

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-3 : VNet Peering.

Date : 23/01/26

Objective :-

The objective of this task is to configure VNet peering between two Azure virtual networks. This enables secure and private communication between resources in different VNets. It helps improve network connectivity without using the public internet.

VNet Peering :-

Azure VNet Peering is a feature that connects two virtual networks in Azure, allowing resources in both networks to communicate privately using Azure's internal network. It provides low-latency, secure connectivity without using the public internet or gateways.

Types of Azure VNet Peering:

1. Regional VNet Peering :-

- Connects virtual networks in the same Azure region.
- Uses Azure's private network for fast and secure communication.
- Commonly used when resources are located in one region.

→ To implement the Regional Vnet peering, created two Virtual Networks VN01 and VN02 in the same region.

- Now to establish the peering connection between the two Vnets we need to peering to the Vnet 1 & 2.

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes the Microsoft Azure logo, an 'Upgrade' button, a search bar, and various account and service icons. The main title is 'Network foundation | Virtual networks'. On the left, a sidebar menu lists options like Overview, Favorites, Virtual networks, Virtual network, and more. The 'Virtual networks' option is selected. The main content area shows a table with columns: Name, Resource Group, Location, and Subscription. Two entries are listed: VN01 and VN02, both associated with RG01, Central India, and Azure subscription. There are also filter and search tools at the top of the table.

fig(1) Created two Virtual Networks.

The screenshot shows the Microsoft Azure portal interface, specifically the 'Peerings' section for Virtual Network VN02. The top navigation bar and sidebar are similar to the previous screenshot. The main title is 'VN02 | Peerings'. The sidebar shows 'Peerings' is selected. The main content area displays a table of peerings. One entry is shown: VN01-VN02, with a status of 'Fully Synchronized' and 'Connected'. Other columns include 'Peering sync status', 'Remote', 'Virtual', and 'Cross-tenant'.

fig(2) Successfully added Peering to Vnets.

- Now to connect the two Vnets, log in to the Virtual Machine with their public ip address. And use the Ping command to connect.

```

2.0.207.192.119 (azureadmin)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
3.20.207.192.119 (azureadmin) 4.52.172.252.253 (azureadmin)
Get:38 http://azure.archive.ubuntu.com/ubuntu noble-security/universe Translation-en [209 kB]
Get:39 http://azure.archive.ubuntu.com/ubuntu noble-security/universe amd64 Components [74.2 kB]
Get:41 http://azure.archive.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:42 http://azure.archive.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [28.8 kB]
Get:43 http://azure.archive.ubuntu.com/ubuntu noble-security/multiverse Translation-en [6492 B]
Get:44 http://azure.archive.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Get:45 http://azure.archive.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [396 B]
Reading package lists...
Building dependency tree...
Reading state information...
All packages are up to date.
root@VM01:/home/azureadmin# ping 10.2.1.4
PING 10.2.1.4 (10.2.1.4) 56(84) bytes of data.
64 bytes from 10.2.1.4: icmp_seq=1 ttl=64 time=6.27 ms
64 bytes from 10.2.1.4: icmp_seq=2 ttl=64 time=0.760 ms
64 bytes from 10.2.1.4: icmp_seq=3 ttl=64 time=2.26 ms
64 bytes from 10.2.1.4: icmp_seq=4 ttl=64 time=1.52 ms
64 bytes from 10.2.1.4: icmp_seq=5 ttl=64 time=7.61 ms
64 bytes from 10.2.1.4: icmp_seq=6 ttl=64 time=1.14 ms
64 bytes from 10.2.1.4: icmp_seq=7 ttl=64 time=1.01 ms
64 bytes from 10.2.1.4: icmp_seq=8 ttl=64 time=0.945 ms
64 bytes from 10.2.1.4: icmp_seq=9 ttl=64 time=0.963 ms
64 bytes from 10.2.1.4: icmp_seq=10 ttl=64 time=1.31 ms
64 bytes from 10.2.1.4: icmp_seq=11 ttl=64 time=0.950 ms
64 bytes from 10.2.1.4: icmp_seq=12 ttl=64 time=2.67 ms
^C
--- 10.2.1.4 ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 11039ms
rtt min/avg/max/mdev = 0.760/2.282/7.605/2.169 ms
root@VM01:/home/azureadmin#

```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobatext.mobatek.net>

fig(3) successfully connected two Vnets - Regional Vnet Peering

2. Global VNet Peering :-

- Connects virtual networks in different Azure regions.
 - Enables private and secure communication across regions
 - Useful for multi-region applications and disaster recovery setups.
- To implement the Global Vnet Peering, Create two Virtual Networks with two Virtual Machines in two different regions.

- Now we need to add peering to the Vnets.
- log in to the machines and establish the peering using the ping command

Eg:- ping <vm private ip>

The screenshot shows a MobaXterm window titled "20.106.34.211 (azureadmin)". The terminal session is running on VM02, with the command "ping 10.1.1.4" being executed. The output of the ping command is displayed, showing multiple ICMP packets being sent to the target IP address. The terminal window also shows the file structure of the user's home directory (~/.cache, .ssh, .bash_logout, .bashrc, .profile, .Xauthority).

```

Building dependency tree... Done
Reading state information... Done
All packages are up to date.
root@VM02:/home/azureadmin# ping 10.1.1.4
PING 10.1.1.4 (10.1.1.4) 56(84) bytes of data.
64 bytes from 10.1.1.4: icmp_seq=1 ttl=64 time=245 ms
64 bytes from 10.1.1.4: icmp_seq=2 ttl=64 time=241 ms
64 bytes from 10.1.1.4: icmp_seq=3 ttl=64 time=243 ms
64 bytes from 10.1.1.4: icmp_seq=4 ttl=64 time=242 ms
64 bytes from 10.1.1.4: icmp_seq=5 ttl=64 time=260 ms
64 bytes from 10.1.1.4: icmp_seq=6 ttl=64 time=242 ms
64 bytes from 10.1.1.4: icmp_seq=7 ttl=64 time=242 ms
64 bytes from 10.1.1.4: icmp_seq=8 ttl=64 time=242 ms
64 bytes from 10.1.1.4: icmp_seq=9 ttl=64 time=283 ms
64 bytes from 10.1.1.4: icmp_seq=10 ttl=64 time=241 ms
64 bytes from 10.1.1.4: icmp_seq=11 ttl=64 time=242 ms
64 bytes from 10.1.1.4: icmp_seq=12 ttl=64 time=249 ms
64 bytes from 10.1.1.4: icmp_seq=13 ttl=64 time=242 ms
64 bytes from 10.1.1.4: icmp_seq=14 ttl=64 time=242 ms
64 bytes from 10.1.1.4: icmp_seq=15 ttl=64 time=242 ms
64 bytes from 10.1.1.4: icmp_seq=16 ttl=64 time=266 ms
64 bytes from 10.1.1.4: icmp_seq=17 ttl=64 time=242 ms
64 bytes from 10.1.1.4: icmp_seq=18 ttl=64 time=241 ms
64 bytes from 10.1.1.4: icmp_seq=19 ttl=64 time=245 ms
64 bytes from 10.1.1.4: icmp_seq=20 ttl=64 time=274 ms
64 bytes from 10.1.1.4: icmp_seq=21 ttl=64 time=241 ms
^C
--- 10.1.1.4 ping statistics ---
21 packets transmitted, 21 received, 0% packet loss, time 20009ms
rtt min/avg/max/mdev = 241.355/247.867/283.467/11.927 ms
root@VM02:/home/azureadmin#

```

At the bottom of the terminal window, there is a status bar showing various system metrics such as CPU usage, memory, and network activity.

fig(4) Successfully established Global Vnet Peering.