

A daemonset is a special type of pod that will run once on each worker node. We will use these special pods to install the weka software on the worker nodes.

Open up an Azure cloud shell and create a directory called k8s:

```
alex [ ~ ]$ mkdir k8s
```

Inside the k8s directory, populate a file named 'daemonset.yaml' with the following contents:

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: installer
  namespace: default
spec:
  selector:
    matchLabels:
      job: installer
  template:
    metadata:
      labels:
        job: installer
    spec:
      hostPID: true
      hostNetwork: true
      restartPolicy: Always
      containers:
      - image: patnaikshekhar/node-installer:1.3
        name: installer
        securityContext:
          privileged: true
        volumeMounts:
        - name: install-script
          mountPath: /tmp
        - name: host-mount
          mountPath: /host
      volumes:
      - name: install-script
        configMap:
          name: sample-installer-config
      - name: host-mount
        hostPath:
          path: /tmp/install
```

Next, go back to the console on your WEKA cluster and grab 3 IP addresses from 3 backends:

```
weka@alexz-alexz-backend-0:~$ weka cluster host -b -o ips --no-header | head -3
10.224.1.82
10.224.1.79
10.224.1.80
```

We will need these addresses to create the next file.

Create an additional file in this same directory called "config_map.yaml" with the following contents. Note - you NEED TO MODIFY THIS FILE with the addresses above. See below in the example, the first of the 3 addresses listed above I used for the curl command. For the weka mount command I used all 3 addresses:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: sample-installer-config
  namespace: default
data:
  install.sh: |
    #!/bin/bash
    # Update and install packages
    mkdir /weka
    ssh-keygen -t rsa -N '' -f ~/.ssh/id_rsa <<< y
    cat ~/.ssh/id_rsa.pub >> ~azureuser/.ssh/authorized_keys
    ip addr
    ssh -i ~/.ssh/id_rsa -o "StrictHostKeyChecking no" azureuser@localhost "curl http://10.224.1.82:14000/dist
/v1/install | sudo sh"
    ssh -i ~/.ssh/id_rsa -o "StrictHostKeyChecking no" azureuser@localhost "sudo mount -t wekafs -o net=udp,nu
m_cores=1 10.224.1.82,10.224.1.79,10.224.1.80/default /weka"
```

Now, cd out of the directory and issue a kubectl apply on the entire k8s directory:

```
alex [ ~ ]$ kubectl apply -f k8s
configmap/sample-installer-config created
daemonset.apps/installer created
```

We can now see the daemonset pods running, 1 on each node:

```
alex [ ~ ]$ kubectl get pods -o wide
NAME              READY   STATUS    RESTARTS   AGE    IP             NODE                                NOMINATED NODE   READINESS GATES
installer-bwbrd   1/1     Running   0          2m8s   10.224.0.113   aks-agentpool-23409389-vmss000001   <none>           <none>
installer-gtv4m   1/1     Running   0          2m8s   10.224.0.222   aks-agentpool-23409389-vmss000000   <none>           <none>
installer-rdjph   1/1     Running   0          2m8s   10.224.0.4     aks-agentpool-23409389-vmss000002   <none>           <none>
```

If we look at the logs for one of these pods, we can see that the client and mount has completed successfully:

```
alex [ ~ ]$ kubectl logs installer-bwbrd
Generating public/private rsa key pair.
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:qAxecvcxS1ZTUpIXKw4PSB2yIw60pSERvAIJjUDxNcc root@aks-agentpool-23409389-vmss000001
The key's randomart image is:
+---[RSA 2048]----+
|@O+ .o+o..oo+.   |
|++o=..oE. .+..   |
|. =.. + o +..    |
|.. o . o * o     |
|. o + o S o      |
| . * o + +       |
|  . o   o        |
|                 |
|                 |
+----[SHA256]-----+
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
```

```
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:0d:3a:8c:44:4c brd ff:ff:ff:ff:ff:ff
    inet 10.224.0.113/16 brd 10.224.255.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20d:3aff:fe8c:444c/64 scope link
       valid_lft forever preferred_lft forever
3: enP58549s1: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc mq master eth0 state UP group default ql
en 1000
    link/ether 00:0d:3a:8c:44:4c brd ff:ff:ff:ff:ff:ff
5: azv7c4f37356eb@if4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 10
00
    link/ether aa:aa:aa:aa:aa:aa brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet6 fe80::a8aa:aaff:feaa:aaaa/64 scope link
       valid_lft forever preferred_lft forever
7: azv55ab279cdf6@if6: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 10
00
    link/ether aa:aa:aa:aa:aa:aa brd ff:ff:ff:ff:ff:ff link-netnsid 1
    inet6 fe80::a8aa:aaff:feaa:aaaa/64 scope link
       valid_lft forever preferred_lft forever
9: azv283f14b0873@if8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 10
00
    link/ether aa:aa:aa:aa:aa:aa brd ff:ff:ff:ff:ff:ff link-netnsid 2
    inet6 fe80::a8aa:aaff:feaa:aaaa/64 scope link
       valid_lft forever preferred_lft forever
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1416  100  1416    0     0   345k      0 --:--:-- --:--:-- --:--:--  460k
Downloading WekaIO CLI 4.1.0.71
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 92.4M  100 92.4M    0     0   457M      0 --:--:-- --:--:-- --:--:--  457M
Installing...
Installing agent of version 4.1.0.71
Waiting for agent service to be ready
Installation finished successfully
WekaIO CLI 4.1.0.71 is now installed
Mounting 10.224.1.82,10.224.1.79,10.224.1.80/default on /weka
Basing mount on container client
Downloading [1/19] http://10.224.1.79:14000/dist/v1/image/weka-node-7dd65bb3b9240a695217fe1e98f1c96d.squashfs
Downloading [2/19] http://10.224.1.79:14000/dist/v1/image/container-samba-weka-4.7.12.3-95d7e0704b141138b43d76
b226f55eda.squashfs
Downloading [3/19] http://10.224.1.79:14000/dist/v1/image/dependencies-1.0.0-81dd3682be55880a5ef2b337bce9ae4d.
squashfs
Downloading [4/19] http://10.224.1.79:14000/dist/v1/image/weka-ganesha-18ca10b28151817a7c8bb7267e6f5e9d.squash
fs
Downloading [5/19] http://10.224.1.79:14000/dist/v1/image/weka-samba-fa23b1ed5b86200b1d682f072bc9af61.squashfs
Downloading [6/19] http://10.224.1.79:14000/dist/v1/image/driver-uio-pci-generic-1.0.0-d644841c998c88e4fc66529
e4484dbb6.squashfs
Downloading [7/19] http://10.224.1.79:14000/dist/v1/image/api-52da22b2cd0ed1b50978e57d509a47c6.squashfs
Downloading [8/19] http://10.224.1.79:14000/dist/v1/image/weka-driver-1.0.0-595420f10959c344dc93b1eff50bb016.s
quashfs
Downloading [9/19] http://10.224.1.79:14000/dist/v1/image/weka-smbw-b038d6b57ae8f593e98282b2709b69b1.squashfs
Downloading [10/19] http://10.224.1.79:14000/dist/v1/image/container-s3-weka-release-459ba907ea2ca2fc6c3940deb
a2dc631.squashfs
Downloading [11/19] http://10.224.1.79:14000/dist/v1/image/weka-container-2.3.0-9c60adc0f77d96f577211d30cbf9ef
3c.squashfs
Downloading [12/19] http://10.224.1.79:14000/dist/v1/image/container-ganesha-dev-weka-2-2a9043bc5a3669e6d3498a
134575e134.squashfs
```

```
Downloading [13/19] http://10.224.1.79:14000/dist/v1/image/ofed-1b295470b56ec067af7340f2cca7e27a.squashfs
Downloading [14/19] http://10.224.1.79:14000/dist/v1/image/weka-s3-152101ca875b1b3e68598a8436da01ba.squashfs
Downloading [15/19] http://10.224.1.79:14000/dist/v1/image/weka-hostside-be7022b2fb8d07e029e3ae414452a3e5.squa
shfs
Downloading [16/19] http://10.224.1.79:14000/dist/v1/image/dashboard-8dbfa0f8a99260cd06825181ea0eb6d6.squashfs
Downloading [17/19] http://10.224.1.79:14000/dist/v1/image/ui-1.0.0-5bc747765d326e6e1c3488285822f459.squashfs
Downloading [18/19] http://10.224.1.79:14000/dist/v1/image/weka-driver-igb-uio-4.0.0-b8dc002ff96443389fdef3f08
462b238.squashfs
Downloading [19/19] http://10.224.1.79:14000/dist/v1/image/container-smbw-weka-4.7.12.3-2f44067d8868530d8f7e86
d035fbb7a8.squashfs
Finished getting version 4.1.0.71
Creating Weka container 'client' in version 4.1.0.71
Applying resources
Starting container 'client'
Waiting for container 'client' to join cluster
client: Allocated core 2 to slot 1 on "aks-agentpool-23409389-vmss000001":"client" (1/1)
client: Starting hugepages allocation for "aks-agentpool-23409389-vmss000001":"client"
client: Container "aks-agentpool-23409389-vmss000001":"client" allocated 703 out of 703 required hugepages aft
er 1 retries
client: Allocated 1406MB hugepages memory from 1 NUMA nodes for "aks-agentpool-23409389-vmss000001":"client"
client: Bandwidth of "aks-agentpool-23409389-vmss000001":"client" set to unlimited
client: WekaFS driver attached by "NodeId<65534>" on "aks-agentpool-23409389-vmss000001":"client"
Container "client" is ready (pid = 181547)
Calling the mount command
Mount completed successfully
```

Furthermore, we can see on the WEKA cluster, 3 new clients have joined (the 3 worker nodes):

```
weka@alexz-alexz-backend-0:~$ weka cluster host | grep client
18       aks-agentpool-23409389-vmss000002  client     10.224.0.4    UP      4.1.0.71             0      1.46 GB              0:01:57h
19       aks-agentpool-23409389-vmss000000  client     10.224.0.222  UP      4.1.0.71             0      1.46 GB              0:01:55h
20       aks-agentpool-23409389-vmss000001  client     10.224.0.113  UP      4.1.0.71             0      1.46 GB              0:01:46h
```

Now that we have our daemonset running, we can scale the AKS cluster. In this example, doubling the number of worker nodes from 3 to 6:

```
alex [ ~ ]$ az aks scale -n alexh-nifty-k8s-cluster --resource-group weka-alexh-rg --node-count 6
The behavior of this command has been altered by the following extension: aks-preview
{
  "aadProfile": null,
  "addonProfiles": {
    "azureKeyvaultSecretsProvider": {
      "config": null,
      "enabled": false,
      "identity": null
    },
    "azurepolicy": {
      "config": null,
      "enabled": false,
      "identity": null
    },
    "httpApplicationRouting": {
      "config": null,
      "enabled": false,
      "identity": null
    },
    "omsAgent": {
      "config": {
        "logAnalyticsWorkspaceResourceID": "/subscriptions/<subscription_id>/resourcegroups/defaultresourcegro
up-weu/providers/microsoft.operationalinsights/workspaces/defaultworkspace-<subscription_id>-weu"
      },
```

```
      "enabled": true,
      "identity": {
        "clientId": "fb014aee-817f-488c-bfbe-94a106ffa409",
        "objectId": "9439df06-6b14-4aff-a652-860eedcd9815",
        "resourceId": "/subscriptions/<subscription_id>/resourcegroups/MC_weka-alexh-rg_alexh-nifty-k8s-cluste
r_eastus/providers/Microsoft.ManagedIdentity/userAssignedIdentities/omsagent-alexh-nifty-k8s-cluster"
      }
    }
  },
  "agentPoolProfiles": [
    {
      "availabilityZones": [
        "1"
      ],
      "capacityReservationGroupId": null,
      "count": 6,
      "creationData": null,
      "currentOrchestratorVersion": "1.24.9",
      "enableAutoScaling": false,
      "enableCustomCaTrust": false,
      "enableEncryptionAtHost": null,
      "enableFips": false,
      "enableNodePublicIp": false,
      "enableUltraSsd": null,
      "gpuInstanceProfile": null,
      "hostGroupId": null,
      "kubeletConfig": null,
      "kubeletDiskType": "OS",
      "linuxOsConfig": null,
      "maxCount": null,
      "maxPods": 110,
      "messageOfTheDay": null,
      "minCount": null,
      "mode": "System",
      "name": "agentpool",
      "networkProfile": null,
      "nodeImageVersion": "AKSUbuntu-1804gen2containerd-2023.02.15",
      "nodeLabels": null,
      "nodePublicIpPrefixId": null,
      "nodeTaints": null,
      "orchestratorVersion": "1.24.9",
      "osDiskSizeGb": 128,
      "osDiskType": "Ephemeral",
      "osSku": "Ubuntu",
      "osType": "Linux",
      "podSubnetId": null,
      "powerState": {
        "code": "Running"
      },
      "provisioningState": "Succeeded",
      "proximityPlacementGroupId": null,
      "scaleDownMode": null,
      "scaleSetEvictionPolicy": null,
      "scaleSetPriority": null,
      "spotMaxPrice": null,
      "tags": null,
      "type": "VirtualMachineScaleSets",
      "upgradeSettings": null,
      "vmSize": "Standard_D8s_v3",
      "vnetSubnetId": "/subscriptions/<subscription_id>/resourceGroups/weka-alexh-rg/providers/Microsoft.Netwo
rk/virtualNetworks/wekaalexhrgvnet852/subnets/default",
```

```
      "windowsProfile": null,
      "workloadRuntime": null
    }
  ],
  "apiServerAccessProfile": null,
  "autoScalerProfile": null,
  "autoUpgradeProfile": {
    "nodeOsUpgradeChannel": null,
    "upgradeChannel": "none"
  },
  "azureMonitorProfile": null,
  "azurePortalFqdn": "alexh-nifty-k8s-cluster-dns-bjhtunbv.portal.hcp.eastus.azmk8s.io",
  "creationData": null,
  "currentKubernetesVersion": "1.24.9",
  "disableLocalAccounts": false,
  "diskEncryptionSetId": null,
  "dnsPrefix": "alexh-nifty-k8s-cluster-dns",
  "enableNamespaceResources": null,
  "enablePodSecurityPolicy": null,
  "enableRbac": true,
  "extendedLocation": null,
  "fqdn": "alexh-nifty-k8s-cluster-dns-bjhtunbv.hcp.eastus.azmk8s.io",
  "fqdnSubdomain": null,
  "guardrailsProfile": null,
  "httpProxyConfig": null,
  "id": "/subscriptions/<subscription_id>/resourcegroups/weka-alexh-rg/providers/Microsoft.ContainerService/ma
nagedClusters/alexh-nifty-k8s-cluster",
  "identity": {
    "principalId": "579a5460-d4ba-4886-ada6-402565283556",
    "tenantId": "93ba0df2-e204-4bfc-99ef-cb9e273ce33f",
    "type": "SystemAssigned",
    "userAssignedIdentities": null
  },
  "identityProfile": {
    "kubeletidentity": {
      "clientId": "c256a96f-d598-4b89-8951-2ddc71e6bd08",
      "objectId": "4387613e-c8cf-4866-80c7-d55777661c77",
      "resourceId": "/subscriptions/<subscription_id>/resourcegroups/MC_weka-alexh-rg_alexh-nifty-k8s-cluster_
eastus/providers/Microsoft.ManagedIdentity/userAssignedIdentities/alexh-nifty-k8s-cluster-agentpool"
    }
  },
  "ingressProfile": null,
  "kubernetesVersion": "1.24.9",
  "linuxProfile": null,
  "location": "eastus",
  "maxAgentPools": 100,
  "name": "alexh-nifty-k8s-cluster",
  "networkProfile": {
    "dnsServiceIp": "10.0.0.10",
    "dockerBridgeCidr": "172.17.0.1/16",
    "ebpfDataplane": null,
    "ipFamilies": [
      "IPv4"
    ],
    "kubeProxyConfig": null,
    "loadBalancerProfile": {
      "allocatedOutboundPorts": null,
      "backendPoolType": "nodeIPConfiguration",
      "effectiveOutboundIPs": [
        {
          "id": "/subscriptions/<subscription_id>/resourceGroups/MC_weka-alexh-rg_alexh-nifty-k8s-cluster_east
```

```
us/providers/Microsoft.Network/publicIPAddresses/ce2cc73e-3780-4b51-8c78-59dd75494c44",
          "resourceGroup": "MC_weka-alexh-rg_alexh-nifty-k8s-cluster_eastus"
        }
      ],
      "enableMultipleStandardLoadBalancers": null,
      "idleTimeoutInMinutes": null,
      "managedOutboundIPs": {
        "count": 1,
        "countIpv6": null
      },
      "outboundIPs": null,
      "outboundIpPrefixes": null
    },
    "loadBalancerSku": "Standard",
    "natGatewayProfile": null,
    "networkMode": null,
    "networkPlugin": "azure",
    "networkPluginMode": null,
    "networkPolicy": "azure",
    "outboundType": "loadBalancer",
    "podCidr": null,
    "podCidrs": null,
    "serviceCidr": "10.0.0.0/16",
    "serviceCidrs": [
      "10.0.0.0/16"
    ]
  },
  "nodeResourceGroup": "MC_weka-alexh-rg_alexh-nifty-k8s-cluster_eastus",
  "nodeResourceGroupProfile": null,
  "oidcIssuerProfile": {
    "enabled": false,
    "issuerUrl": null
  },
  "podIdentityProfile": null,
  "powerState": {
    "code": "Running"
  },
  "privateFqdn": null,
  "privateLinkResources": null,
  "provisioningState": "Succeeded",
  "publicNetworkAccess": null,
  "resourceGroup": "weka-alexh-rg",
  "securityProfile": {
    "azureKeyVaultKms": null,
    "customCaTrustCertificates": null,
    "defender": null,
    "imageCleaner": null,
    "nodeRestriction": null,
    "workloadIdentity": null
  },
  "servicePrincipalProfile": {
    "clientId": "msi",
    "secret": null
  },
  "sku": {
    "name": "Basic",
    "tier": "Paid"
  },
  "storageProfile": {
    "blobCsiDriver": null,
    "diskCsiDriver": {
```

```
      "enabled": true,
      "version": "v1"
    },
    "fileCsiDriver": {
      "enabled": true
    },
    "snapshotController": {
      "enabled": true
    }
  },
  "systemData": null,
  "tags": null,
  "type": "Microsoft.ContainerService/ManagedClusters",
  "upgradeSettings": null,
  "windowsProfile": {
    "adminPassword": null,
    "adminUsername": "azureuser",
    "enableCsiProxy": true,
    "gmsaProfile": null,
    "licenseType": null
  },
  "workloadAutoScalerProfile": {
    "keda": null,
    "verticalPodAutoscaler": null
  }
}
```

Now, we can see that we have 3 additional nodes and 3 additional installer pods (from the daemonset):

```
alex [ ~ ]$ kubectl get nodes
NAME                                STATUS   ROLES   AGE     VERSION
aks-agentpool-23409389-vmss000000   Ready    agent   137m    v1.24.9
aks-agentpool-23409389-vmss000001   Ready    agent   137m    v1.24.9
aks-agentpool-23409389-vmss000002   Ready    agent   137m    v1.24.9
aks-agentpool-23409389-vmss000003   Ready    agent   5m1s    v1.24.9
aks-agentpool-23409389-vmss000004   Ready    agent   5m16s   v1.24.9
aks-agentpool-23409389-vmss000005   Ready    agent   4m58s   v1.24.9

alex [ ~ ]$ kubectl get pods -o wide
NAME              READY   STATUS    RESTARTS   AGE     IP             NODE                                NOMINATED NODE   READINESS GATE
S
installer-9xrhv   1/1     Running   0          4m46s   10.224.1.83    aks-agentpool-23409389-vmss000005   <none>           <none>
installer-bwbrd   1/1     Running   0          13m     10.224.0.113   aks-agentpool-23409389-vmss000001   <none>           <none>
installer-gtv4m   1/1     Running   0          13m     10.224.0.222   aks-agentpool-23409389-vmss000000   <none>           <none>
installer-rdjph   1/1     Running   0          13m     10.224.0.4     aks-agentpool-23409389-vmss000002   <none>           <none>
installer-rk5vv   1/1     Running   0          4m53s   10.224.1.192   aks-agentpool-23409389-vmss000003   <none>           <none>
installer-rmlcs   1/1     Running   0          5m10s   10.224.2.45    aks-agentpool-23409389-vmss000004   <none>           <none>
```

Furthermore, we now see the additional worker nodes have joined the WEKA cluster:

```
weka@alexz-alexz-backend-0:~$ weka cluster host | grep client
18       aks-agentpool-23409389-vmss000002  client     10.224.0.4    UP      4.1.0.71            0      1.46 GB             0:12:48h
19       aks-agentpool-23409389-vmss000000  client     10.224.0.222  UP      4.1.0.71            0      1.46 GB             0:12:46h
20       aks-agentpool-23409389-vmss000001  client     10.224.0.113  UP      4.1.0.71            0      1.46 GB             0:12:37h
21       aks-agentpool-23409389-vmss000004  client     10.224.2.45   UP      4.1.0.71            0      1.46 GB             0:02:45h
22       aks-agentpool-23409389-vmss000003  client     10.224.1.192  UP      4.1.0.71            0      1.46 GB             0:02:19h
23       aks-agentpool-23409389-vmss000005  client     10.224.1.83   UP      4.1.0.71            0      1.46 GB             0:02:18h
```

It is now time to install the WEKA CSI driver.

Deploy WEKA CSI plugin on AKS:

From the azure cloud console, issue the following commands to deploy the CSI plugin:

```
helm repo add csi-wekafs https://weka.github.io/csi-wekafs

helm install csi-wekafs csi-wekafs/csi-wekafsplugin --namespace csi-wekafs --create-namespace
NAME: csi-wekafs
LAST DEPLOYED: Fri Mar 10 21:29:44 2023
NAMESPACE: csi-wekafs
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
Thank you for installing csi-wekafsplugin.

Your release is named csi-wekafs.
The release is installed in namespace csi-wekafs

To learn more about the release, try:

  $ helm status -n csi-wekafs csi-wekafs
  $ helm get all -n csi-wekafs csi-wekafs

Official Weka CSI Plugin documentation can be found here: https://docs.weka.io/appendix/weka-csi-plugin

Examples on how to configure a storage class and start using the driver are here:
https://github.com/weka/csi-wekafs/tree/master/examples
```

You should see that a new namespace has been created:

```
alex [ ~ ]$ kubectl get namespaces | grep csi-weka
csi-wekafs        Active   59s
```

Run FIO in a StatefulSet:

Go to the Azure cloudshell and create a new directory called 'weka_app':

```
alex [ ~ ]$ mkdir weka_app
```

cd to this directory and create a file named 'secrets.yaml'. This is how we will communicate with the WEKA cluster. Note, there are a number of parameters here that we need to base64 encode. It is important to remember that when you use the 'echo' command you must use the '-n' flag to prevent it from adding a newline character as you don't want to add the newline character to your base64 encoded string. For the IP addresses listed, I'm using the 3 IP addresses of the backends that I collected toward the beginning of this document. In my particular case, I've not yet changed the cluster password so the user/login is admin/admin

```
alex [ ~/weka_app ]$ echo -n admin | base64
YWRtaW4=
alex [ ~/weka_app ]$ echo -n Root | base64
Um9vdA==
alex [ ~/weka_app ]$ echo -n 10.224.1.82:14000,10.224.1.79:14000,10.224.1.80:14000 | base64
MTAuMjI0LjEuODI6MTQwMDAsMTAuMjI0LjEuNzk6MTQwMDAsMTAuMjI0LjEuODA6MTQwMDA=
alex [ ~/weka_app ]$ echo -n http | base64
aHR0cA==

alex [ ~/weka_app ]$ cat secrets.yaml
apiVersion: v1
kind: Secret
metadata:
  name: csi-wekafs-api-secret
  namespace: csi-wekafs
type: Opaque
data:
  username: YWRtaW4=
  password: YWRtaW4=
  organization: Um9vdA==
  endpoints: MTAuMjI0LjEuODI6MTQwMDAsMTAuMjI0LjEuNzk6MTQwMDAsMTAuMjI0LjEuODA6MTQwMDA=
  scheme: aHR0cA==
```

Now that we have a proper secrets.yaml file, issue kubectly apply against it:

```
alex [ ~/weka_app ]$ kubectl apply -f secrets.yaml
secret/csi-wekafs-api-secret created
```

Next, we will create a storage class. Save the following contents to a file, sc_wekafs_dir.yaml:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: storageclass-wekafs-dir-api
provisioner: csi.weka.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
allowVolumeExpansion: true
parameters:
  volumeType: dir/v1
  filesystemName: default
  capacityEnforcement: HARD
  # optional parameters setting UID, GID and permissions on volume
  # UID of the volume owner, default 0 (root)
```

```
  #ownerUid: "1000"
  # GID of the volume owner, default 0 (root)
  #ownerGid: "1000"
  # permissions in Unix octal format, default "0750"
  #permissions: "0775"
  # name of the secret that stores API credentials for a cluster
  # change the name of secret to match secret of a particular cluster (if you have several Weka clusters)
  csi.storage.k8s.io/provisioner-secret-name: &secretName csi-wekafs-api-secret
  # change the name of the namespace in which the cluster API credentials
  csi.storage.k8s.io/provisioner-secret-namespace: &secretNamespace csi-wekafs
  # do not change anything below this line, or set to same parameters as above
  csi.storage.k8s.io/controller-publish-secret-name: *secretName
  csi.storage.k8s.io/controller-publish-secret-namespace: *secretNamespace
  csi.storage.k8s.io/controller-expand-secret-name: *secretName
  csi.storage.k8s.io/controller-expand-secret-namespace: *secretNamespace
  csi.storage.k8s.io/node-stage-secret-name: *secretName
  csi.storage.k8s.io/node-stage-secret-namespace: *secretNamespace
  csi.storage.k8s.io/node-publish-secret-name: *secretName
  csi.storage.k8s.io/node-publish-secret-namespace: *secretNamespace
```

Apply this file as well:

```
alex [ ~/weka_app ]$ kubectl apply -f sc_wekafs_dir.yaml
storageclass.storage.k8s.io/storageclass-wekafs-dir-api created
```

Next, we'll create a persistent volume claim. Create a file, pvc.yaml with the following contents:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-wekafs-dir
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: storageclass-wekafs-dir-api
  volumeMode: Filesystem
  resources:
    requests:
      storage: 10Ti
```

Apply this file as well:

```
kubectl apply -f pvc.yaml
```

It may take a few moments, but we should see that the pvc is in status Bound:

```
alex [ ~/weka_app ]$ kubectl get pvc
NAME           STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS                AGE
pvc-wekafs-dir   Bound    pvc-ab92bb35-543f-4417-8e27-1dd6626f53bb   10Ti       RWX            storageclass-wekafs-dir-api   25s
```

If the pvc status remains in pending for more than 20 seconds, check the logs of the PVC. This is an example of a working, bound pvc:

```
alex [ ~/weka_app ]$ kubectl describe pvc pvc-wekafs-dir
Name:          pvc-wekafs-dir
Namespace:     default
StorageClass:  storageclass-wekafs-dir-api
Status:        Bound
Volume:        pvc-ab92bb35-543f-4417-8e27-1dd6626f53bb
```

```
Labels:        <none>
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
               volume.beta.kubernetes.io/storage-provisioner: csi.weka.io
               volume.kubernetes.io/storage-provisioner: csi.weka.io
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      10Ti
Access Modes:  RWX
VolumeMode:    Filesystem
Used By:       <none>
Events:
  Type    Reason                Age                   From                                                                      Message
  ----    ------                ----                  ----                                                                      -------
  Normal  Provisioning          2m39s                 csi.weka.io_csi-wekafs-controller-0_fce26c98-bf4d-48a7-9074-2914f32b6ecf  Externa
l provisioner is provisioning volume for claim "default/pvc-wekafs-dir"
  Normal  ExternalProvisioning  2m37s (x2 over 2m39s)  persistentvolume-controller                                               waiting
for a volume to be created, either by external provisioner "csi.weka.io" or manually created by system administrator
  Normal  ProvisioningSucceeded  2m34s                 csi.weka.io_csi-wekafs-controller-0_fce26c98-bf4d-48a7-9074-2914f32b6ecf  Success
fully provisioned volume pvc-ab92bb35-543f-4417-8e27-1dd6626f53bb
```

The most common error will be user/pass incorrect type error messages due to providing incorrect data in the secrets.yaml base64 encodings. Typically, forgetting to use '-n' with echo will cause this issue....

Now that we have a persistent volume claim, we can create our FIO stateful set. Create a new file 'fio_app.yaml' with the following contents (You can modify parameters accordingly):

```
kind: StatefulSet
apiVersion: apps/v1
metadata:
  name: fio-app
spec:
  serviceName: fio-app
  replicas: 1
  selector:
    matchLabels:
      app: fio-app
  template:
    metadata:
      labels:
        app: fio-app
    spec:
      terminationGracePeriodSeconds: 1800
      containers:
        - name: fio-app
          image: xridge/fio
          volumeMounts:
          - mountPath: "/data"
            name: fio-app-volume
          command: ["fio"]
          args: ["--filename_format=$(NODE_NAME)-$(POD_NAME)-$jobnum",
                 "--directory=/data/",
                 "--direct=1",
                 "--rw=randread",
                 "--bs=1m",
                 "--numjobs=4",
                 "--time_based",
                 "--runtime=500",
                 "--group_reporting",
                 "--name=benchtest",
                 "--size=1G"]
          env:
```

```
            - name: NODE_NAME
              valueFrom:
                fieldRef:
                  fieldPath: spec.nodeName
            - name: POD_NAME
              valueFrom:
                fieldRef:
                  fieldPath: metadata.name

      volumes:
        - name: fio-app-volume
          persistentVolumeClaim:
            claimName: pvc-wekafs-dir # defined in pvc-wekafs-dir.yaml
```

Now apply the stateful set:

```
kubectl apply -f fio_app.yaml
```

At this point, we can see a single fio_app pod running:

```
alex [ ~/weka_app ]$ kubectl get pods -o wide
NAME             READY   STATUS    RESTARTS   AGE   IP             NODE                                NOMINATED NODE   READINESS GATES
fio-app-0        1/1     Running   0          90s   10.224.2.19    aks-agentpool-23409389-vmss000003   <none>           <none>
installer-9xrhv  1/1     Running   0          35m   10.224.1.83    aks-agentpool-23409389-vmss000005   <none>           <none>
installer-bwbrd  1/1     Running   0          43m   10.224.0.113   aks-agentpool-23409389-vmss000001   <none>           <none>
installer-gtv4m  1/1     Running   0          43m   10.224.0.222   aks-agentpool-23409389-vmss000000   <none>           <none>
installer-rdjph  1/1     Running   0          43m   10.224.0.4     aks-agentpool-23409389-vmss000002   <none>           <none>
installer-rk5vv  1/1     Running   0          35m   10.224.1.192   aks-agentpool-23409389-vmss000003   <none>           <none>
installer-rmlcs  1/1     Running   0          35m   10.224.2.45    aks-agentpool-23409389-vmss000004   <none>           <none>
```

We can verify that we have I/Os to the WEKA cluster:

```
weka@alexz-alexz-backend-0:~$ weka status
WekaIO v4.1.0.71 (CLI build 4.1.0.71)

       cluster: alexz (7e3e5033-4804-4698-ae4e-a99d8c3568ef)
        status: OK (18 backend containers UP, 6 drives UP)
    protection: 3+2
     hot spare: 1 failure domains (965.25 GiB)
 drive storage: 4.71 TiB total
         cloud: connected
       license: Unlicensed

     io status: STARTED 1 hour ago (18 io-nodes UP, 138 Buckets UP)
    link layer: Ethernet
       clients: 6 connected
         reads: 237.03 MiB/s (237 IO/s)
        writes: 0 B/s (0 IO/s)
    operations: 238 ops/s
        alerts: 3 active alerts, use `weka alerts` to list them
```

Furthermore, we can see the files created on the weka filesystem:

```
root@alexz-alexz-backend-0:~# mkdir /weka
root@alexz-alexz-backend-0:~# mount -t wekafs default /weka
Mounting default on /weka
Basing mount on container frontend0
This is legacy mount, calling mount directly
Calling the mount command
Mount completed successfully
root@alexz-alexz-backend-0:~# cd /weka/csi-volumes/pvc-ab92bb35-543f-4417-8e27-1dd6626f53bb-3c50c891515c1d52fc
```

```
b3a7134220470fa5435ab3/
root@alexz-alexz-backend-0:/weka/csi-volumes/pvc-ab92bb35-543f-4417-8e27-1dd6626f53bb-3c50c891515c1d52fcb3a713
4220470fa5435ab3# ls
aks-agentpool-23409389-vmss000003-fio-app-0-0  aks-agentpool-23409389-vmss000003-fio-app-0-1  aks-agentpool-23
409389-vmss000003-fio-app-0-2  aks-agentpool-23409389-vmss000003-fio-app-0-3
root@alexz-alexz-backend-0:/weka/csi-volumes/pvc-ab92bb35-543f-4417-8e27-1dd6626f53bb-3c50c891515c1d52fcb3a713
4220470fa5435ab3#
```

Now, let's go ahead and scale up the stateful set. We'll increase from 1 to 6 which should result in running a single FIO pod on each of the 6 worker nodes:

```
alex [ ~/weka_app ]$ kubectl scale statefulsets fio-app --replicas=6
statefulset.apps/fio-app scaled
```

We now have 6 fio-app pods, 1 per worker node:

```
alex [ ~/weka_app ]$ kubectl get pods -o wide
NAME              READY   STATUS    RESTARTS   AGE     IP             NODE                                NOMINATED NODE   READINESS GATE
S
fio-app-0         1/1     Running   0          6m20s   10.224.2.19    aks-agentpool-23409389-vmss000003   <none>           <none>
fio-app-1         1/1     Running   0          57s     10.224.0.143   aks-agentpool-23409389-vmss000001   <none>           <none>
fio-app-2         1/1     Running   0          52s     10.224.1.134   aks-agentpool-23409389-vmss000005   <none>           <none>
fio-app-3         1/1     Running   0          45s     10.224.0.53    aks-agentpool-23409389-vmss000002   <none>           <none>
fio-app-4         1/1     Running   0          39s     10.224.1.16    aks-agentpool-23409389-vmss000000   <none>           <none>
fio-app-5         1/1     Running   0          31s     10.224.2.73    aks-agentpool-23409389-vmss000004   <none>           <none>
installer-9xrhv   1/1     Running   0          40m     10.224.1.83    aks-agentpool-23409389-vmss000005   <none>           <none>
installer-bwbrd   1/1     Running   0          48m     10.224.0.113   aks-agentpool-23409389-vmss000001   <none>           <none>
installer-gtv4m   1/1     Running   0          48m     10.224.0.222   aks-agentpool-23409389-vmss000000   <none>           <none>
installer-rdjph   1/1     Running   0          48m     10.224.0.4     aks-agentpool-23409389-vmss000002   <none>           <none>
installer-rk5vv   1/1     Running   0          40m     10.224.1.192   aks-agentpool-23409389-vmss000003   <none>           <none>
installer-rmlcs   1/1     Running   0          40m     10.224.2.45    aks-agentpool-23409389-vmss000004   <none>           <none>
```

We can also see the increased load on the WEKA cluster:

```
weka@alexz-alexz-backend-0:~$ weka status
WekaIO v4.1.0.71 (CLI build 4.1.0.71)

       cluster: alexz (7e3e5033-4804-4698-ae4e-a99d8c3568ef)
        status: OK (18 backend containers UP, 6 drives UP)
    protection: 3+2
     hot spare: 1 failure domains (965.25 GiB)
 drive storage: 4.71 TiB total
         cloud: connected
       license: Unlicensed

     io status: STARTED 1 hour ago (18 io-nodes UP, 138 Buckets UP)
    link layer: Ethernet
       clients: 6 connected
         reads: 1.46 GiB/s (1496 IO/s)
        writes: 0 B/s (0 IO/s)
    operations: 1497 ops/s
        alerts: 3 active alerts, use `weka alerts` to list them
```

```
weka@alexz-alexz-backend-0:~$ weka stats realtime -s cpu
NODE ID  WRITES/S  WRITE  WRITE LATENCY(µS)  READS/S  READ     READ LATENCY(µS)  OPS/S  CPU%  L6 RECV      L6 SENT      OBS UPLOAD  OBS DOWNLOAD  RDMA RECV  RDMA SENT
301      0.00      0 B/s  0.00               0.00     0 B/s    0.00              0.00   0.03  3.23 KB/s    6.02 KB/s    0 B/s       0 B/s         0 B/s      0 B/s
281      0.00      0 B/s  0.00               0.00     0 B/s    0.00              0.00   0.04  10.29 KB/s   5.78 KB/s    0 B/s       0 B/s         0 B/s      0 B/s
341      0.00      0 B/s  0.00               0.00     0 B/s    0.00              0.00   0.05  11.45 KB/s   23.34 KB/s   0 B/s       0 B/s         0 B/s      0 B/s
241      0.00      0 B/s  0.00               0.00     0 B/s    0.00              0.00   0.05  11.62 KB/s   23.41 KB/s   0 B/s       0 B/s         0 B/s      0 B/s
321      0.00      0 B/s  0.00               0.00     0 B/s    0.00              0.00   0.05  5.65 KB/s    25.05 KB/s   0 B/s       0 B/s         0 B/s      0 B/s
261      0.00      0 B/s  0.00               0.00     0 B/s    0.00              0.00   0.93  4.45 KB/s    9.45 KB/s    0 B/s       0 B/s         0 B/s      0 B/s
181      0.00      0 B/s  0.00               0.00     0 B/s    0.00              0.00   1.93  927.70 KB/s  2.05 MB/s    0 B/s       0 B/s         0 B/s      0 B/s
141      0.00      0 B/s  0.00               0.00     0 B/s    0.00              0.00   2.18  1.45 MB/s    2.46 MB/s    0 B/s       0 B/s         0 B/s      0 B/s
201      0.00      0 B/s  0.00               0.00     0 B/s    0.00              0.00   2.21  1.02 MB/s    2.25 MB/s    0 B/s       0 B/s         0 B/s      0 B/s
161      0.00      0 B/s  0.00               0.00     0 B/s    0.00              0.00   2.25  918.33 KB/s  2.20 MB/s    0 B/s       0 B/s         0 B/s      0 B/s
121      0.00      0 B/s  0.00               0.00     0 B/s    0.00              0.00   2.42  1.41 MB/s    2.69 MB/s    0 B/s       0 B/s         0 B/s      0 B/s
221      0.00      0 B/s  0.00               0.00     0 B/s    0.00              0.00   2.53  1.89 MB/s    2.93 MB/s    0 B/s       0 B/s         0 B/s      0 B/s
61       0.00      0 B/s  0.00               0.00     0 B/s    0.00              0.00   44.56 1.66 MB/s    232.26 MB/s  0 B/s       0 B/s         0 B/s      0 B/s
```

```
81      0.00      0 B/s  0.00              0.00     0 B/s        0.00             0.00    48.04  1.75 MB/s    246.44 MB/s  0 B/s       0 B/s        0 B/s      0 B/s
101     0.00      0 B/s  0.00              0.00     0 B/s        0.00             0.00    50.66  1.72 MB/s    253.00 MB/s  0 B/s       0 B/s        0 B/s      0 B/s
21      0.00      0 B/s  0.00              0.00     0 B/s        0.00             0.00    50.80  1.84 MB/s    271.25 MB/s  0 B/s       0 B/s        0 B/s      0 B/s
1       0.00      0 B/s  0.00              0.00     0 B/s        0.00             0.00    58.37  1.98 MB/s    278.23 MB/s  0 B/s       0 B/s        0 B/s      0 B/s
41      0.00      0 B/s  0.00              0.00     0 B/s        0.00             0.00    60.84  2.05 MB/s    278.97 MB/s  0 B/s       0 B/s        0 B/s      0 B/s
361     0.00      0 B/s  0.00            213.85   224.24 MB/s  18102.23          213.85   91.92  234.74 MB/s  1.40 MB/s    0 B/s       0 B/s        0 B/s      0 B/s
441     0.00      0 B/s  0.00            237.07   248.59 MB/s  16448.45          237.07   93.16  259.61 MB/s  1.76 MB/s    0 B/s       0 B/s        0 B/s      0 B/s
401     0.00      0 B/s  0.00            238.02   249.58 MB/s  16145.40          238.02   93.49  261.70 MB/s  1.66 MB/s    0 B/s       0 B/s        0 B/s      0 B/s
381     0.00      0 B/s  0.00            263.02   275.80 MB/s  14860.94          263.02   94.01  287.56 MB/s  2.05 MB/s    0 B/s       0 B/s        0 B/s      0 B/s
421     0.00      0 B/s  0.00            252.02   264.26 MB/s  15444.52          252.02   94.30  276.41 MB/s  1.90 MB/s    0 B/s       0 B/s        0 B/s      0 B/s
461     0.00      0 B/s  0.00            259.01   271.59 MB/s  15070.87          259.01   94.61  283.29 MB/s  2.14 MB/s    0 B/s       0 B/s        0 B/s      0 B/s
```

Now, let's go ahead and scale the AKS cluster down from the 6 nodes to 3 nodes:

```
alex [ ~/weka_app ]$ az aks scale -n alexh-nifty-k8s-cluster --resource-group weka-alexh-rg --node-count 3
The behavior of this command has been altered by the following extension: aks-preview
{
  "aadProfile": null,
  "addonProfiles": {
    "azureKeyvaultSecretsProvider": {
      "config": null,
      "enabled": false,
      "identity": null
    },
    "azurepolicy": {
      "config": null,
      "enabled": false,
      "identity": null
    },
    "httpApplicationRouting": {
      "config": null,
      "enabled": false,
      "identity": null
    },
    "omsAgent": {
      "config": {
        "logAnalyticsWorkspaceResourceID": "/subscriptions/<subscription_id>/resourcegroups/defaultresourcegro
up-weu/providers/microsoft.operationalinsights/workspaces/defaultworkspace-<subscription_id>-weu"
      },
      "enabled": true,
      "identity": {
        "clientId": "fb014aee-817f-488c-bfbe-94a106ffa409",
        "objectId": "9439df06-6b14-4aff-a652-860eedcd9815",
        "resourceId": "/subscriptions/<subscription_id>/resourcegroups/MC_weka-alexh-rg_alexh-nifty-k8s-cluste
r_eastus/providers/Microsoft.ManagedIdentity/userAssignedIdentities/omsagent-alexh-nifty-k8s-cluster"
      }
    }
  },
  "agentPoolProfiles": [
    {
      "availabilityZones": [
        "1"
      ],
      "capacityReservationGroupId": null,
      "count": 3,
      "creationData": null,
      "currentOrchestratorVersion": "1.24.9",
      "enableAutoScaling": false,
      "enableCustomCaTrust": false,
      "enableEncryptionAtHost": null,
      "enableFips": false,
      "enableNodePublicIp": false,
      "enableUltraSsd": null,
      "gpuInstanceProfile": null,
      "hostGroupId": null,
```

```
      "kubeletConfig": null,
      "kubeletDiskType": "OS",
      "linuxOsConfig": null,
      "maxCount": null,
      "maxPods": 110,
      "messageOfTheDay": null,
      "minCount": null,
      "mode": "System",
      "name": "agentpool",
      "networkProfile": null,
      "nodeImageVersion": "AKSUbuntu-1804gen2containerd-2023.02.15",
      "nodeLabels": null,
      "nodePublicIpPrefixId": null,
      "nodeTaints": null,
      "orchestratorVersion": "1.24.9",
      "osDiskSizeGb": 128,
      "osDiskType": "Ephemeral",
      "osSku": "Ubuntu",
      "osType": "Linux",
      "podSubnetId": null,
      "powerState": {
        "code": "Running"
      },
      "provisioningState": "Succeeded",
      "proximityPlacementGroupId": null,
      "scaleDownMode": null,
      "scaleSetEvictionPolicy": null,
      "scaleSetPriority": null,
      "spotMaxPrice": null,
      "tags": null,
      "type": "VirtualMachineScaleSets",
      "upgradeSettings": null,
      "vmSize": "Standard_D8s_v3",
      "vnetSubnetId": "/subscriptions/<subscription_id>/resourceGroups/weka-alexh-rg/providers/Microsoft.Netwo
rk/virtualNetworks/wekaalexhrgvnet852/subnets/default",
      "windowsProfile": null,
      "workloadRuntime": null
    }
  ],
  "apiServerAccessProfile": null,
  "autoScalerProfile": null,
  "autoUpgradeProfile": {
    "nodeOsUpgradeChannel": null,
    "upgradeChannel": "none"
  },
  "azureMonitorProfile": null,
  "azurePortalFqdn": "alexh-nifty-k8s-cluster-dns-bjhtunbv.portal.hcp.eastus.azmk8s.io",
  "creationData": null,
  "currentKubernetesVersion": "1.24.9",
  "disableLocalAccounts": false,
  "diskEncryptionSetId": null,
  "dnsPrefix": "alexh-nifty-k8s-cluster-dns",
  "enableNamespaceResources": null,
  "enablePodSecurityPolicy": null,
  "enableRbac": true,
  "extendedLocation": null,
  "fqdn": "alexh-nifty-k8s-cluster-dns-bjhtunbv.hcp.eastus.azmk8s.io",
  "fqdnSubdomain": null,
  "guardrailsProfile": null,
  "httpProxyConfig": null,
  "id": "/subscriptions/<subscription_id>/resourcegroups/weka-alexh-rg/providers/Microsoft.ContainerService/ma
```

```
nagedClusters/alexh-nifty-k8s-cluster",
  "identity": {
    "principalId": "579a5460-d4ba-4886-ada6-402565283556",
    "tenantId": "93ba0df2-e204-4bfc-99ef-cb9e273ce33f",
    "type": "SystemAssigned",
    "userAssignedIdentities": null
  },
  "identityProfile": {
    "kubeletidentity": {
      "clientId": "c256a96f-d598-4b89-8951-2ddc71e6bd08",
      "objectId": "4387613e-c8cf-4866-80c7-d55777661c77",
      "resourceId": "/subscriptions/<subscription_id>/resourcegroups/MC_weka-alexh-rg_alexh-nifty-k8s-cluster_
eastus/providers/Microsoft.ManagedIdentity/userAssignedIdentities/alexh-nifty-k8s-cluster-agentpool"
    }
  },
  "ingressProfile": null,
  "kubernetesVersion": "1.24.9",
  "linuxProfile": null,
  "location": "eastus",
  "maxAgentPools": 100,
  "name": "alexh-nifty-k8s-cluster",
  "networkProfile": {
    "dnsServiceIp": "10.0.0.10",
    "dockerBridgeCidr": "172.17.0.1/16",
    "ebpfDataplane": null,
    "ipFamilies": [
      "IPv4"
    ],
    "kubeProxyConfig": null,
    "loadBalancerProfile": {
      "allocatedOutboundPorts": null,
      "backendPoolType": "nodeIPConfiguration",
      "effectiveOutboundIPs": [
        {
          "id": "/subscriptions/<subscription_id>/resourceGroups/MC_weka-alexh-rg_alexh-nifty-k8s-cluster_east
us/providers/Microsoft.Network/publicIPAddresses/ce2cc73e-3780-4b51-8c78-59dd75494c44",
          "resourceGroup": "MC_weka-alexh-rg_alexh-nifty-k8s-cluster_eastus"
        }
      ],
      "enableMultipleStandardLoadBalancers": null,
      "idleTimeoutInMinutes": null,
      "managedOutboundIPs": {
        "count": 1,
        "countIpv6": null
      },
      "outboundIPs": null,
      "outboundIpPrefixes": null
    },
    "loadBalancerSku": "Standard",
    "natGatewayProfile": null,
    "networkMode": null,
    "networkPlugin": "azure",
    "networkPluginMode": null,
    "networkPolicy": "azure",
    "outboundType": "loadBalancer",
    "podCidr": null,
    "podCidrs": null,
    "serviceCidr": "10.0.0.0/16",
    "serviceCidrs": [
      "10.0.0.0/16"
    ]
```

```
},
"nodeResourceGroup": "MC_weka-alexh-rg_alexh-nifty-k8s-cluster_eastus",
"nodeResourceGroupProfile": null,
"oidcIssuerProfile": {
  "enabled": false,
  "issuerUrl": null
},
"podIdentityProfile": null,
"powerState": {
  "code": "Running"
},
"privateFqdn": null,
"privateLinkResources": null,
"provisioningState": "Succeeded",
"publicNetworkAccess": null,
"resourceGroup": "weka-alexh-rg",
"securityProfile": {
  "azureKeyVaultKms": null,
  "customCaTrustCertificates": null,
  "defender": null,
  "imageCleaner": null,
  "nodeRestriction": null,
  "workloadIdentity": null
},
"servicePrincipalProfile": {
  "clientId": "msi",
  "secret": null
},
"sku": {
  "name": "Basic",
  "tier": "Paid"
},
"storageProfile": {
  "blobCsiDriver": null,
  "diskCsiDriver": {
    "enabled": true,
    "version": "v1"
  },
  "fileCsiDriver": {
    "enabled": true
  },
  "snapshotController": {
    "enabled": true
  }
},
"systemData": null,
"tags": null,
"type": "Microsoft.ContainerService/ManagedClusters",
"upgradeSettings": null,
"windowsProfile": {
  "adminPassword": null,
  "adminUsername": "azureuser",
  "enableCsiProxy": true,
  "gmsaProfile": null,
  "licenseType": null
},
"workloadAutoScalerProfile": {
  "keda": null,
```

```
      "verticalPodAutoscaler": null
    }
}
```

As expected, we see 3 out of the 6 worker nodes have been removed. We now see a total of 3 daemonset pods (remember, we only run a single daemonset pod on every work node) and we see a total of 6 fio-app pods. Two pods running per worker node. Notice the difference in the runtime due to the pods restarting on the remaining worker nodes:

```
alex [ ~/weka_app ]$ kubectl get pods -o wide
NAME              READY   STATUS    RESTARTS   AGE     IP             NODE                                NOMINATED NODE   READINESS GATE
S
fio-app-0         1/1     Running   0          109s    10.224.0.55    aks-agentpool-23409389-vmss000002   <none>           <none>
fio-app-1         1/1     Running   0          7m7s    10.224.0.143   aks-agentpool-23409389-vmss000001   <none>           <none>
fio-app-2         1/1     Running   0          108s    10.224.0.230   aks-agentpool-23409389-vmss000000   <none>           <none>
fio-app-3         1/1     Running   0          6m55s   10.224.0.53    aks-agentpool-23409389-vmss000002   <none>           <none>
fio-app-4         1/1     Running   0          6m49s   10.224.1.16    aks-agentpool-23409389-vmss000000   <none>           <none>
fio-app-5         1/1     Running   0          106s    10.224.0.138   aks-agentpool-23409389-vmss000001   <none>           <none>
installer-bwbrd   1/1     Running   0          54m     10.224.0.113   aks-agentpool-23409389-vmss000001   <none>           <none>
installer-gtv4m   1/1     Running   0          54m     10.224.0.222   aks-agentpool-23409389-vmss000000   <none>           <none>
installer-rdjph   1/1     Running   0          54m     10.224.0.4     aks-agentpool-23409389-vmss000002   <none>           <none>
```

Cleanup

The environment should be cleaned up in the following order:

1. Issue terraform destroy on the WEKA cluster
2. Go into the azure portal web console and delete the AKS cluster

© 2019-2023 All rights reserved. WekaIO, WekaFS, WIN, Weka Innovation Network, the Weka brand mark, the Weka logo, and Radically Simple Storage are trademarks of WekaIO, Inc. and its affiliates in the United States and/or other countries. Other trademarks are the property of their respective companies. References in this publication to WekaIO's products, programs, or services do not imply that WekaIO intends to make these available in all countries in which it operates. Product specifications provided are sample specifications and do not constitute a warranty. Information is true as of the date of publication and is subject to change. Actual specifications for unique part numbers may vary.
WKA350–01 04/2023