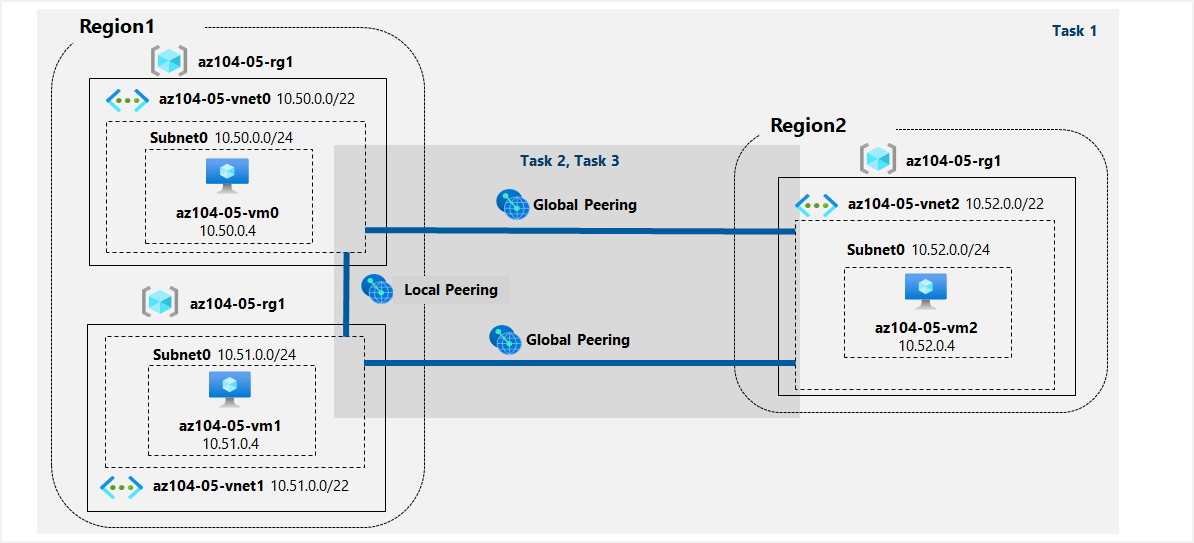
Lab 05 - Implement Intersite Connectivity

Contoso has its datacenters in Boston, New York, and Seattle offices connected via a mesh wide-area network links, with full connectivity between them. You need to implement a lab environment that will reflect the topology of the Contoso’s on-premises networks and verify its functionality.

## **Objectives**

* Task 1: Provision the lab environment
* Task 2: Configure local and global virtual network peering
* Task 3: Test intersite connectivity

Architecture diagram



Task 1: Provision the lab environment

In this task, we will deploy three virtual machines, each into a separate virtual network, with two of them in the same Azure region and the third one in another Azure region.

For this task we need to get the files “az104-05-vnetvm-loop-template.json” and “az104-05-vnetvm-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.

$location1 = 'eastus'

$location2 = 'westus'

$rgName = 'az104-05-rg1'

New-AzResourceGroup -Name $rgName -Location $location1

Then we run the following to create the three virtual networks and deploy virtual machines into them by using the template and parameter files that we uploaded previously.

New-AzResourceGroupDeployment `

-ResourceGroupName $rgName `

-TemplateFile $HOME/az104-05-vnetvm-loop-template.json `

-TemplateParameterFile $HOME/az104-05-vnetvm-loop-parameters.json `

-location1 $location1 `

-location2 $location2

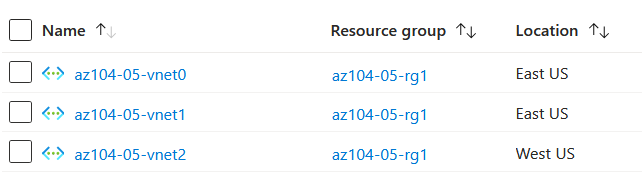
A picture containing text

Description automatically generated

Task 2: Configure local and global virtual network peering

In this task, we will configure local and global peering between the virtual networks we deployed in the previous tasks.

First, we verify that from the virtual networks we created two of them are in the same and one in different Azure region.



Then we go on the **az104-05-vnet0** virtual network blade, in the Settings section we click Peerings and then Add with the following peering settings:

| Setting | Value |
| --- | --- |
| This virtual network: Peering link name | **az104-05-vnet0\_to\_az104-05-vnet1** |
| This virtual network: Traffic to remote virtual network | **Allow (default)** |
| This virtual network: Traffic forwarded from remote virtual network | **Block traffic that originates from outside this virtual network** |
| Virtual network gateway | **None** |
| Remote virtual network: Peering link name | **az104-05-vnet1\_to\_az104-05-vnet0** |
| Virtual network deployment model | **Resource manager** |
| I know my resource ID | unselected |
| Subscription | the name of the Azure subscription you are using in this lab |
| Virtual network | **az104-05-vnet1** |
| 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** |

Then we add the second peering, but because I run into an issue with the Azure portal interface not displaying the virtual network **az104-05-vnet2** created in the previous task, I configured it by running the following PowerShell commands from Cloud Shell:

$rgName = 'az104-05-rg1'

$vnet0 = Get-AzVirtualNetwork -Name 'az104-05-vnet0' -ResourceGroupName $rgname

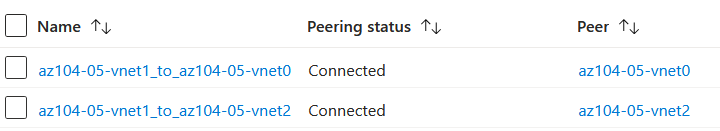
$vnet2 = Get-AzVirtualNetwork -Name 'az104-05-vnet2' -ResourceGroupName $rgname

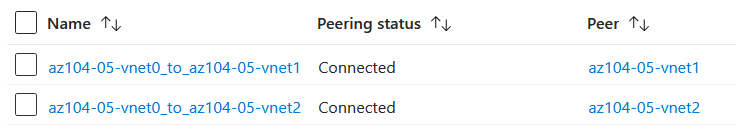
Add-AzVirtualNetworkPeering -Name 'az104-05-vnet0\_to\_az104-05-vnet2' -VirtualNetwork $vnet0 -RemoteVirtualNetworkId $vnet2.Id

Add-AzVirtualNetworkPeering -Name 'az104-05-vnet2\_to\_az104-05-vnet0' -VirtualNetwork $vnet2 -RemoteVirtualNetworkId $vnet0.Id

Now we navigate back to the Virtual Networks and select **az104-05-vnet1.** We go to the Settings section, click on Peerings and add with the following settings:

| Setting | Value |
| --- | --- |
| This virtual network: Peering link name | **az104-05-vnet1\_to\_az104-05-vnet2** |
| This virtual network: Traffic to remote virtual network | **Allow (default)** |
| This virtual network: Traffic forwarded from remote virtual network | **Block traffic that originates from outside this virtual network** |
| Virtual network gateway | **None** |
| Remote virtual network: Peering link name | **az104-05-vnet2\_to\_az104-05-vnet1** |
| Virtual network deployment model | **Resource manager** |
| I know my resource ID | unselected |
| Subscription | the name of the Azure subscription you are using in this lab |
| Virtual network | **az104-05-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** |





Task 3: Test intersite connectivity

In this task, we will test connectivity between virtual machines on the three virtual networks that we connected via local and global peering in the previous task.

In the Azure portal we search and select Virtual machines. From the list we select **az104-05-vm0** and click Connect with RDP then Download RDP File.

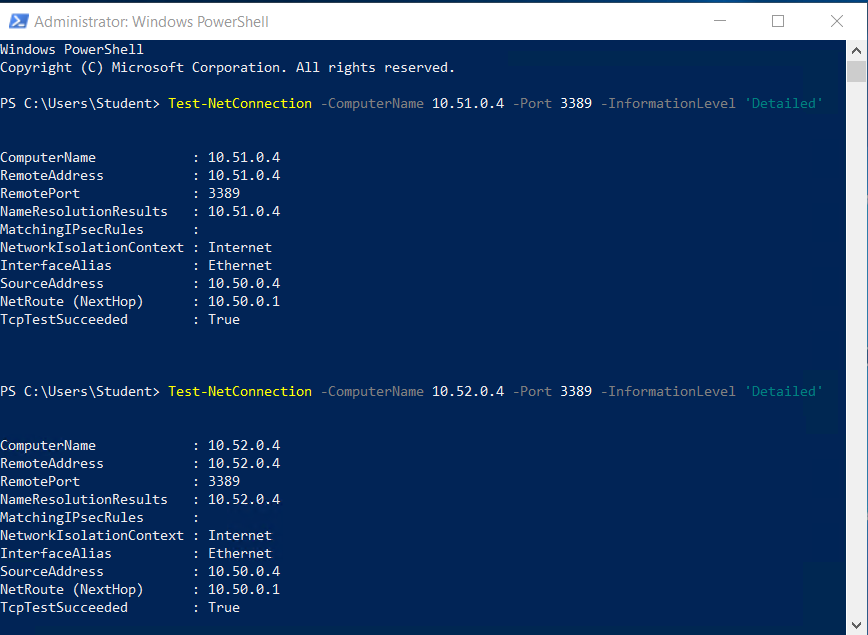
We log in with Student and the password that we set in the parameters file.

We open PowerShell as Admin and test the connectivity to **az104-05-vm1** with the following command:

Test-NetConnection -ComputerName 10.51.0.4 -Port 3389 -InformationLevel 'Detailed'

After we examine the output and verify that it was successful we test the connectivity to **az104-05-vm2** with the following command:

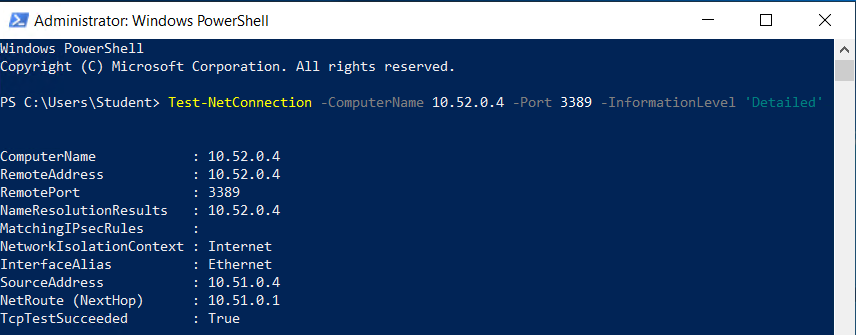
Test-NetConnection -ComputerName 10.52.0.4 -Port 3389 -InformationLevel 'Detailed'



Now we go back to the Virtual Machines blade and connect to **az104-05-vm1.**

We do the same steps and open PowerShell as Admin and test the connectivity to **az104-05-vm2** with the following command:

Test-NetConnection -ComputerName 10.52.0.4 -Port 3389 -InformationLevel 'Detailed'



Clean up resources

We clean up the created resources with the following command:

Get-AzResourceGroup -Name 'az104-05\*' | Remove-AzResourceGroup -Force -AsJob

After few minutes we can doublecheck if all resources are deleted from the Azure portal.