Lab 06 - Implement Traffic Management

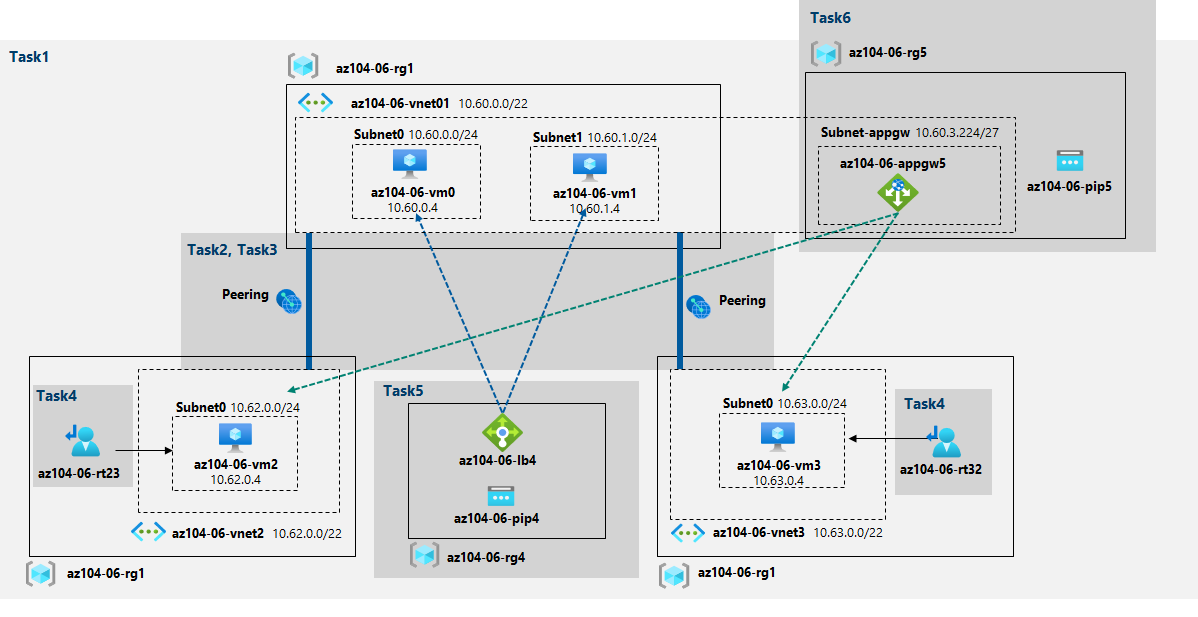
## **Lab scenario**

You were tasked with testing managing network traffic targeting Azure virtual machines in the hub and spoke network topology, which Contoso considers implementing in its Azure environment (instead of creating the mesh topology, which you tested in the previous lab). This testing needs to include implementing connectivity between spokes by relying on user defined routes that force traffic to flow via the hub, as well as traffic distribution across virtual machines by using layer 4 and layer 7 load balancers. For this purpose, you intend to use Azure Load Balancer (layer 4) and Azure Application Gateway (layer 7).

## **Objectives**

* Task 1: Provision the lab environment
* Task 2: Configure the hub and spoke network topology
* Task 3: Test transitivity of virtual network peering
* Task 4: Configure routing in the hub and spoke topology
* Task 5: Implement Azure Load Balancer
* Task 6: Implement Azure Application Gateway

## **Architecture diagram**



Exercise 1

Task 1: Provision the lab environment

In this task, we will deploy four virtual machines into the same Azure region. The first two will reside in a hub virtual network, while each of the remaining two will reside in a separate spoke virtual network.

For this task we need to get the files “az104-06-vms-loop-template.json” and “az104-06-vms-loop-parameters.json” from Microsoft’s Learning page on [GitHub](https://github.com/MicrosoftLearning) and change the password from the **Parameters** file.

We open **Azure Cloud Shell** and select **PowerShell**. Here we upload the previously downloaded files and run the following command to create the resource group that will be hosting the lab environment.

From the Cloud Shell pane we run the following to create the first resource group that will be hosting the lab environment:

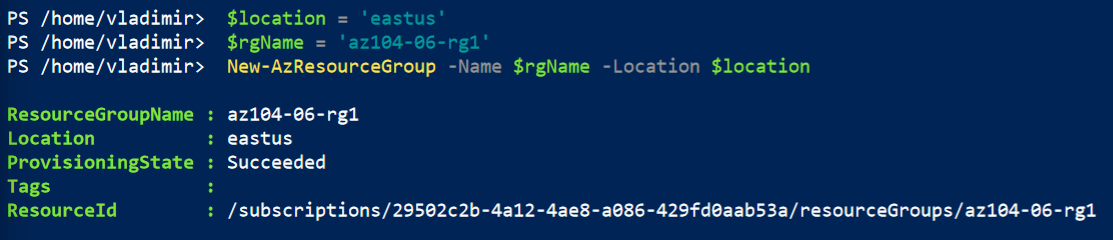
$location = 'eastus'

Then the resource group name:

$rgName = 'az104-06-rg1'

And finally create the resource group in our desired location:

New-AzResourceGroup -Name $rgName -Location $location



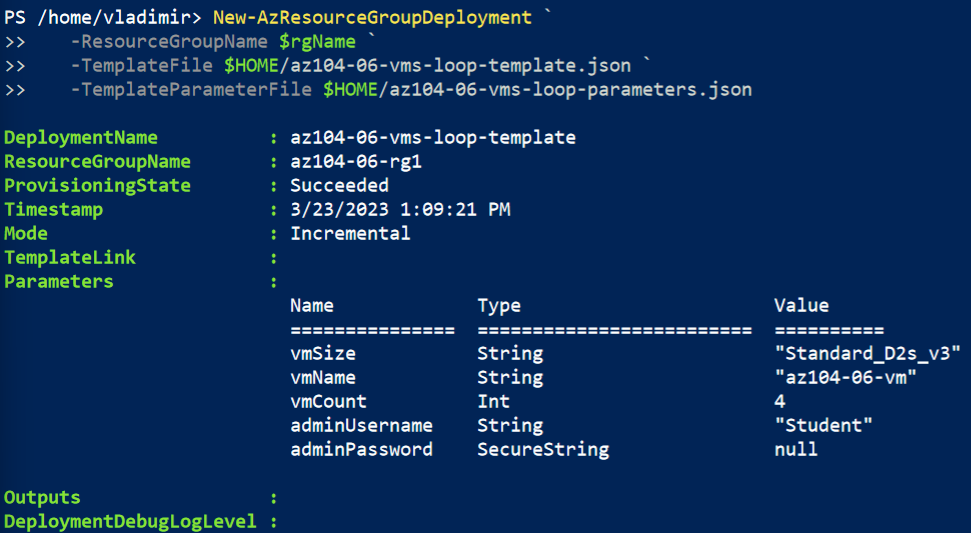
From the Cloud Shell pane, we run the following to create the three virtual networks and four Azure VMs into them by using the template and parameter files we uploaded:

New-AzResourceGroupDeployment `

-ResourceGroupName $rgName `

-TemplateFile $HOME/az104-06-vms-loop-template.json `

-TemplateParameterFile $HOME/az104-06-vms-loop-parameters.json



From the Cloud Shell pane, we run the following to install the Network Watcher extension on the Azure VMs deployed in the previous step:

$rgName = 'az104-06-rg1'

$location = (Get-AzResourceGroup -ResourceGroupName $rgName).location

$vmNames = (Get-AzVM -ResourceGroupName $rgName).Name

foreach ($vmName in $vmNames) {

Set-AzVMExtension `

-ResourceGroupName $rgName `

-Location $location `

-VMName $vmName `

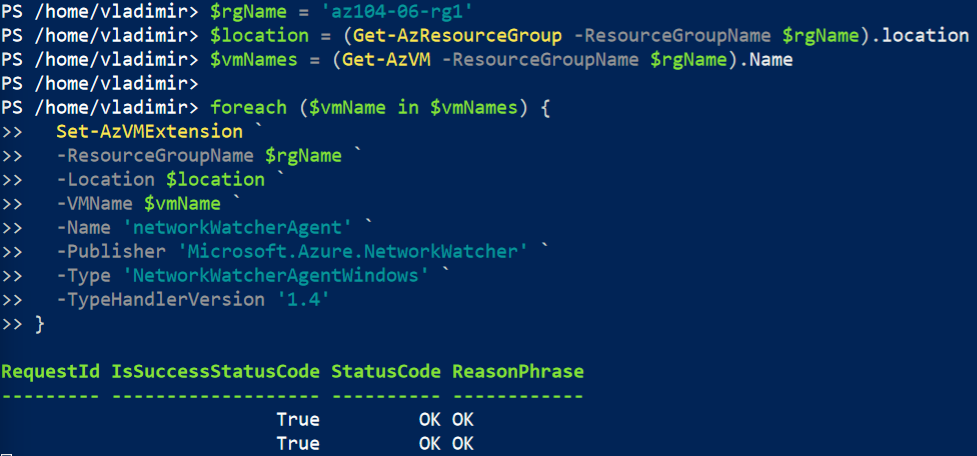
-Name 'networkWatcherAgent' `

-Publisher 'Microsoft.Azure.NetworkWatcher' `

-Type 'NetworkWatcherAgentWindows' `

-TypeHandlerVersion '1.4'

}



Task 2: Configure the hub and spoke network topology

In this task, we will configure local peering between the virtual networks we deployed in the previous tasks in order to create a hub and spoke network topology.

First we go to our Virtual networks and check and copy the Resource ID property from the Properties blade on the networks **az104-06-vnet2 and az104-06-vnet3.**

**/subscriptions/29502c2b-4a12-4ae8-a086-429fd0aab53a/resourceGroups/az104-06-rg1/providers/Microsoft.Network/virtualNetworks/az104-06-vnet2**

**/subscriptions/29502c2b-4a12-4ae8-a086-429fd0aab53a/resourceGroups/az104-06-rg1/providers/Microsoft.Network/virtualNetworks/az104-06-vnet3**

**Then we go to the az104-06-vnet01** virtual network blade, and on the Settings section we click Peerings and then Add one with the following settings:

| Setting | Value |
| --- | --- |
| This virtual network: Peering link name | **az104-06-vnet01\_to\_az104-06-vnet2** |
| Traffic to remote virtual network | **Allow (default)** |
| Traffic forwarded from remote virtual network | **Block traffic that originates from outside this virtual network** |
| Virtual network gateway | **None (default)** |
| Remote virtual network: Peering link name | **az104-06-vnet2\_to\_az104-06-vnet01** |
| Virtual network deployment model | **Resource manager** |
| I know my resource ID | enabled |
| Resource ID | **/subscriptions/29502c2b-4a12-4ae8-a086-429fd0aab53a/resourceGroups/az104-06-rg1/providers/Microsoft.Network/virtualNetworks/az104-06-vnet2** |
| Traffic to remote virtual network | **Allow (default)** |
| Traffic forwarded from remote virtual network | **Allow (default)** |
| Virtual network gateway | **None (default)** |

Then we add another Peering with the following settings:

| Setting | Value |
| --- | --- |
| This virtual network: Peering link name | **az104-06-vnet01\_to\_az104-06-vnet3** |
| Traffic to remote virtual network | **Allow (default)** |
| Traffic forwarded from remote virtual network | **Block traffic that originates from outside this virtual network** |
| Virtual network gateway | **None (default)** |
| Remote virtual network: Peering link name | **az104-06-vnet3\_to\_az104-06-vnet01** |
| Virtual network deployment model | **Resource manager** |
| I know my resource ID | enabled |
| Resource ID | **/subscriptions/29502c2b-4a12-4ae8-a086-429fd0aab53a/resourceGroups/az104-06-rg1/providers/Microsoft.Network/virtualNetworks/az104-06-vnet3** |
| Traffic to remote virtual network | **Allow (default)** |
| Traffic forwarded from remote virtual network | **Allow (default)** |
| Virtual network gateway | **None (default)** |

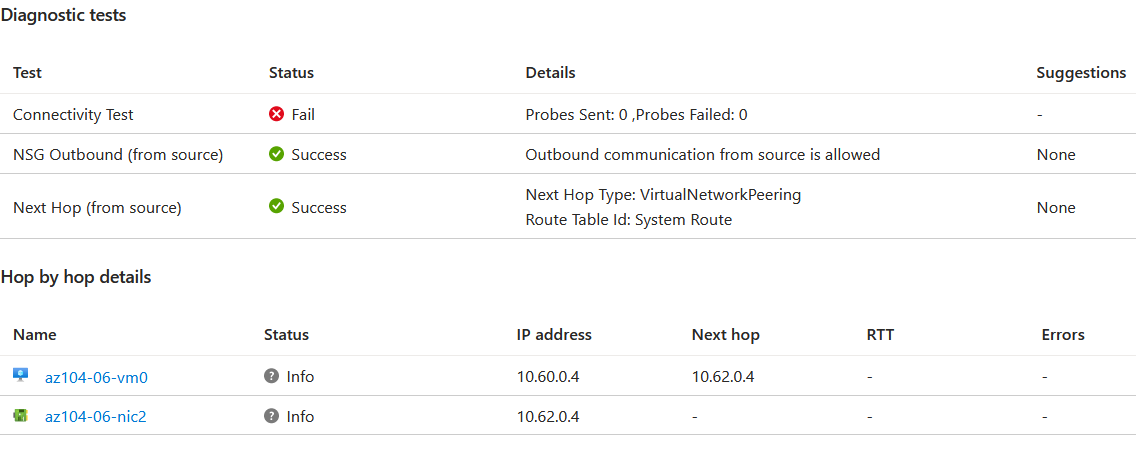
Task 3: Test transitivity of virtual network peering

In this task, we will test transitivity of virtual network peering by using Network Watcher.

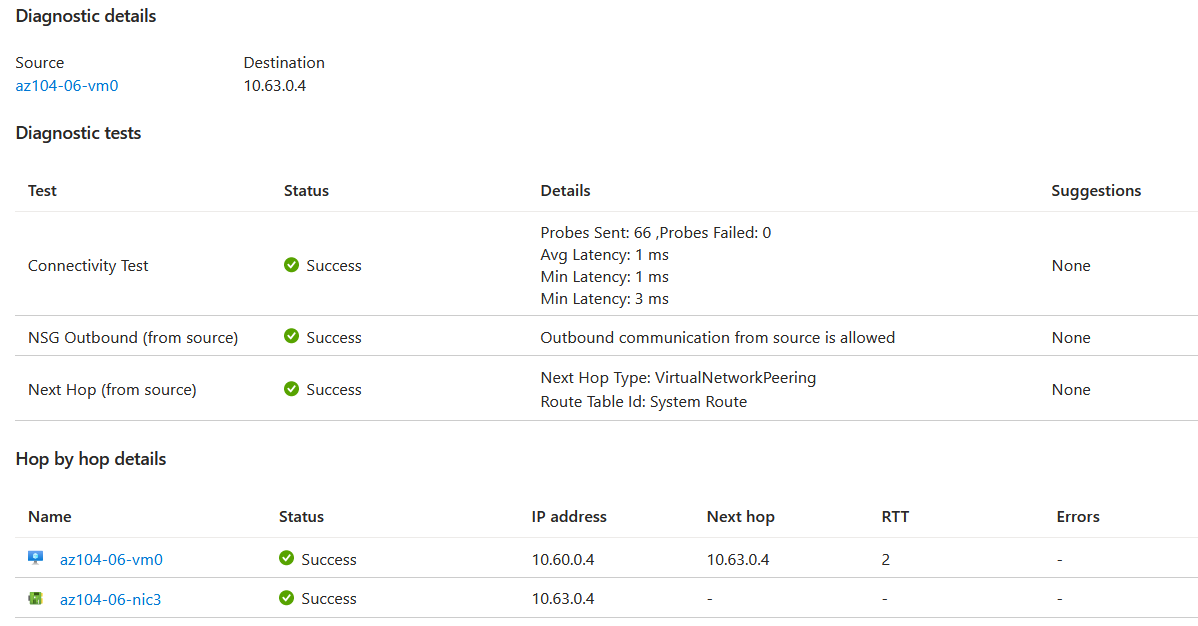
We navigate to Network Watcher and click on Connection troubleshoot. Here we initiate a check with the following settings:

| Setting | Value |
| --- | --- |
| Subscription | the name of the Azure subscription you are using in this lab |
| Resource group | **az104-06-rg1** |
| Source type | **Virtual machine** |
| Virtual machine | **az104-06-vm0** |
| Destination | **Specify manually** |
| URI, FQDN or IPv4 | **10.62.0.4** |
| Protocol | **TCP** |
| Destination Port | **3389** |

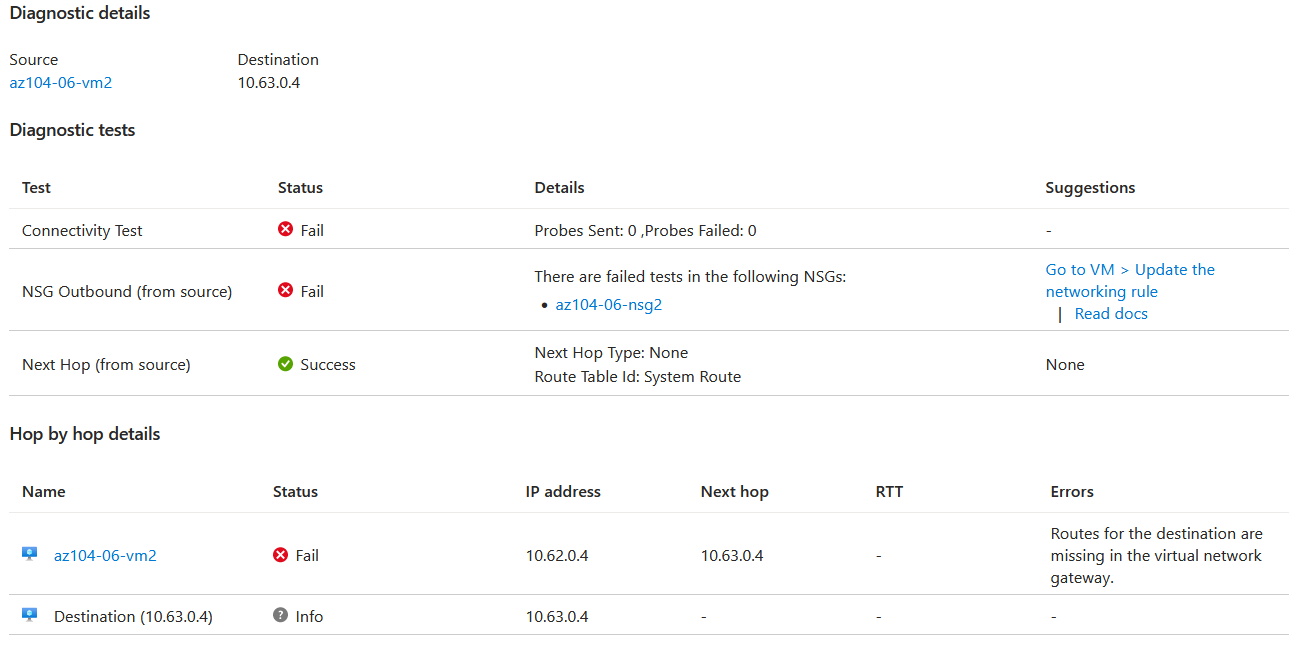
We verify if the status is Reachable.



We do the same just with changing the IP address **10.63.0.4 of az104-06-vm3**



And last, we try again with changing the parameter for the virtual machine to **az104-06-vm2**

****

We can note that the status is Unreachable.

Task 4: Configure routing in the hub and spoke topology

In this task, we will configure and test routing between the two spoke virtual networks by enabling IP forwarding on the network interface of the **az104-06-vm0** virtual machine, enabling routing within its operating system, and configuring user-defined routes on the spoke virtual network.

In the Azure portal we select Virtual machines and from there we click **az104-06-vm0**. From the Settings we click on Networking and then **az104-06-nic0**. On the network interface blade under Settings, we click IP configurations and set **IP forwarding** to **Enabled.**

Back on the **az104-06-vm0, in the Operations section we click Run command and in the list of commands RunPowerShellScript.**

We use “Install-WindowsFeature RemoteAccess -IncludeManagementTools” to install the Remote Access Windows Server role.

Then we use:

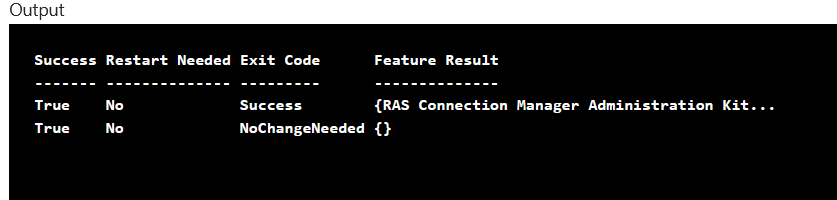
Install-WindowsFeature -Name Routing -IncludeManagementTools -IncludeAllSubFeature

Install-WindowsFeature -Name "RSAT-RemoteAccess-Powershell"

Install-RemoteAccess -VpnType RoutingOnly

Get-NetAdapter | Set-NetIPInterface -Forwarding Enabled

to install the Routing role service.

****

Now we go to **Route tables** and we create route table with the following settings:

| Setting | Value |
| --- | --- |
| Subscription | the name of the Azure subscription you are using in this lab |
| Resource group | **az104-06-rg1** |
| Location | East US |
| Name | **az104-06-rt23** |
| Propagate gateway routes | **No** |

We go to the resource and click **Routes** and **Add** to add new route with the following settings:

| Setting | Value |
| --- | --- |
| Route name | **az104-06-route-vnet2-to-vnet3** |
| Address prefix destination | **IP Addresses** |
| Destination IP addresses/CIDR ranges | **10.63.0.0/20** |
| Next hop type | **Virtual appliance** |
| Next hop address | **10.60.0.4** |

Now we go to **Subnets** and click **Associate.** We associate the route table with the following subnet:

| Setting | Value |
| --- | --- |
| Virtual network | **az104-06-vnet2** |
| Subnet | **subnet0** |

We navigate back to **Route tables** and click **Create** with the following settings:

| Setting | Value |
| --- | --- |
| Subscription | the name of the Azure subscription you are using in this lab |
| Resource group | **az104-06-rg1** |
| Region | the name of the Azure region in which you created the virtual networks |
| Name | **az104-06-rt32** |
| Propagate gateway routes | **No** |

We now go to the resource and click on **Routes** and **Add** one with the following settings:

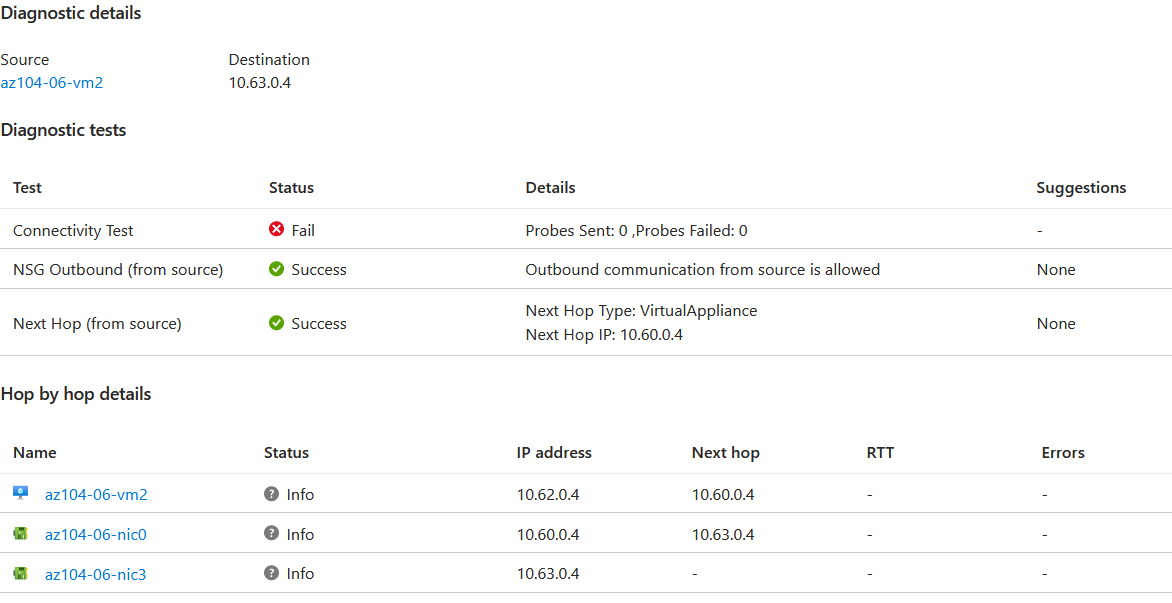
| Setting | Value |
| --- | --- |
| Route name | **az104-06-route-vnet3-to-vnet2** |
| Address prefix destination | **IP Addresses** |
| Destination IP addresses/CIDR ranges | **10.62.0.0/20** |
| Next hop type | **Virtual appliance** |
| Next hop address | **10.60.0.4** |

Then we go to **Subnets** and click **Associate** with the following subnet:

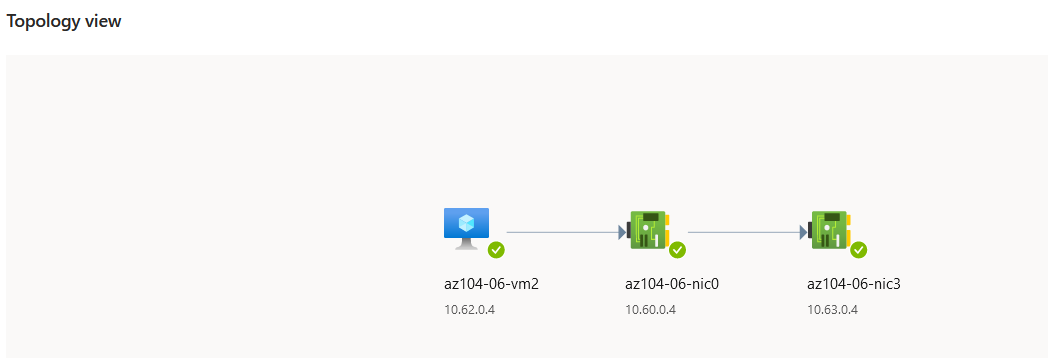
| Setting | Value |
| --- | --- |
| Virtual network | **az104-06-vnet3** |
| Subnet | **subnet0** |

We navigate back to **Network Watcher – Connection troubleshoot** blade and we initiate a check with the following settings:

| Setting | Value |
| --- | --- |
| Subscription | the name of the Azure subscription you are using in this lab |
| Resource group | **az104-06-rg1** |
| Source type | **Virtual machine** |
| Virtual machine | **az104-06-vm2** |
| Destination | **Specify manually** |
| URI, FQDN or IPv4 | **10.63.0.4** |
| Protocol | **TCP** |
| Destination Port | **3389** |



We can view the topology of the network with Network Watcher.



Task 5: Implement Azure Load Balancer

In this task, you will implement an Azure Load Balancer in front of the two Azure virtual machines in the hub virtual network.

We navigate to **Load balancers** and click on **Create** with the following settings:

| Setting | Value |
| --- | --- |
| Subscription | the name of the Azure subscription you are using in this lab |
| Resource group | **az104-06-rg4** (if necessary create) |
| Name | **az104-06-lb4** |
| Region | East US |
| SKU | **Standard** |
| Type | **Public** |
| Tier | **Regional** |

Then we go to the **Frontend IP configuration** tab click on add and use the following settings:

| Setting | Value |
| --- | --- |
| Name | **az104-06-pip4** |
| IP version | IPv4 |
| IP type | IP address |
| Public IP address | **Create new** |
| Availability zone | **No Zone** |

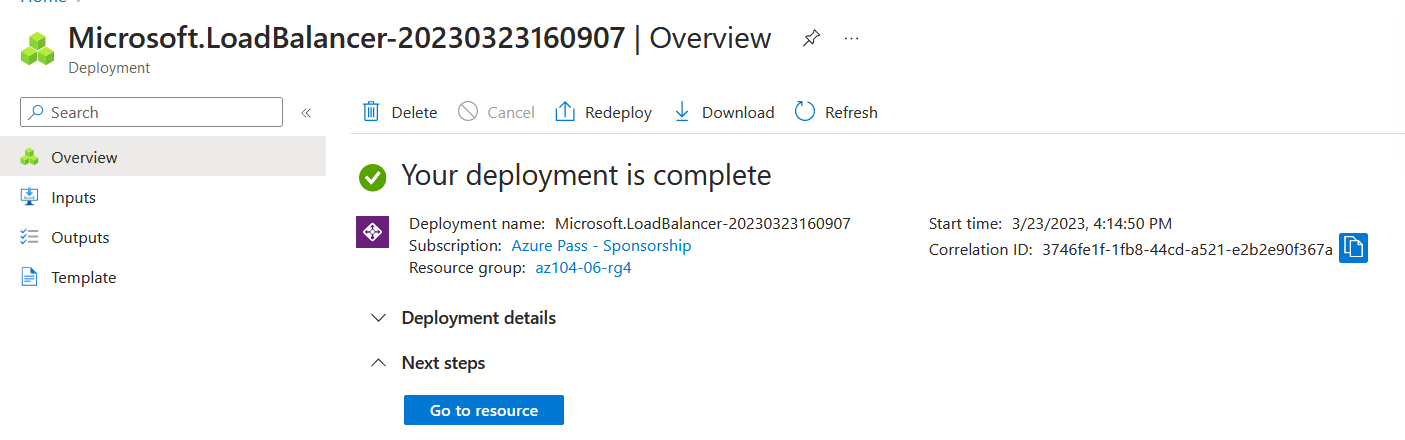
On the **Backend pools** tab, we click **Add a backend pool** with the following settings:

| Setting | Value |
| --- | --- |
| Name | **az104-06-lb4-be1** |
| Virtual network | **az104-06-vnet01** |
| Backend Pool Configuration | **NIC** |
| IP Version | **IPv4** |
| Click **Add** to add a virtual machine |  |
| az104-06-vm0 | **check the box** |
| az104-06-vm1 | **check the box** |

Next and last we set Inbound rules with **Add a load balancing rule**. For that we use the following settings:

| Setting | Value |
| --- | --- |
| Name | **az104-06-lb4-lbrule1** |
| IP Version | **IPv4** |
| Frontend IP Address | **az104-06-pip4** |
| Backend pool | **az104-06-lb4-be1** |
| Protocol | **TCP** |
| Port | **80** |
| Backend port | **80** |
| Health probe | **Create new** |
| Name | **az104-06-lb4-hp1** |
| Protocol | **TCP** |
| Port | **80** |
| Interval | **5** |
| Unhealthy threshold | **2** |
| Close the create health probe window | **OK** |
| Session persistence | **None** |
| Idle timeout (minutes) | **4** |
| TCP reset | **Disabled** |
| Floating IP | **Disabled** |
| Outbound source network address translation (SNAT) | **Recommended** |

We can now click **Review and create** and if there are no errors we click **Create.**



After the deployment is completed, we go to resource and select **Frontend IP configuration** from the Load Balancer resource page. We copy the IP address and we past it in another browser tab.

With refresh of the page we can confirm that the message changes between the virtual machines.





Task 6: Implement Azure Application Gateway

In this task, we will implement an Azure Application Gateway in front of the two Azure virtual machines in the spoke virtual networks.

From the Azure portal we select **Virtual networks**. On the **Virtual networks** blade, in the list of virtual networks, we click **az104-06-vnet01. We navigate to Subnets and add a subnet with the following settings:**

| Setting | Value |
| --- | --- |
| Name | **subnet-appgw** |
| Subnet address range | **10.60.3.224/27** |

In the Azure portal we select **Application Gateways** and, on the **Application Gateways** blade we **Create** new with the following settings:

On the **Basic** tab:

| Setting | Value |
| --- | --- |
| Subscription | the name of the Azure subscription you are using in this lab |
| Resource group | **az104-06-rg5** (create new) |
| Application gateway name | **az104-06-appgw5** |
| Region | East US |
| Tier | **Standard V2** |
| Enable autoscaling | **No** |
| Instance count | **2** |
| Availability zone | **None** |
| HTTP2 | **Disabled** |
| Virtual network | **az104-06-vnet01** |
| Subnet | **subnet-appgw (10.60.3.224/27)** |

On the **Frontends** tab:

| Setting | Value |
| --- | --- |
| Frontend IP address type | **Public** |
| Public IP address | **Add new** |
| Name | **az104-06-pip5** |
| Availability zone | **None** |

On the **Backends** tab we add backend pool with the settings

| Setting | Value |
| --- | --- |
| Name | **az104-06-appgw5-be1** |
| Add backend pool without targets | **No** |
| IP address or FQDN | **10.62.0.4** |
| IP address or FQDN | **10.63.0.4** |

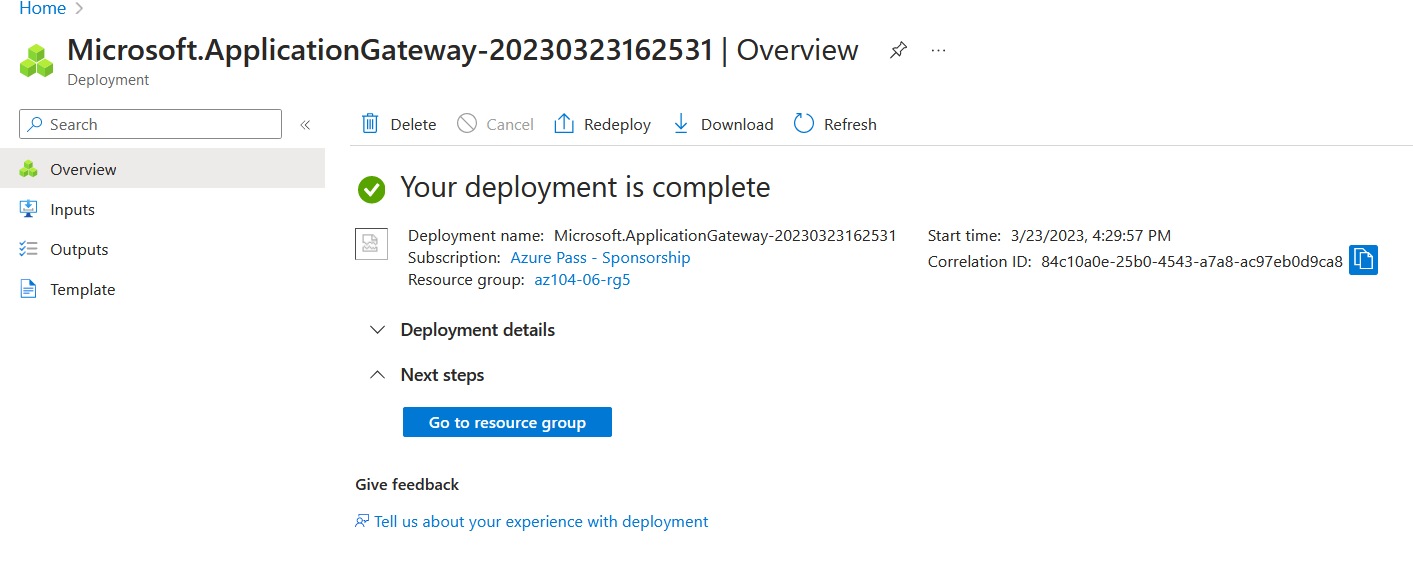
On the **Configuration** tab we add a routing rule with the settings:

| Setting | Value |
| --- | --- |
| Rule name | **az104-06-appgw5-rl1** |
| Priority | **10** |
| Listener name | **az104-06-appgw5-rl1l1** |
| Frontend IP | **Public** |
| Protocol | **HTTP** |
| Port | **80** |
| Listener type | **Basic** |
| Error page url | **No** |

Then we switch the **Backend targets** tab and specify the following settings:

| Setting | Value |
| --- | --- |
| Target type | **Backend pool** |
| Backend target | **az104-06-appgw5-be1** |
| Backend settings | **Add new** |
| Backend settings name | **az104-06-appgw5-http1** |
| Backend protocol | **HTTP** |
| Backend port | **80** |
| Additional settings | **take the defaults** |
| Host name | **take the defaults** |

After this we go to **Review + create** and then **Create.**

****

We then navigate to the resource, click on **az104-06-appgw5** Application Gateway blade, copy the **Frontend public IP address,** and past it in a browser to verify the message **Hello World from az104-06-vm2** or **Hello World from az104-06-vm3** with refreshing the window.

We can now clean up the resources.