Resource Group and Networking

The first step in our infrastructure deployment is to create a resource group, which acts as a logical container for our Azure resources. We define a resource group using the azurerm_resource_group resource in Terraform, specifying its name and location.

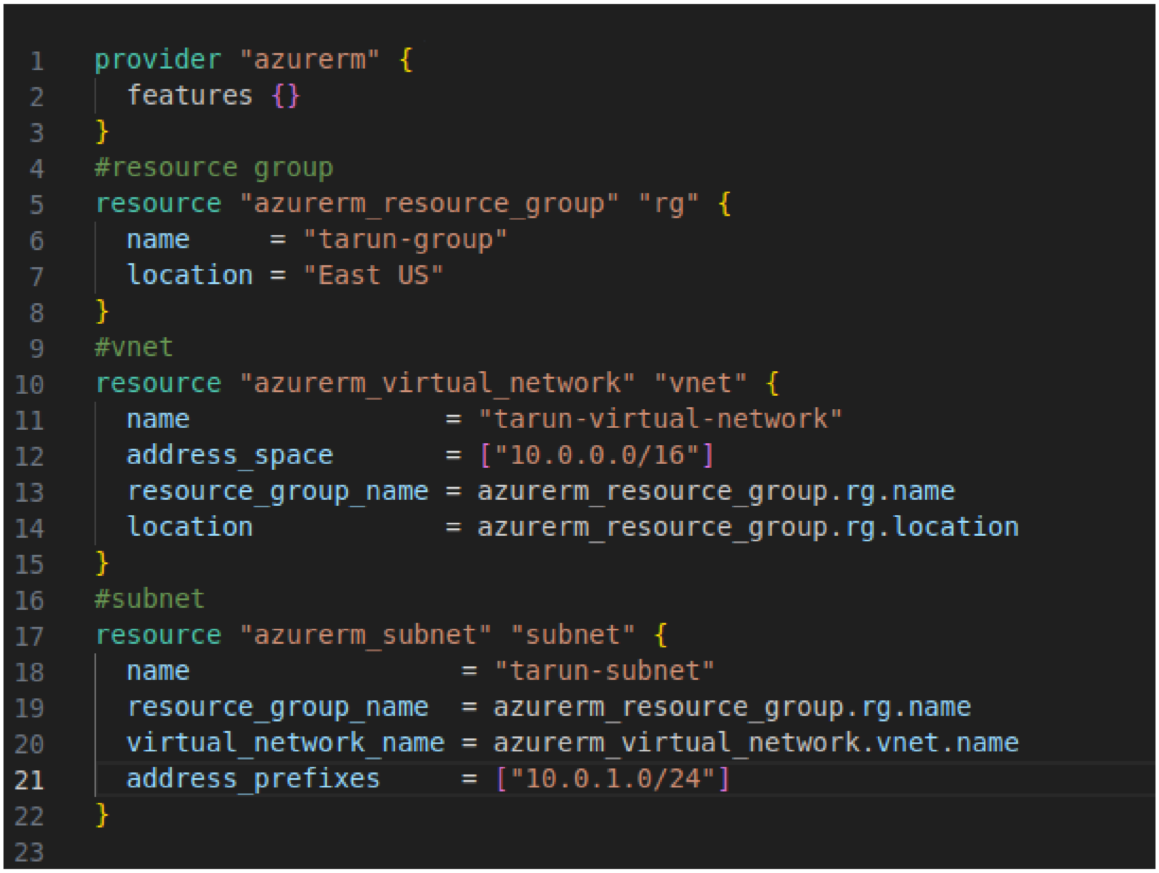
Next, we set up the networking components. We create a virtual network (azurerm_virtual_network) and a subnet (azurerm_subnet) within that virtual network. These resources define the network configuration for our infrastructure, such as IP address ranges and connectivity rules. We associate the subnet with the virtual network and the resource group.

Provider: The code specifies the Azure provider and enables features.

Resource Group: The azurerm_resource_group resource creates an Azure resource group with the specified name and location.

Virtual Network: The azurerm_virtual_network resource creates a virtual network with the specified name, address space, and associated resource group and location.

Subnet: The azurerm_subnet resource creates a subnet within the virtual network with the specified name, address prefix, and associated resource group and virtual network.



```
provider "azurerm" {
  features {}
}
#resource group
resource "azurerm_resource_group" "rg" {
  name     = "tarun-group"
  location = "East US"
}
#vnet
resource "azurerm_virtual_network" "vnet" {
  name                = "tarun-virtual-network"
  address_space       = ["10.0.0.0/16"]
  resource_group_name = azurerm_resource_group.rg.name
  location            = azurerm_resource_group.rg.location
}
#subnet
resource "azurerm_subnet" "subnet" {
  name                 = "tarun-subnet"
  resource_group_name  = azurerm_resource_group.rg.name
  virtual_network_name = azurerm_virtual_network.vnet.name
  address_prefixes     = ["10.0.1.0/24"]
}

```

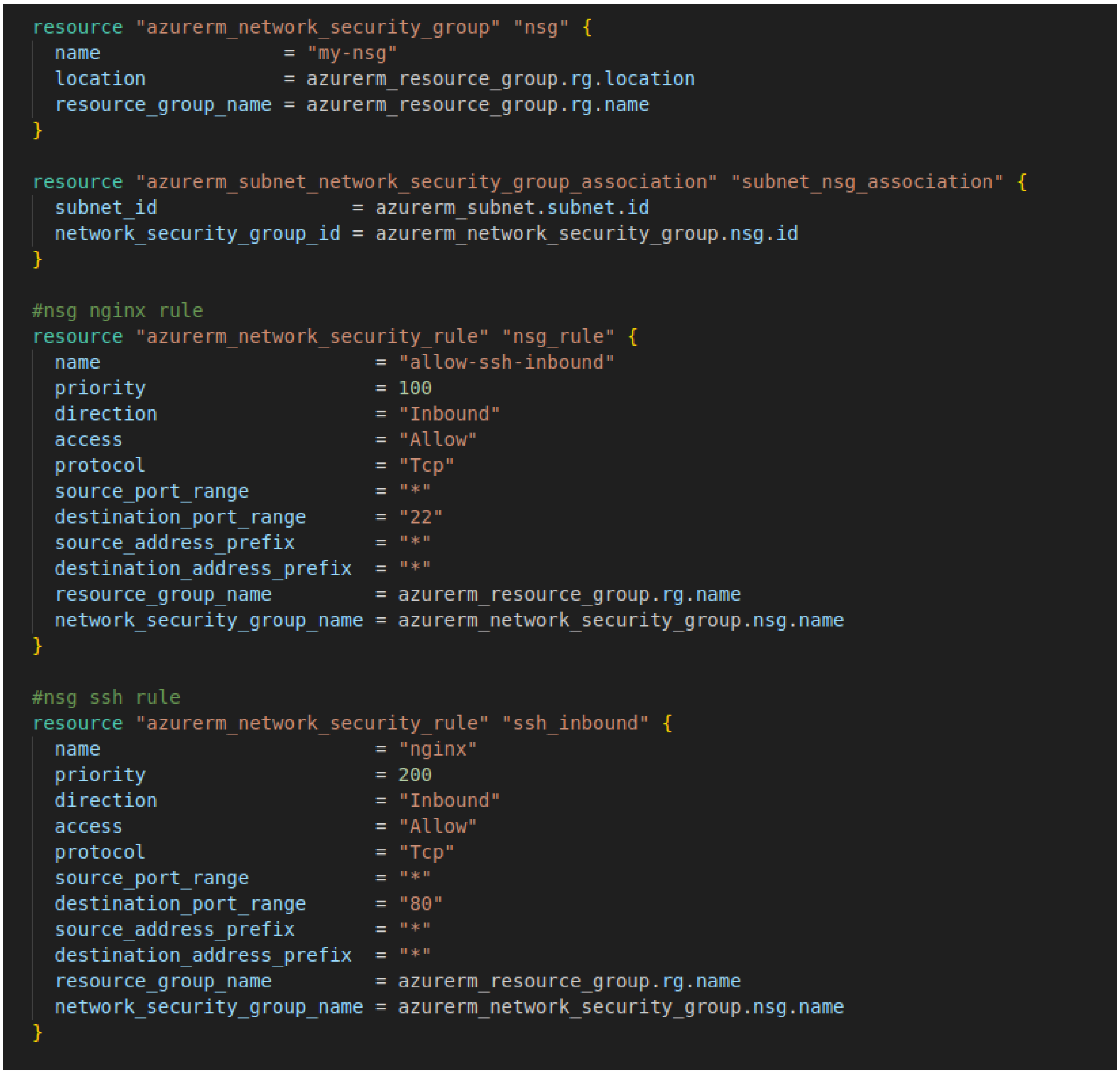
Associating subnet with the NSG rules

To control inbound and outbound traffic to our infrastructure, we configure a network security group (azurerm_network_security_group) and define security rules using azurerm_network_security_rule. These rules allow or deny specific network traffic based on protocols, ports, and IP addresses. We associate the network security group with the subnet using azurerm_subnet_network_security_group_association.

Network Security Group (NSG): The azurerm_network_security_group resource creates a network security group with the specified name, location, and associated resource group.

NSG Subnet Association: The azurerm_subnet_network_security_group_association resource associates the subnet created earlier with the network security group.

Network Security Rule (NSG Rule): The azurerm_network_security_rule resource creates a network security rule within the network security group. Two rules are created: one for SSH inbound traffic (port 22) and another for inbound traffic on port 80.

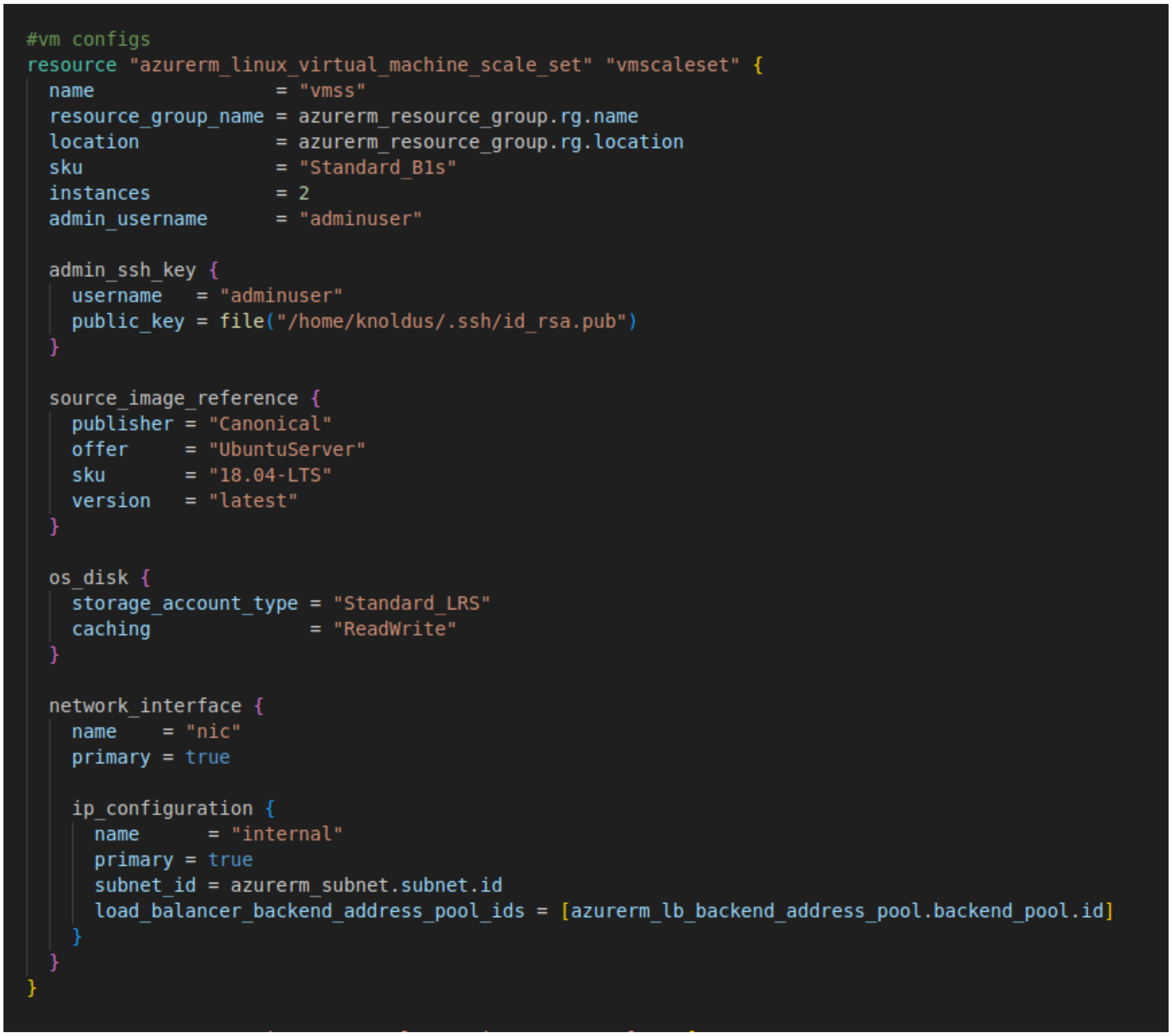


```
resource "azurerm_network_security_group" "nsg" {
  name                = "my-nsg"
  location            = azurerm_resource_group.rg.location
  resource_group_name = azurerm_resource_group.rg.name
}

resource "azurerm_subnet_network_security_group_association" "subnet_nsg_association" {
  subnet_id                 = azurerm_subnet.subnet.id
  network_security_group_id = azurerm_network_security_group.nsg.id
}

#nsg nginx rule
resource "azurerm_network_security_rule" "nsg_rule" {
  name                        = "allow-ssh-inbound"
  priority                    = 100
  direction                   = "Inbound"
  access                      = "Allow"
  protocol                    = "Tcp"
  source_port_range           = "*"
  destination_port_range      = "22"
  source_address_prefix       = "*"
  destination_address_prefix  = "*"
  resource_group_name         = azurerm_resource_group.rg.name
  network_security_group_name = azurerm_network_security_group.nsg.name
}

#nsg ssh rule
resource "azurerm_network_security_rule" "ssh_inbound" {
  name                        = "nginx"
  priority                    = 200
  direction                   = "Inbound"
  access                      = "Allow"
  protocol                    = "Tcp"
  source_port_range           = "*"
  destination_port_range      = "80"
  source_address_prefix       = "*"
  destination_address_prefix  = "*"
  resource_group_name         = azurerm_resource_group.rg.name
  network_security_group_name = azurerm_network_security_group.nsg.name
}
```

Creating a Virtual Machine Scale Set

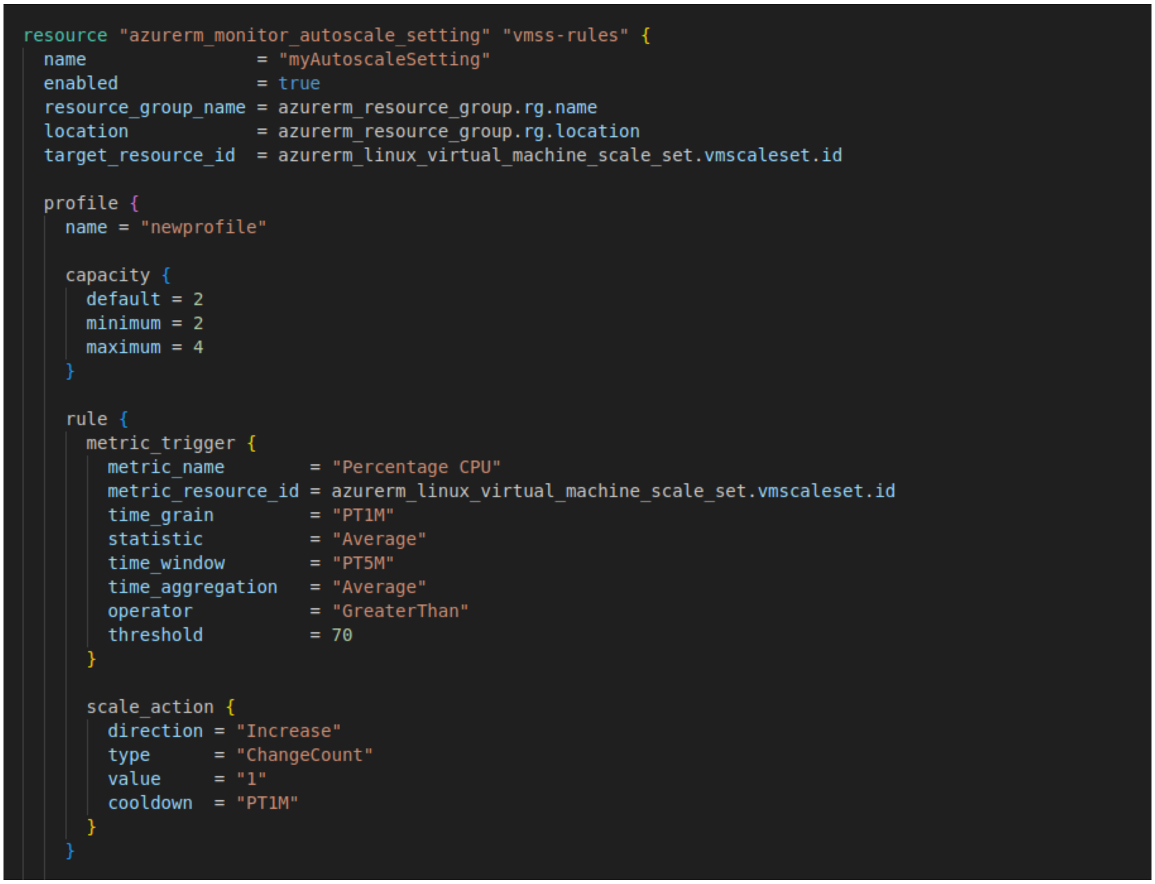
To enable automatic scaling of our application, we use a virtual machine scale set (azurerm_linux_virtual_machine_scale_set). This resource allows us to create and manage a group of identical virtual machines that can scale up or down based on predefined rules. We specify the instance size, operating system image, and number of instances. We also configure the SSH key for remote access.

Virtual Machine Scale Set (VMSS): The azurerm_linux_virtual_machine_scale_set resource creates a virtual machine scale set with the specified name, resource group, location, SKU, number of instances, and other configurations such as SSH key, OS image, disk, and network interface. It also associates the VMSS with the subnet and load balancer backend address pool.

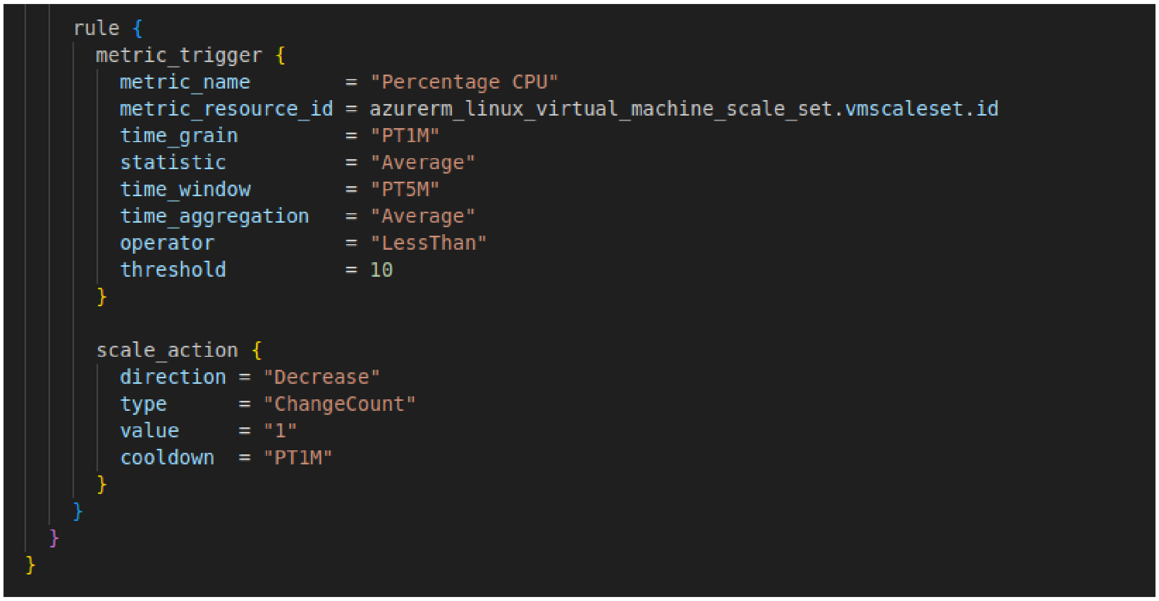


```
#vm configs
resource "azurerm_linux_virtual_machine_scale_set" "vmscaleset" {
  name                = "vmss"
  resource_group_name = azurerm_resource_group.rg.name
  location            = azurerm_resource_group.rg.location
  sku                 = "Standard_B1s"
  instances           = 2
  admin_username      = "adminuser"

  admin_ssh_key {
    username   = "adminuser"
    public_key = file("/home/knoldus/.ssh/id_rsa.pub")
  }

  source_image_reference {
    publisher = "Canonical"
    offer     = "UbuntuServer"
    sku       = "18.04-LTS"
    version   = "latest"
  }

  os_disk {
    storage_account_type = "Standard_LRS"
    caching              = "ReadWrite"
  }

  network_interface {
    name    = "nic"
    primary = true

    ip_configuration {
      name      = "internal"
      primary = true
      subnet_id = azurerm_subnet.subnet.id
      load_balancer_backend_address_pool_ids = [azurerm_lb_backend_address_pool.backend_pool.id]
    }
  }
}
```

Autoscale Setting: The azurerm_monitor_autoscale_setting resource enables autoscaling for the virtual machine scale set based on CPU utilization thresholds. It defines scaling rules that increase or decrease the number of instances based on the average CPU percentage.



```
resource "azurerm_monitor_autoscale_setting" "vmss-rules" {
  name                = "myAutoscaleSetting"
  enabled             = true
  resource_group_name = azurerm_resource_group.rg.name
  location            = azurerm_resource_group.rg.location
  target_resource_id  = azurerm_linux_virtual_machine_scale_set.vmscaleset.id

  profile {
    name = "newprofile"

    capacity {
      default = 2
      minimum = 2
      maximum = 4
    }

    rule {
      metric_trigger {
        metric_name        = "Percentage CPU"
        metric_resource_id = azurerm_linux_virtual_machine_scale_set.vmscaleset.id
        time_grain         = "PT1M"
        statistic          = "Average"
        time_window        = "PT5M"
        time_aggregation   = "Average"
        operator           = "GreaterThan"
        threshold          = 70
      }

      scale_action {
        direction = "Increase"
        type      = "ChangeCount"
        value     = "1"
        cooldown  = "PT1M"
      }
    }
```



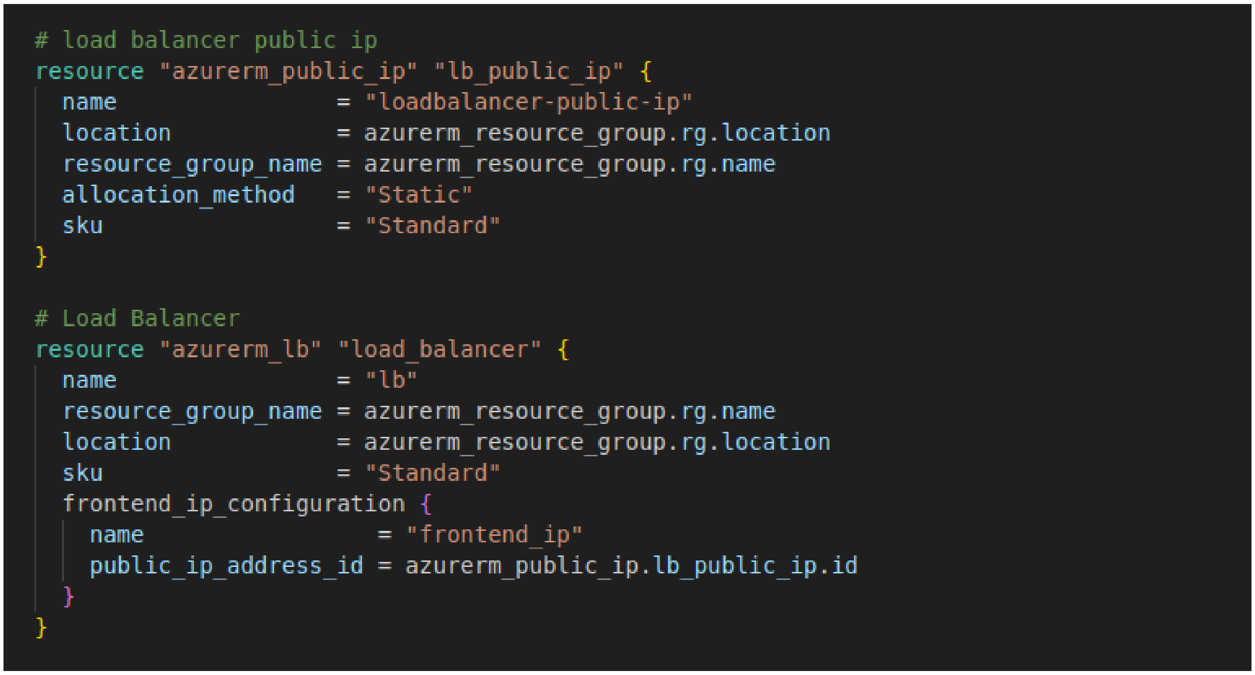
```
    rule {
      metric_trigger {
        metric_name        = "Percentage CPU"
        metric_resource_id = azurerm_linux_virtual_machine_scale_set.vmscaleset.id
        time_grain         = "PT1M"
        statistic          = "Average"
        time_window        = "PT5M"
        time_aggregation   = "Average"
        operator           = "LessThan"
        threshold          = 10
      }

      scale_action {
        direction = "Decrease"
        type      = "ChangeCount"
        value     = "1"
        cooldown  = "PT1M"
      }
    }
  }
}
```

Implementing load balancing

Load Balancer: To distribute incoming traffic across multiple instances of our application, we set up a load balancer (azurerm_lb) with a public IP address (azurerm_public_ip). The load balancer distributes traffic based on defined rules, such as TCP port forwarding. We configure a backend address pool and a probe to monitor the health of the instances. Finally, we create a rule that maps incoming requests to the backend pool.

Public IP: The azurerm_public_ip resource creates a public IP address with the specified name, location, allocation method (static), and SKU (Standard). This public IP address is associated with the load balancer.

Load Balancer: The azurerm_lb resource creates a load balancer with the specified name, resource group, location, and SKU. It also defines a frontend IP configuration that associates the load balancer with the previously created public IP.

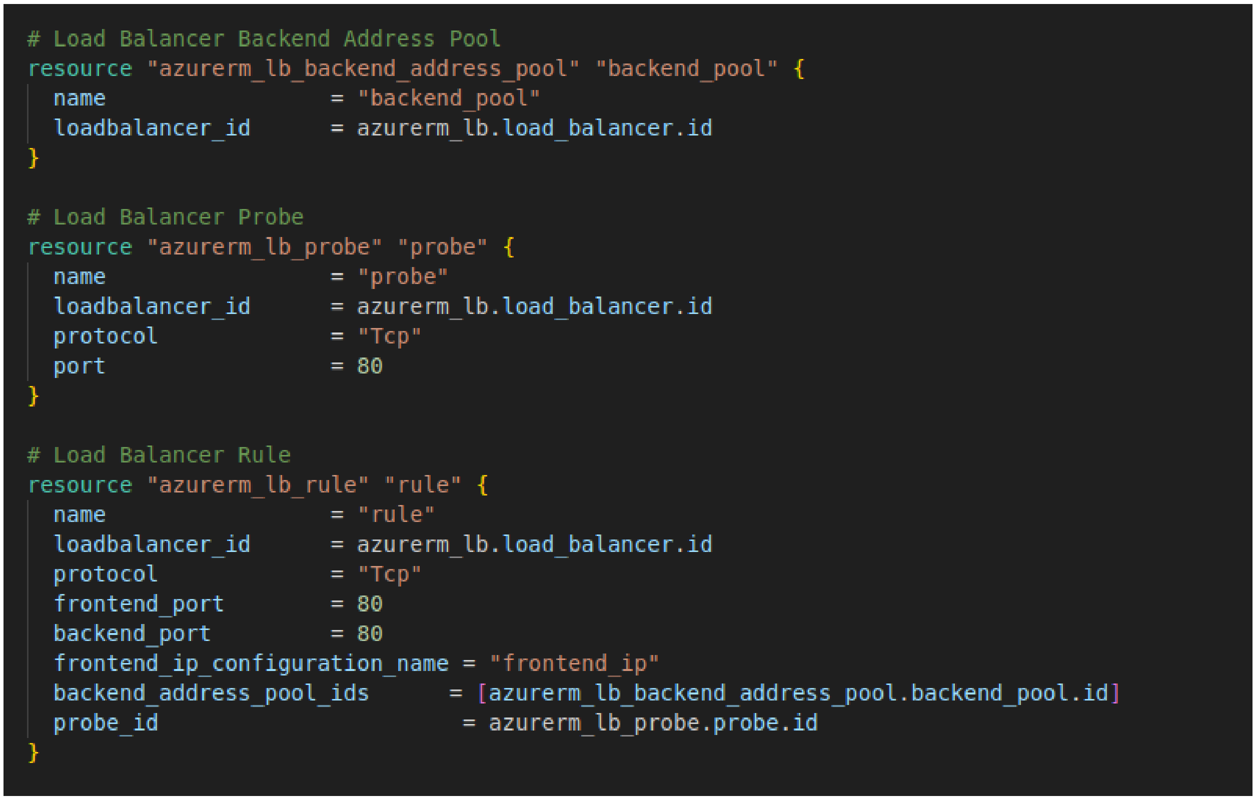


```
# load balancer public ip
resource "azurerm_public_ip" "lb_public_ip" {
  name                = "loadbalancer-public-ip"
  location            = azurerm_resource_group.rg.location
  resource_group_name = azurerm_resource_group.rg.name
  allocation_method   = "Static"
  sku                 = "Standard"
}

# Load Balancer
resource "azurerm_lb" "load_balancer" {
  name                = "lb"
  resource_group_name = azurerm_resource_group.rg.name
  location            = azurerm_resource_group.rg.location
  sku                 = "Standard"
  frontend_ip_configuration {
    name                 = "frontend_ip"
    public_ip_address_id = azurerm_public_ip.lb_public_ip.id
  }
}
```

load Balancer Backend Address Pool: The azurerm_lb_backend_address_pool resource defines a backend address pool for the load balancer, which determines the set of resources that receive traffic from the load balancer.

Load Balancer Probe: The azurerm_lb_probe resource defines a health probe for the load balancer, which checks the availability of backend resources by sending TCP requests to port 80.

Load Balancer Rule: The azurerm_lb_rule resource defines a load balancer rule that maps incoming traffic on port 80 to the backend resources in the backend address pool. It uses the previously defined frontend IP configuration and health probe.



```
# Load Balancer Backend Address Pool
resource "azurerm_lb_backend_address_pool" "backend_pool" {
  name                 = "backend_pool"
  loadbalancer_id      = azurerm_lb.load_balancer.id
}

# Load Balancer Probe
resource "azurerm_lb_probe" "probe" {
  name                 = "probe"
  loadbalancer_id      = azurerm_lb.load_balancer.id
  protocol             = "Tcp"
  port                 = 80
}

# Load Balancer Rule
resource "azurerm_lb_rule" "rule" {
  name                 = "rule"
  loadbalancer_id      = azurerm_lb.load_balancer.id
  protocol             = "Tcp"
  frontend_port        = 80
  backend_port         = 80
  frontend_ip_configuration_name = "frontend_ip"
  backend_address_pool_ids      = [azurerm_lb_backend_address_pool.backend_pool.id]
  probe_id                       = azurerm_lb_probe.probe.id
}
```

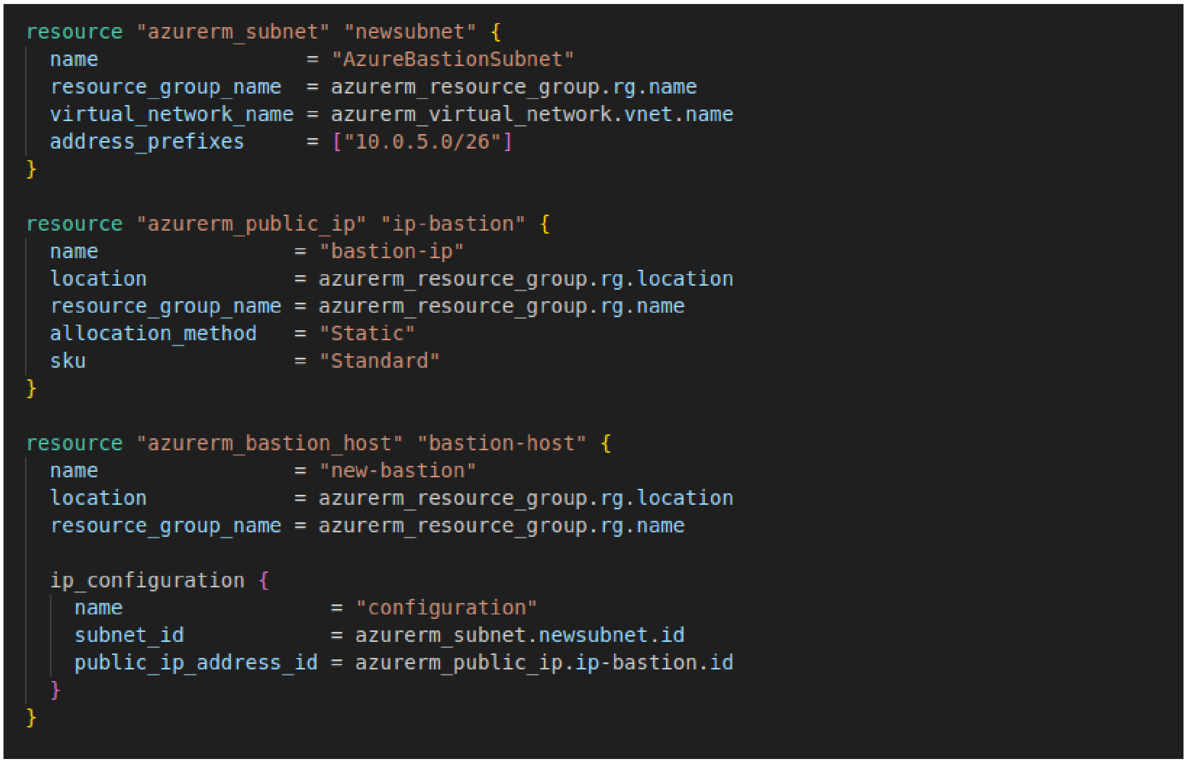
Creating Bastion service to access a specific VMS securely

For secure remote access to our infrastructure, we set up Azure Bastion (azurerm_bastion_host). Azure Bastion provides a fully managed, browser-based SSH and RDP gateway to connect to virtual machines in the virtual network subnet securely.

Bastion Subnet: The azurerm_subnet resource creates an additional subnet within the virtual network specifically for Azure Bastion. It has a specified name, address prefix, and associated resource group and virtual network.

Bastion Public IP: The azurerm_public_ip resource creates a public IP address specifically for Azure Bastion with the specified name, location, allocation method (static), and SKU.

Azure Bastion Host: The azurerm_bastion_host resource provisions an Azure Bastion host with the specified name, location, and associated resource group. It is configured with an IP configuration that links it to the Bastion



```
resource "azurerm_subnet" "newsubnet" {
  name                 = "AzureBastionSubnet"
  resource_group_name  = azurerm_resource_group.rg.name
  virtual_network_name = azurerm_virtual_network.vnet.name
  address_prefixes     = ["10.0.5.0/26"]
}

resource "azurerm_public_ip" "ip-bastion" {
  name                = "bastion-ip"
  location            = azurerm_resource_group.rg.location
  resource_group_name = azurerm_resource_group.rg.name
  allocation_method   = "Static"
  sku                 = "Standard"
}

resource "azurerm_bastion_host" "bastion-host" {
  name                = "new-bastion"
  location            = azurerm_resource_group.rg.location
  resource_group_name = azurerm_resource_group.rg.name

  ip_configuration {
    name                 = "configuration"
    subnet_id            = azurerm_subnet.newsubnet.id
    public_ip_address_id = azurerm_public_ip.ip-bastion.id
  }
}
```

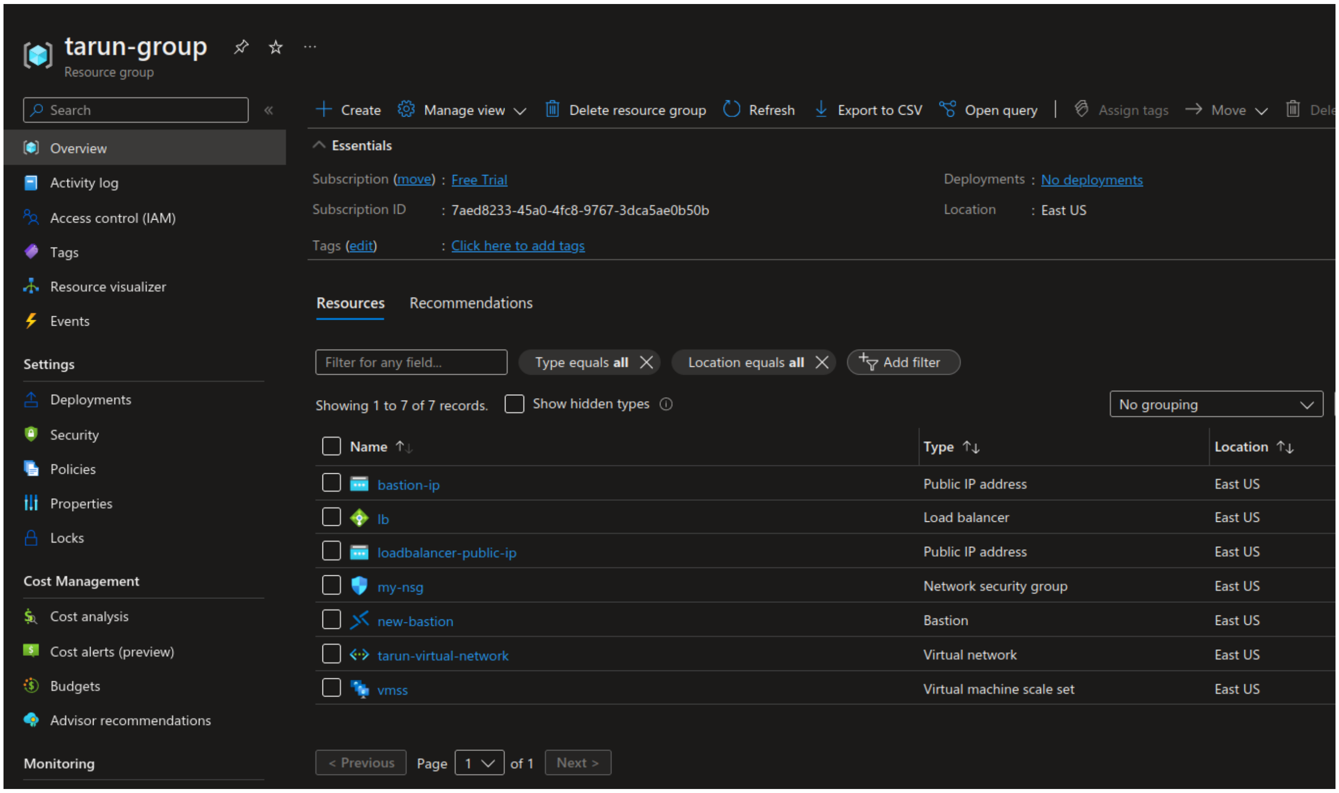
Implement the code saving the code in a main.tf file and execute with terraform init, plan and apply command.

Terraform init: The terraform init command is used to initialize a Terraform working directory. It downloads the necessary provider plugins and sets up the backend configuration. During initialization, Terraform checks for any configuration files in the working directory and automatically downloads the required provider plugins specified in the configuration. This command needs to be executed only once in a new or existing Terraform project.

Terraform plan: The terraform plan command is used to create an execution plan for Terraform. It examines the current configuration and compares it with the deployed infrastructure to determine the changes that need to be made. It generates a detailed report that includes resource creation, modification, or deletion. This command allows you to review the proposed changes before actually applying them, providing an opportunity to catch any errors or unintended modifications.

Terraform apply: The terraform apply command is used to apply the changes defined in the Terraform configuration. It creates, modifies, or deletes resources based on the execution plan generated by terraform plan. When running terraform apply, Terraform prompts for confirmation before making any modifications to the infrastructure. It also displays a summary of the changes that will be applied. Once confirmed, Terraform starts provisioning or modifying the resources as specified in the configuration.

Resource Group

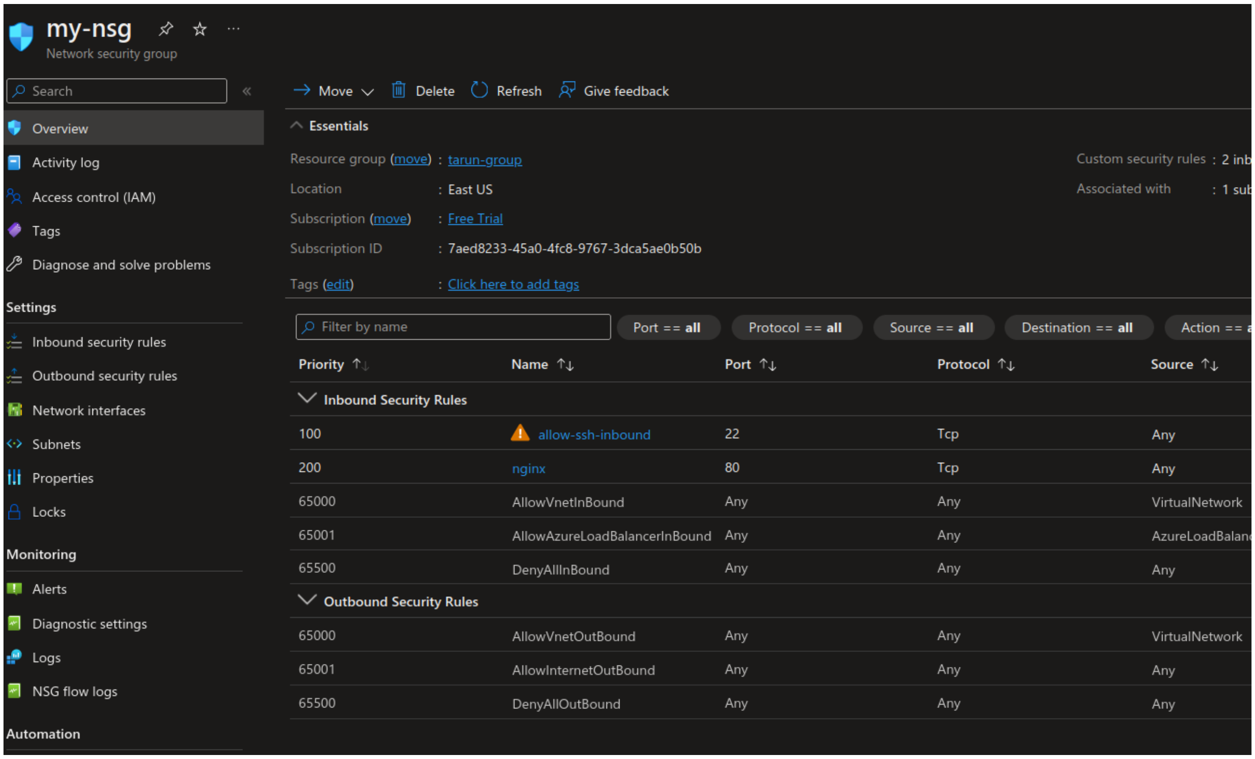


Subscription (move) : Free Trial
Subscription ID : 7aed8233-45a0-4fc8-9767-3dca5ae0b50b
Tags (edit) : Click here to add tags
Deployments : No deployments
Location : East US

Showing 1 to 7 of 7 records. □ Show hidden types

□ Name	Type	Location
□ bastion-ip	Public IP address	East US
□ lb	Load balancer	East US
□ loadbalancer-public-ip	Public IP address	East US
□ my-nsg	Network security group	East US
□ new-bastion	Bastion	East US
□ tarun-virtual-network	Virtual network	East US
□ vmss	Virtual machine scale set	East US

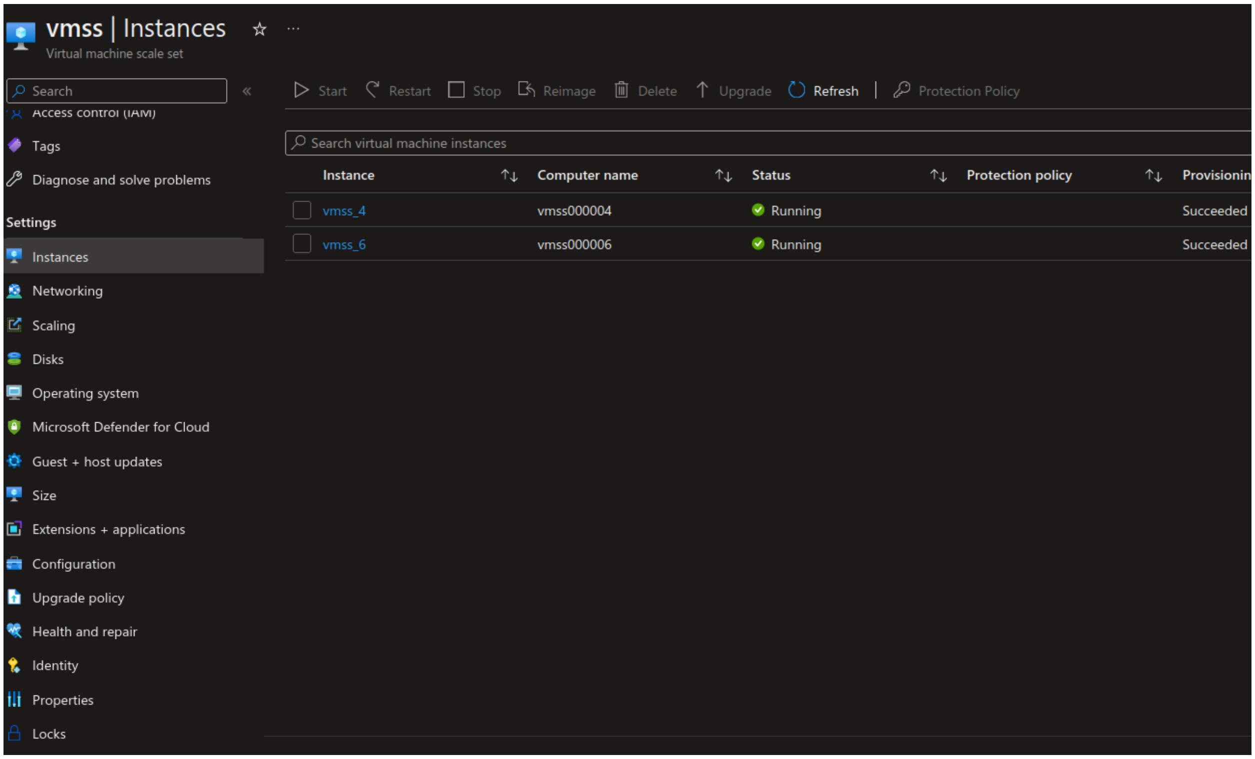
Network Security Group



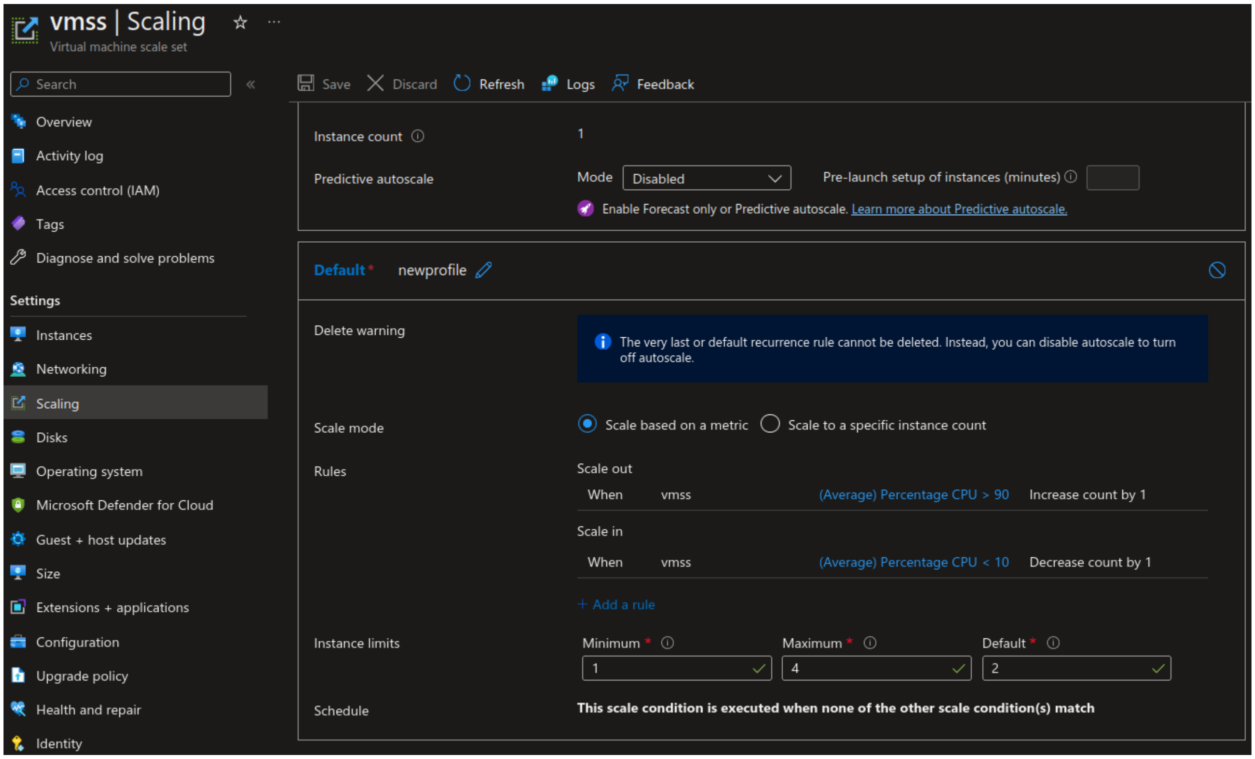
Resource group (move) : tarun-group
Location : East US
Subscription (move) : Free Trial
Subscription ID : 7aed8233-45a0-4fc8-9767-3dca5ae0b50b
Tags (edit) : Click here to add tags
Custom security rules : 2 inb
Associated with : 1 sub

Priority	Name	Port	Protocol	Source
Inbound Security Rules				
100	⚠ allow-ssh-inbound	22	Tcp	Any
200	nginx	80	Tcp	Any
65000	AllowVnetInBound	Any	Any	VirtualNetwork
65001	AllowAzureLoadBalancerInBound	Any	Any	AzureLoadBalan
65500	DenyAllInBound	Any	Any	Any
Outbound Security Rules				
65000	AllowVnetOutBound	Any	Any	VirtualNetwork
65001	AllowInternetOutBound	Any	Any	Any
65500	DenyAllOutBound	Any	Any	Any

Scale Set resources and Scaling Rules



	Instance	Computer name	Status	Protection policy	Provisionin
□	vmss_4	vmss000004	✓ Running		Succeeded
□	vmss_6	vmss000006	✓ Running		Succeeded



Instance count ⓘ 1

Predictive autoscale Mode Disabled Pre-launch setup of instances (minutes) ⓘ

Enable Forecast only or Predictive autoscale. Learn more about Predictive autoscale.

Default * newprofile

Delete warning: The very last or default recurrence rule cannot be deleted. Instead, you can disable autoscale to turn off autoscale.

Scale mode ● Scale based on a metric ○ Scale to a specific instance count

Rules

Scale out

When	vmss	(Average) Percentage CPU > 90	Increase count by 1

Scale in

When	vmss	(Average) Percentage CPU < 10	Decrease count by 1

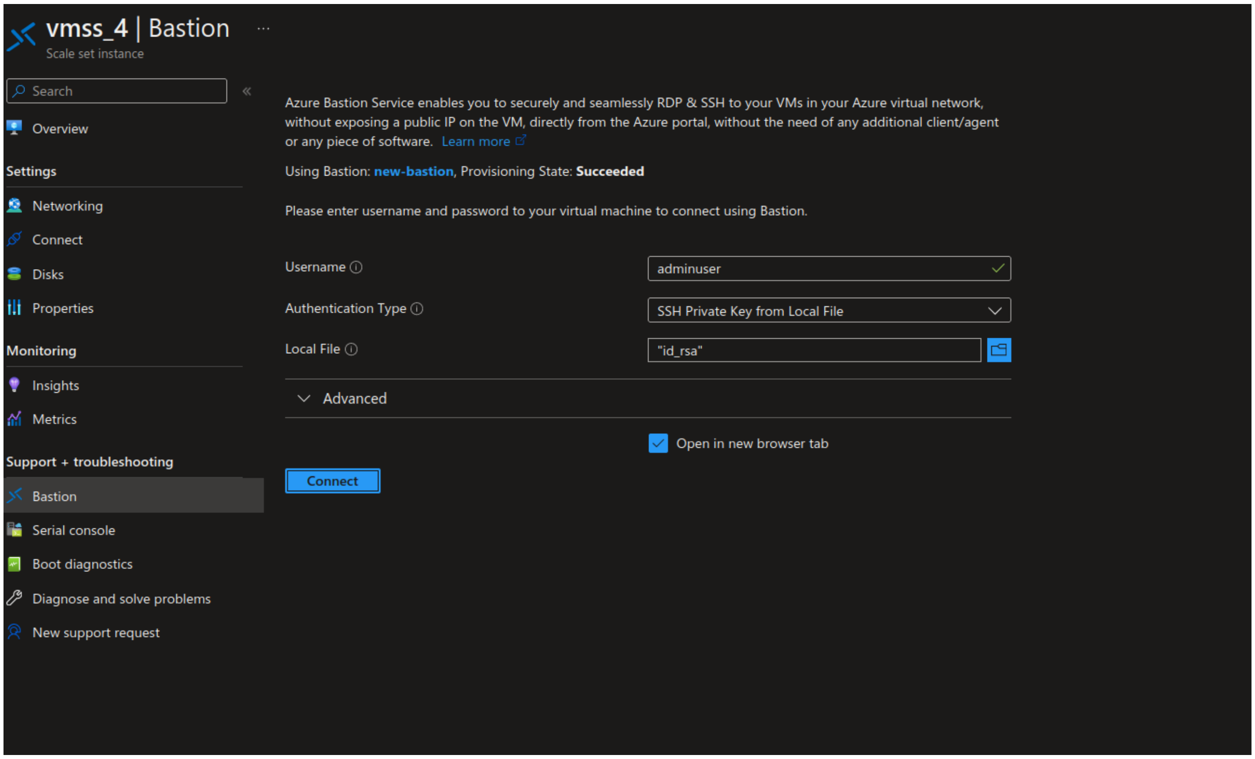
+ Add a rule

Instance limits

Minimum * ⓘ	Maximum * ⓘ	Default * ⓘ
1	4	2

Schedule **This scale condition is executed when none of the other scale condition(s) match**

Bastion



Azure Bastion Service enables you to securely and seamlessly RDP & SSH to your VMs in your Azure virtual network, without exposing a public IP on the VM, directly from the Azure portal, without the need of any additional client/agent or any piece of software. Learn more

Using Bastion: **new-bastion**, Provisioning State: **Succeeded**

Please enter username and password to your virtual machine to connect using Bastion.

Username ⓘ adminuser

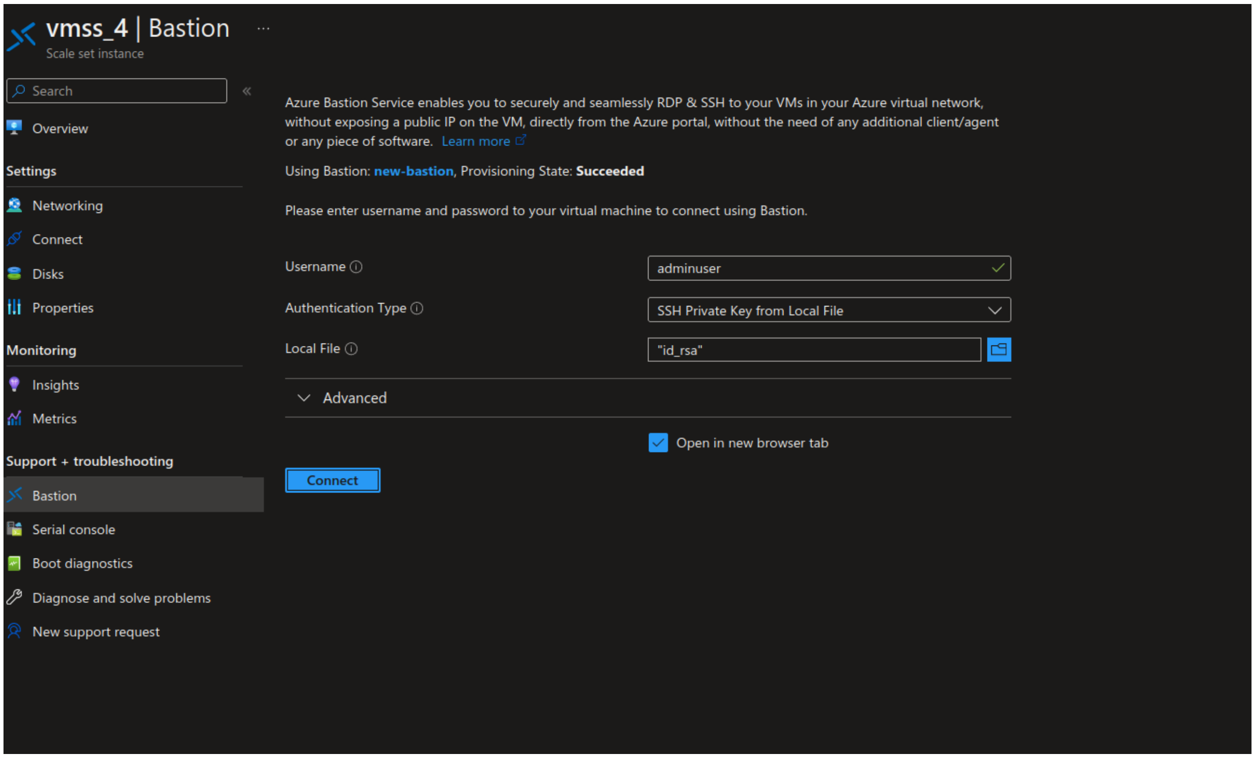
Authentication Type ⓘ SSH Private Key from Local File

Local File ⓘ "id_rsa"

Advanced

☑ Open in new browser tab

Connect



Azure Bastion Service enables you to securely and seamlessly RDP & SSH to your VMs in your Azure virtual network, without exposing a public IP on the VM, directly from the Azure portal, without the need of any additional client/agent or any piece of software. Learn more

Using Bastion: **new-bastion**, Provisioning State: **Succeeded**

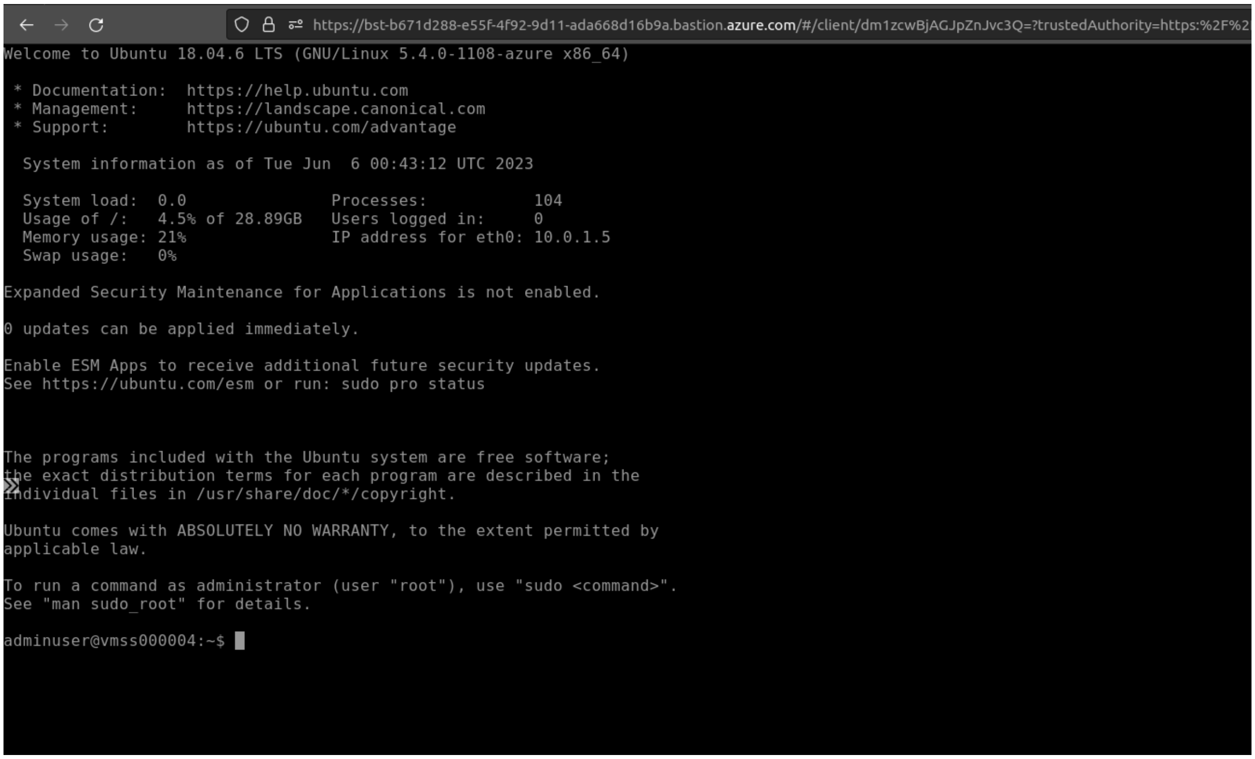
Please enter username and password to your virtual machine to connect using Bastion.

Username: adminuser

Authentication Type: SSH Private Key from Local File

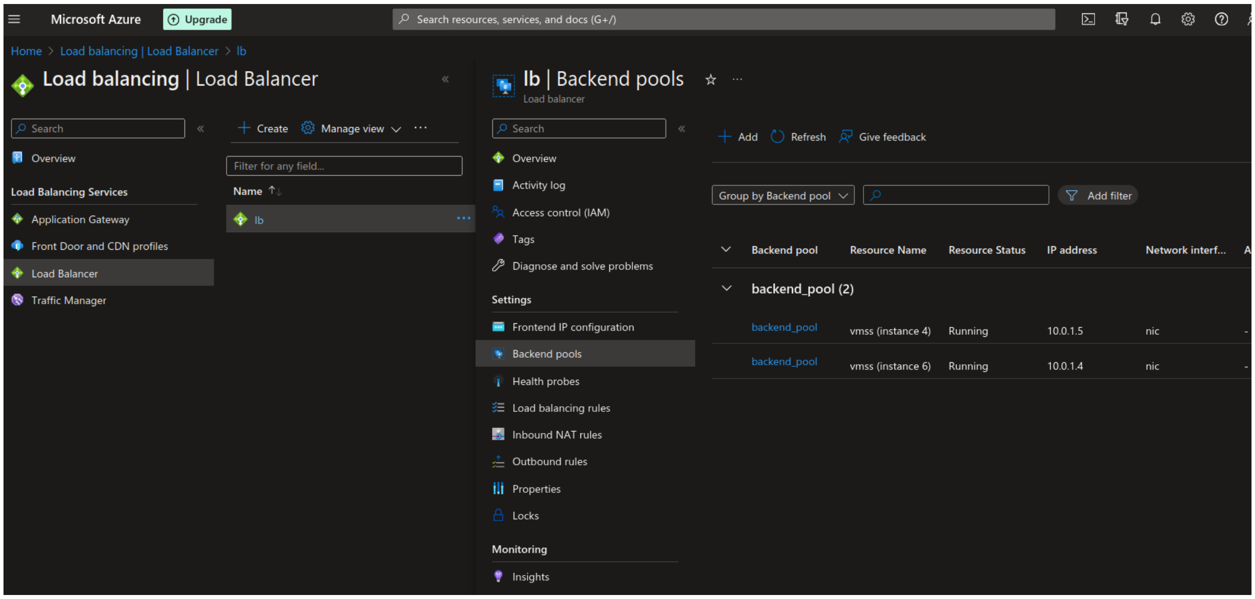
Local File: "id_rsa"

☑ Open in new browser tab

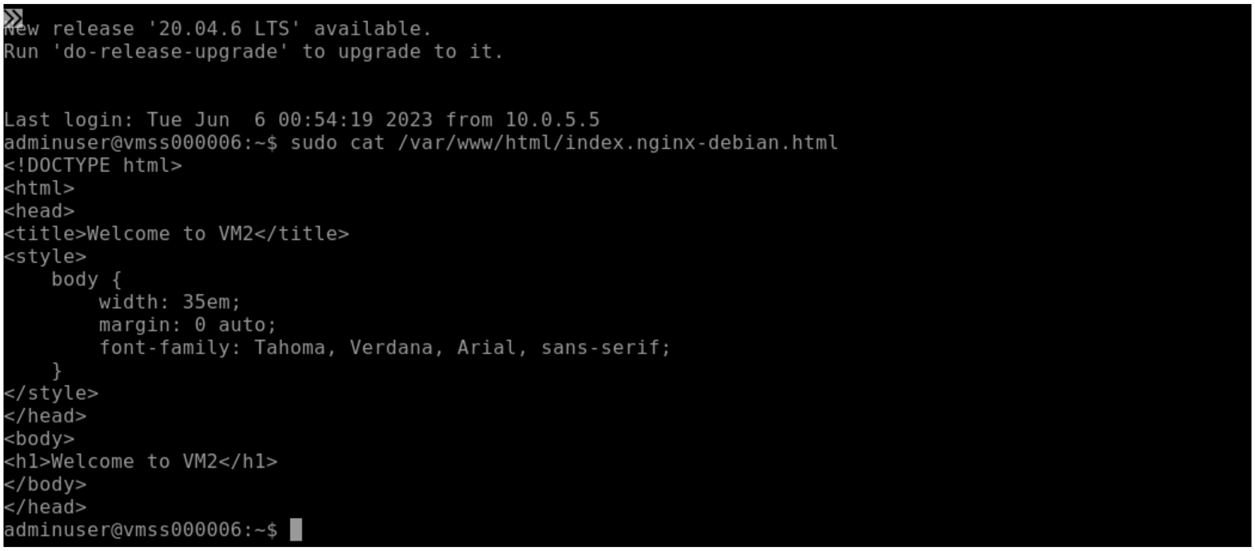

https://bst-b671d288-e55f-4f92-9d11-ada668d16b9a.bastion.azure.com/#/client/dm1zcwBjAGJpZnJvc3Q=?trustedAuthority=https:%2F%2

```
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1108-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Tue Jun  6 00:43:12 UTC 2023

  System load:  0.0               Processes:           104
  Usage of /:   4.5% of 28.89GB   Users logged in:     0
  Memory usage: 21%               IP address for eth0: 10.0.1.5
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

adminuser@vmss000004:~$ 
```

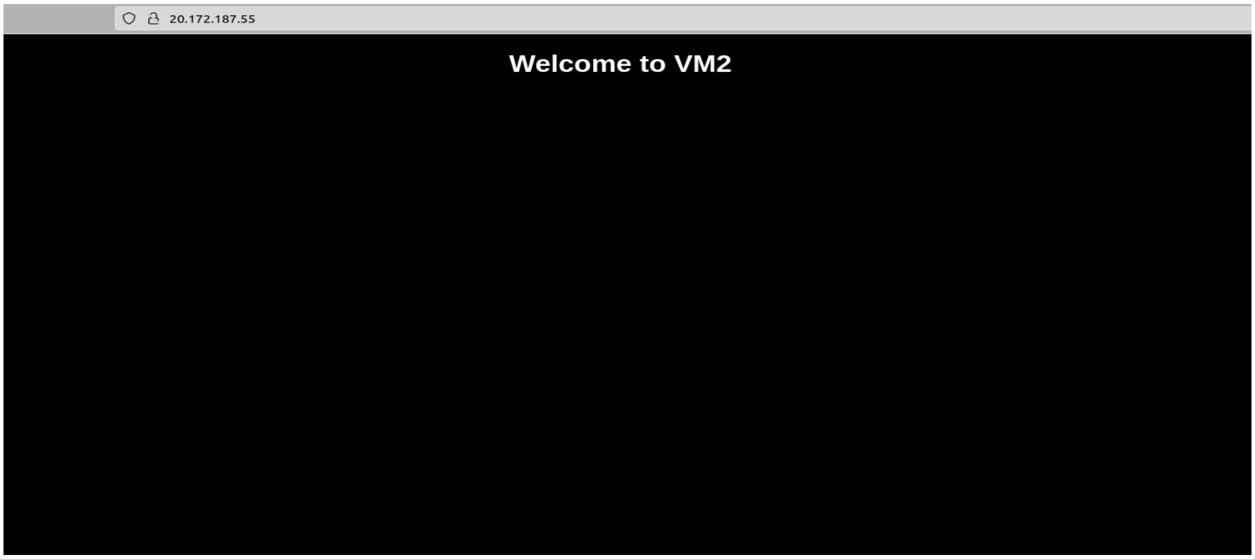
Installing nginx on VMs in the pool with Load balancer



Backend pool	Resource Name	Resource Status	IP address	Network interf...	A
backend_pool (2)					
backend_pool	vmss (instance 4)	Running	10.0.1.5	nic	-
backend_pool	vmss (instance 6)	Running	10.0.1.4	nic	-

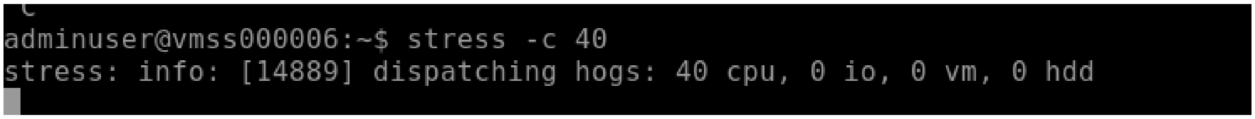


```
New release '20.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Tue Jun  6 00:54:19 2023 from 10.0.5.5
adminuser@vmss000006:~$ sudo cat /var/www/html/index.nginx-debian.html
<!DOCTYPE html>
<html>
<head>
<title>Welcome to VM2</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to VM2</h1>
</body>
</head>
adminuser@vmss000006:~$
```

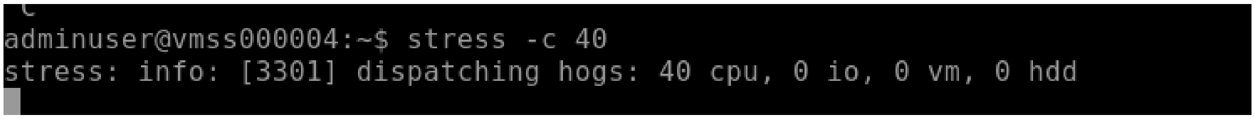


Welcome to VM2

Building stress to auto-scale
Using command stress -c gor cpu utilization

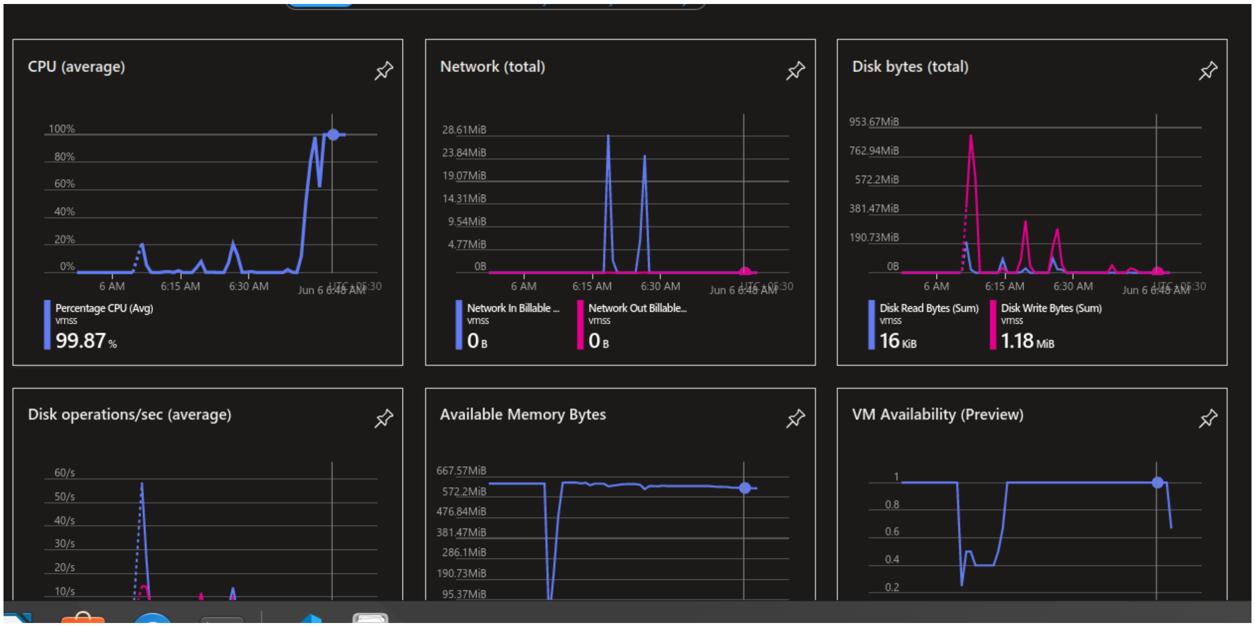


```
adminuser@vmss000006:~$ stress -c 40
stress: info: [14889] dispatching hogs: 40 cpu, 0 io, 0 vm, 0 hdd
```

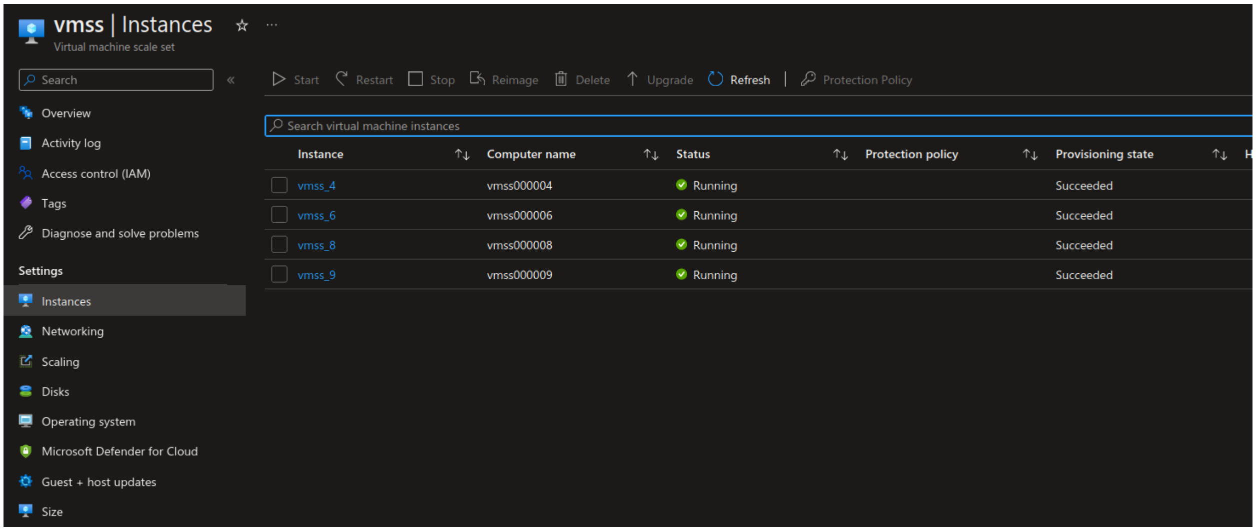


```
adminuser@vmss000004:~$ stress -c 40
stress: info: [3301] dispatching hogs: 40 cpu, 0 io, 0 vm, 0 hdd
```

Monitoring the VMs to check if stress worked



VMs added in the set the moment cpu utilization croos 90%



Instance	Computer name	Status	Protection policy	Provisioning state
☐ vmss_4	vmss000004	Running		Succeeded
☐ vmss_6	vmss000006	Running		Succeeded
☐ vmss_8	vmss000008	Running		Succeeded
☐ vmss_9	vmss000009	Running		Succeeded