

# **DEVOPS with MULTI-CLOUD**

## **Practice Tasks**

**Institute Name** : V Cube software solutions  
**Course** : DevOps with Multi-Cloud  
**Batch** : 30  
**Trainer** : Krishna reddy sir

**Prepared by** : G.Bhavish  
(MCD-AZ30-024)

## **TASK-18 :- Auto Scaling.**

**Date :** 12/02/26

### **Objective :-**

To automatically increase or decrease resources based on workload demand in order to maintain performance and optimize cost.

### **Auto Scaling :-**

Auto Scaling in Azure is a cloud capability that dynamically adjusts the number of running instances (VMs or services) based on predefined rules like CPU usage or schedule, ensuring high availability and cost efficiency.

→ There are two types of scalings :-

- **Vertical Scaling :-**

- It will upgrade the h/w size
- Eg:- standard b1\_s ⇒ d2s\_v3
- 1cpu & 1gb RAM ⇒ 2cpu & 8gb RAM
- For this type of upgrade we will use vertical scaling.
- Vertical scaling mainly for the db related server, file server.

- Horizontal Scaling :-

- Adding machines automatically is called horizontal scaling.
- In horizontal scaling we use Vmss.
- The horizontal Scaling is used for web servers(since they don't have data, they have only web pages.)
- When there is a huge load the machines are automatically created and when load is decreased the vm's are deleted.

→ Why vmss is only for web servers, why not DB?.

- if there is a huge load on servers, the instances will create automatically, then also the data will be stored.
- So if we use this for DB, the data will be lost, because after the decrease of load the instances will be deleted automatically.

→ In the vmss the machines will be added and deleted automatically, this process is done based on the "scaling conditions".

→ we use cpu metric condition in the scaling condition i.e

Eg:-

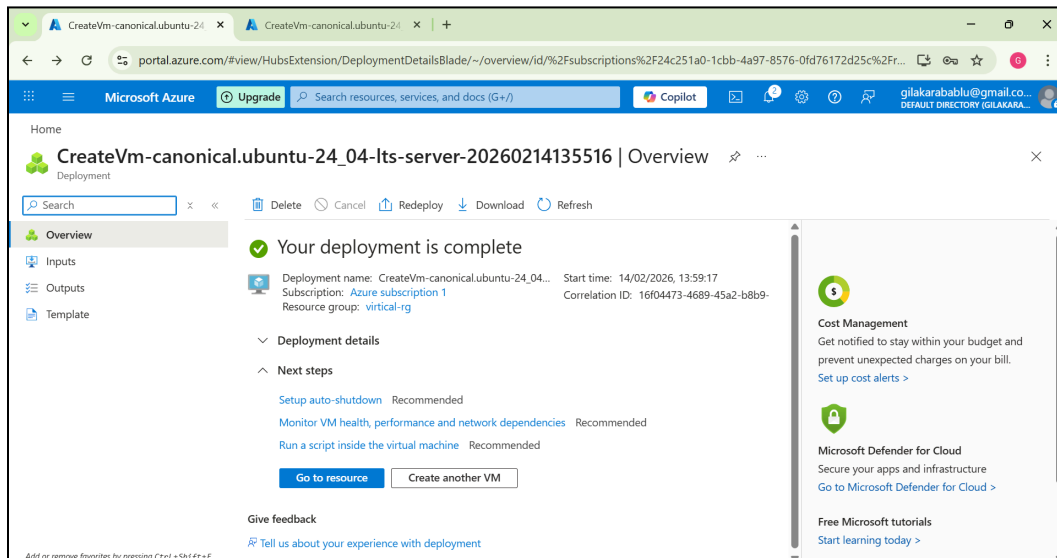
if cpu% is  $\geq 70\%$  then 3 machines should add

if cpu% is  $\leq 20\%$  then 3 machines should delete.

## □ Vertical scaling :-

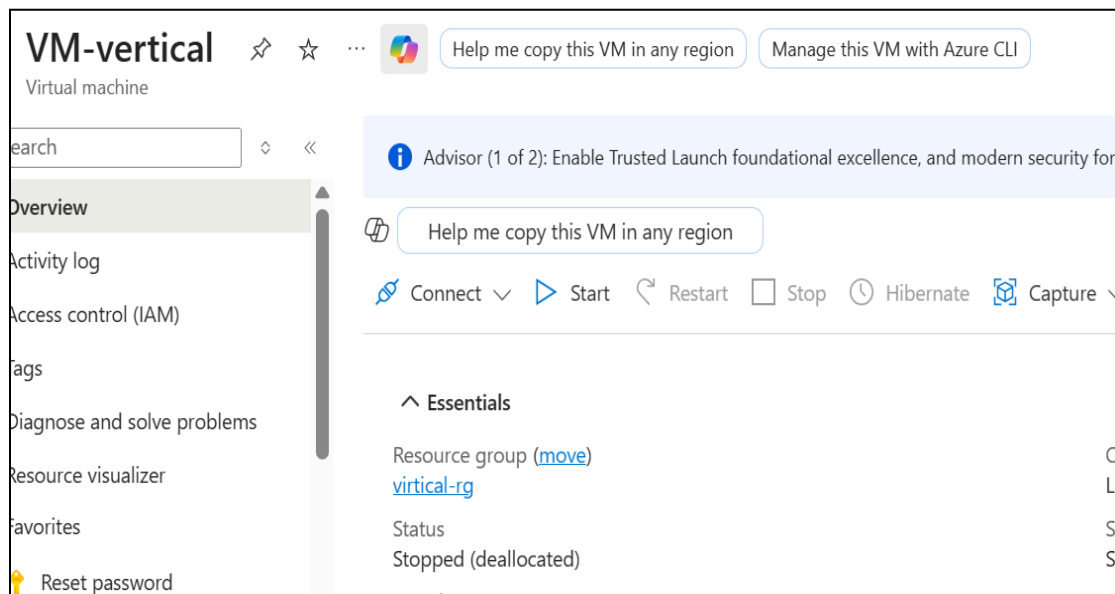
→ Create a virtual machine,

Vertical-rg > vm-vertical > os-linux/windows.



fig(1) virtual machine is created.

→ To upgrade the size, we need to stop the machine and upgrade(to ignore any issues.)



fig(2) the machine is stopped.

Operating system
Linux
Size
Standard DC2s v3 (2 vcpus, 16 GiB memory)
Primary NIC public IP
<a href="#">20.120.76.137</a>
<a href="#">1 associated public IPs</a>
Virtual network/subnet
<a href="#">vnet-eastus/snet-eastus-1</a>
DNS name
<a href="#">Not configured</a>

fig(3) before upgrade.

Operating system
Linux
Size
Standard DC4ds v3 (4 vcpus, 32 GiB memory)
Primary NIC public IP
<a href="#">20.120.76.137</a>
<a href="#">1 associated public IPs</a>
Virtual network/subnet
<a href="#">vnet-eastus/snet-eastus-1</a>
DNS name
<a href="#">Not configured</a>

fig(4) after upgrade.

## □ Horizontal Scaling :-

→ To implement the horizontal scaling we use the vmss - Virtual Machine Scale Set.

Search vmss > +create

Name ↑	Computer name	Status	Type	Provisioning st...	Size
<a href="#">VM-Original_2eee4158</a>	...	Running	VM	Succeeded	Standard_D2s_v5
<a href="#">VM-Original_3ef050cc</a>	...	Running	VM	Succeeded	Standard_D2s_v5

fig(5) vmss is created.

→ we can see defaultly 2 instances are created, coz it is default rule we can change it in scaling condition.

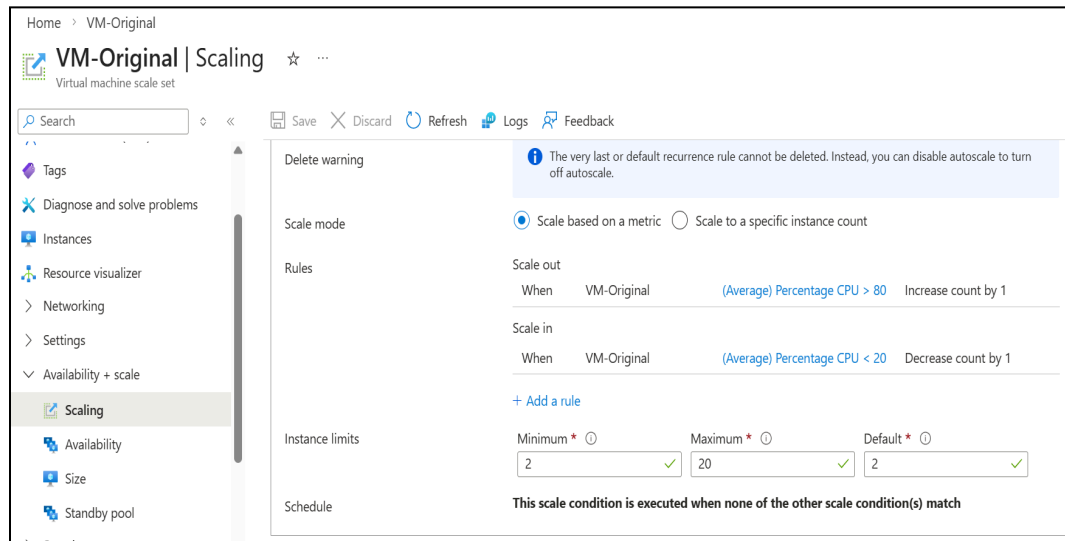
→ To change :-

Original vm > scaling > configure.

Change (instance limits)

Min - 1, max - 20, default - 1

→ now there will be only one instance.



fig(6) scaling in vmss.

→ writing scaling condition :-

Original vm > scaling > configure > default condition

**Scale out :-**

- How many vm want to increase based on condition.
- Increasing the number of instances (VMs or services) to handle more traffic or load.

**Scale in :-**

- How many vm want to decrease based on condition.
- Decreasing the number of instances when the load is low to save cost and resources.

☆ ...
Save ✕ Discard ↻ Refresh 📄 Logs 💬 Feedback

---

Delete warning

---

Scale mode

☒ Scale based on a metric

---

Rules

Scale out

When	VM-Origin
------	-----------

Scale in

When

VM-Origin

+ Add a rule

---

Instance limits

Minimum \* ⓘ

Schedule

## Scale rule

Percentage CPU (Maximum)

☐ Enable metric divide by instance count ⓘ

Operator \*

Metric threshold to trigger scale action \* ⓘ

Duration (minutes) \* ⓘ

Time grain (minutes) ⓘ

Time grain statistic \* ⓘ

Time aggregation \* ⓘ

Action

Operation \*

Cool down (minutes) \* ⓘ

instance count \*

✓

This scale condition is

[Update](#)
[Delete](#)

☆

⋮

Save

Discard

Refresh

Logs

Feedback

Delete warning

ⓘ

The very last or de off autoscale.

Scale mode

●

Scale based on a m

Rules

Scale out

When

VM-Origin

Scale in

When

VM-Origin

+ Add a rule

Instance limits

Minimum ⓘ

1

Schedule

This scale condition is

+ Add a scale condition

Scale rule

Percentage CPU (Minimum)

13.8 %

☐

Enable metric divide by instance count ⓘ

Operator \*

Less than or equal to

Metric threshold to trigger scale action \* ⓘ

20

%

Duration (minutes) \* ⓘ

1

Time grain (minutes) ⓘ

1

Time grain statistic \* ⓘ

Average

Time aggregation \* ⓘ

Minimum

⚙️ Action

Operation \*

Decrease count by

Cool down (minutes) \* ⓘ

1

instance count \*

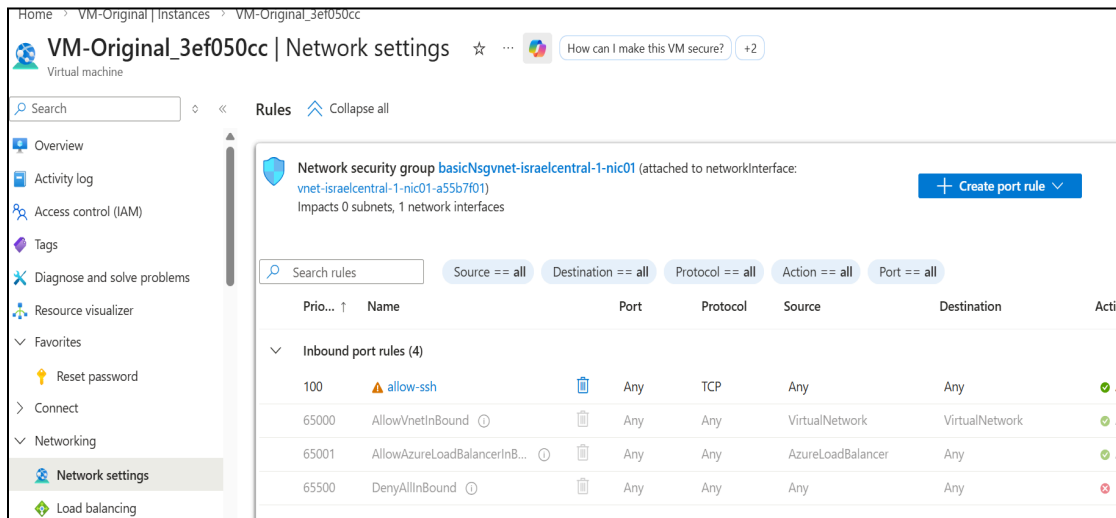
3

✓

Update

Delete

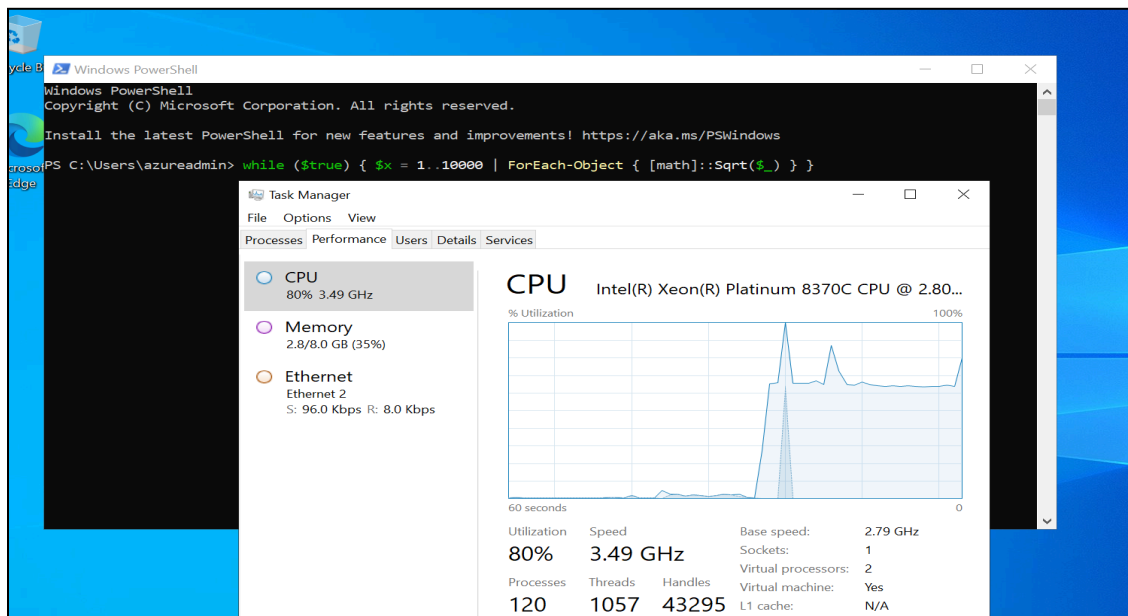
→ Now edit the nsg rules to allow all traffic.



fig(9) nsg rules allowing all traffic.

→ note:- we also created azure load balancer in the n/w while creating vmss.

→ now login to the machine and increase the cpu performance i.e apply stress.

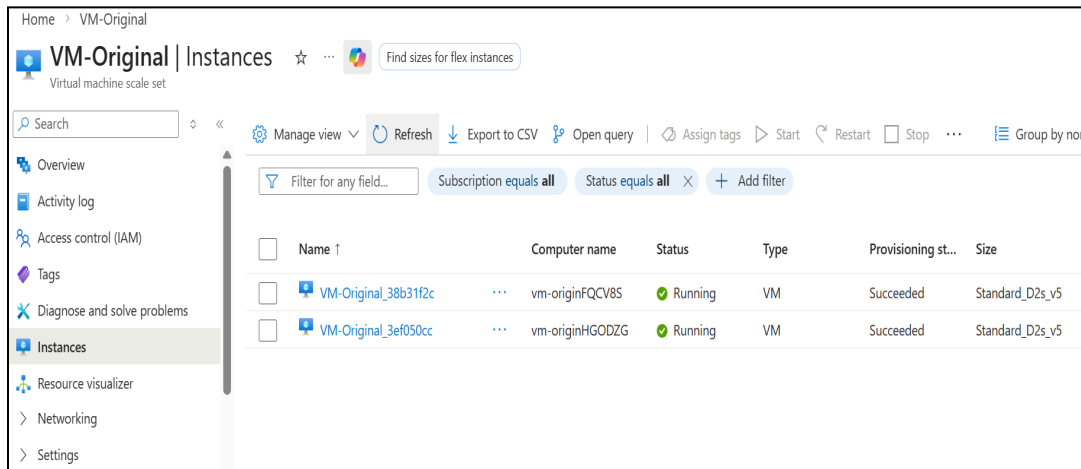


fig(10) applied stress.



→ Here we are increasing stress in the windows machine.  
By running the below command in the windows powershell.

```
while ($true) { $x = 1..100000 | ForEach-Object {  
[math]::Sqrt($_) } }
```



fig(11) one instance is created automatically due to high performance of cpu.

**Note :- To apply stress for linux machines.**

```
$ Sudo su
```

```
#apt update
```

```
#Apt install stress
```

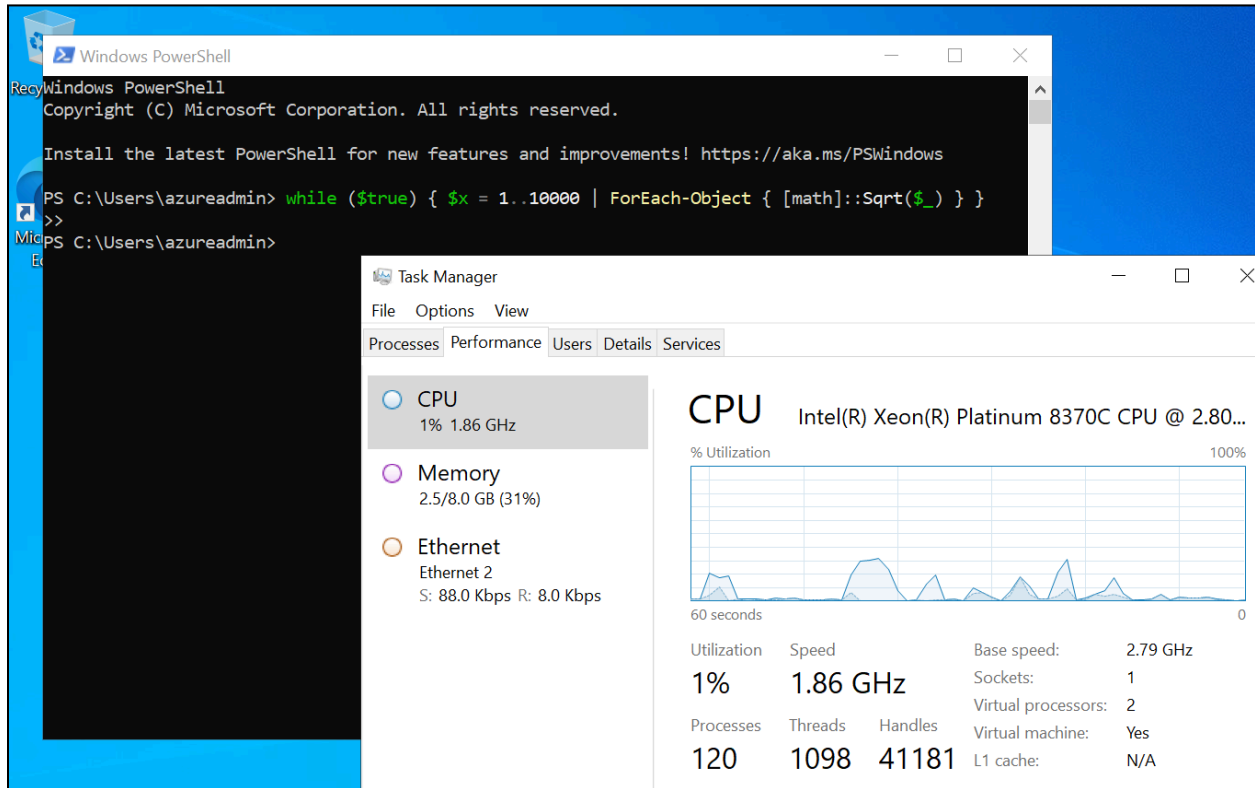
```
#stress
```

We will get an example (change time 10 or 20min)  
copy and paste.

```
Now we can check the cpu %
```

```
#htop (shows the cpu%)
```

→ Now after the decrease of the load the instances will be deleted automatically, type **ctrl+c** in windows powershell for removing stress.



fig(12) removed stress hence cpu % is decreased.

The screenshot shows the Azure portal interface for a virtual machine scale set named 'VM-Original'. The 'Instances' tab is selected, displaying a table of the current instances.

Name	Computer name	Status	Type	Provisioning st...	Size
VM-Original_5c4b2652	vm-originPLW7DE	Running	VM	Succeeded	Standard_D2s_v5

fig(13) since no increase in cpu %, the instances are deleted.

-----X-----