Networking Lab 14

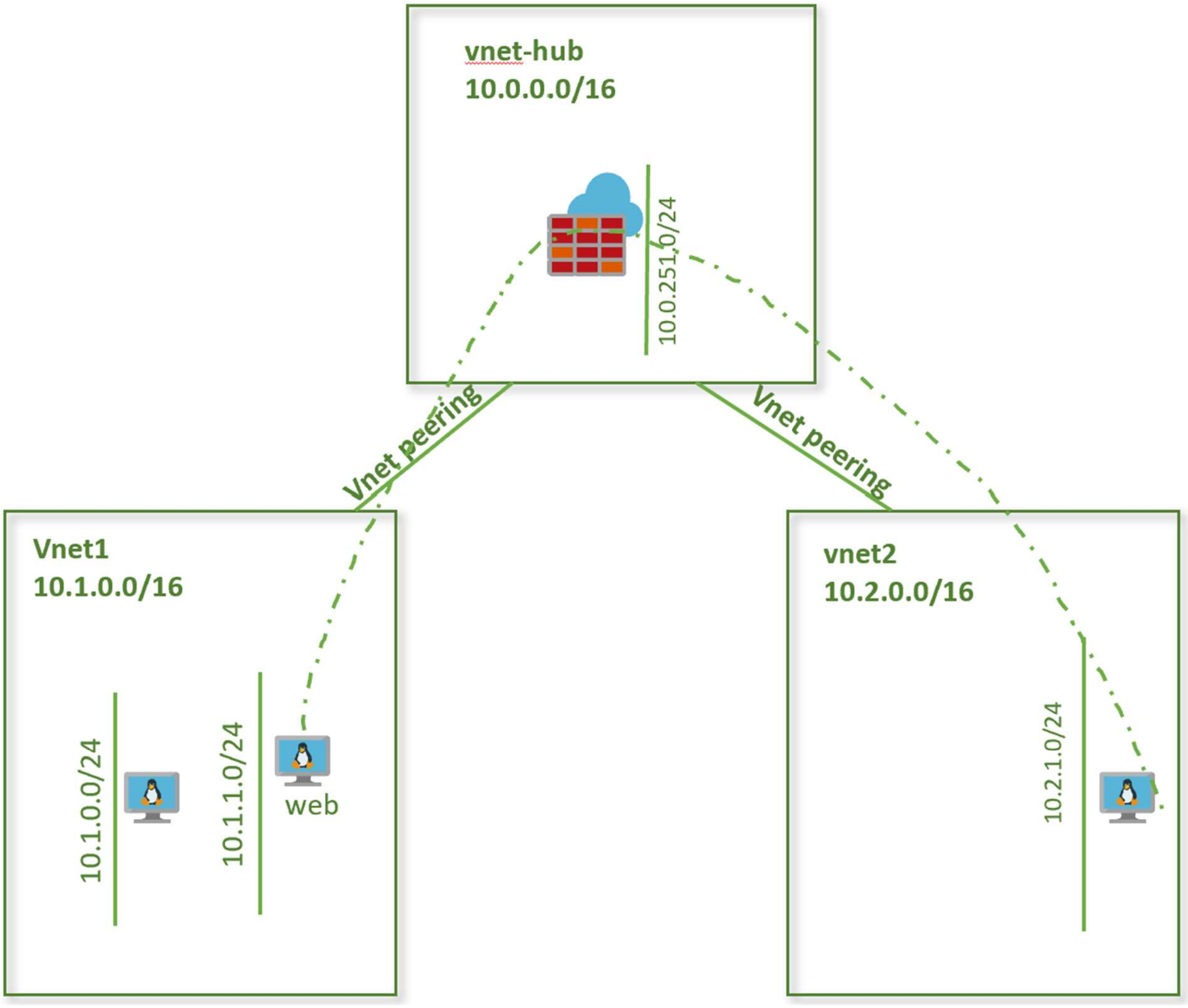
Azure Firewall

Spoke to spoke communication

Lab Overview

In this lab, we will how Azure firewall can enable traffic flow across spoke virtual networks through centralized firewall control in the hub virtual network. We have a hub and spoke topology as shown in the diagram.

Lab Diagram



Lab setup

From the previous firewall lab, we have a firewall deployed in the hub virtual network vnet-hub. We have two spoke vnets, vnet1 and vnet2. Vnet1 has a default route pointing to the firewall. We want to enable virtual machines in vnet1 to be able to talk to vnet2. For this we need to add a route to vnet1's networks with a next hop as the Azure firewall.

Create virtual network vnet2

If you already have vnet2 created, skip this step and go to the next step.

```
ResourceGroup=rg-lab
VnetName=vnet2
VnetPrefix=10.2.0.0/16
SubnetName=vnet2-subnet1
SubnetPrefix=10.2.1.0/24
Location=westus2
az network vnet create -g $ResourceGroup -n $VnetName --address-prefix $VnetPrefix --subnet-name
$SubnetName --subnet-prefix $SubnetPrefix -l $Location
```

Verify vnet vnet2 is created:

```
az network vnet list -o table
Name     ResourceGroup    Location    NumSubnets    Prefixes      DnsServers    DDOSProtection
VMProtection
--------  ---------------  ----------  ------------  -------------  ------------  ----------------  --------------
onprem    rg-lab1        westus2     2          10.128.0.0/16            False          False
vnet-hub  rg-lab1        westus2     3          10.0.0.0/16             False          False
vnet1     rg-lab1        westus2     2          10.1.0.0/16             False          False
vnet2     rg-lab1        westus2     1          10.2.0.0/16             False          False
```

Peer spoke virtual network vnet2 with hub vnet-hub

If you already have vnet2 peering created, skip this step and go to the next step.

Create peering on vnet2 side:

```
PeeringName=peer-vnet2-vnet-hub
VnetName=vnet2
RemoteVnet=vnet-hub
```

```
az network vnet peering create --name $PeeringName --remote-vnet $RemoteVnet --resource-group
$ResourceGroup --vnet-name $VnetName --allow-forwarded-traffic  --allow-vnet-access
```

Create peering on vnet-hub side:

```
PeeringName=peer-vnethub-to-vnet2
VnetName=vnet-hub
RemoteVnet=vnet2
```

```
az network vnet peering create --name $PeeringName --remote-vnet $RemoteVnet --resource-group
$ResourceGroup --vnet-name $VnetName --allow-forwarded-traffic  --allow-vnet-access
```

Ensure Allow Forwarded traffic is enabled for vnet1 peering:

```
PeeringName=peer-vnet1-to-vnet-hub
VnetName=vnet1
ResourceGroup=rg-lab
```

```
az network vnet peering update -g $ResourceGroup -n $PeeringName --vnet-name $VnetName --set
allowForwardedTraffic=true
```

```
PeeringName=peer-vnet-hub-to-vnet1
VnetName=vnet-hub
ResourceGroup=rg-lab
```

```
az network vnet peering update -g $ResourceGroup -n $PeeringName --vnet-name $VnetName --set
allowForwardedTraffic=true
```

Verify peering status between vnet2 and vnet-hub:

Run the following commands in the cloud shell:

```
VnetName=vnet1
ResourceGroup=rg-lab
az network vnet peering  list -g $ResourceGroup  --vnet-name $VnetName -o table
AllowForwardedTraffic    AllowGatewayTransit    AllowVirtualNetworkAccess    Name               PeeringState    ProvisioningState
ResourceGroup    UseRemoteGateways
-----------------------  ---------------------  ---------------------------  ----------------------  --------------  -------------------  ---------------  -------------------
-
True                  False               True                    peer-vnet1-to-vnet-hub  Connected       Succeeded          rg-lab1         False
```

```
binal@Azure:~$ VnetName=vnet2
binal@Azure:~$ az network vnet peering  list -g rg-lab1  --vnet-name $VnetName -o table
AllowForwardedTraffic    AllowGatewayTransit    AllowVirtualNetworkAccess    Name             PeeringState    ProvisioningState
ResourceGroup    UseRemoteGateways
-----------------------  ---------------------  ---------------------------  -------------------  --------------  -------------------  ---------------  -------------------
True                  False               True                    peer-vnet2-vnet-hub  Connected       Succeeded          rg-lab1         False
```

Add virtual machine to vnet2

```
Nsg=nsg-hub
az network vnet subnet update -g $ResourceGroup -n $SubnetName --vnet-name $VnetName --network-security-group $Nsg

ResourceGroup=rg-lab
VmName=vnet2-vm1
SubnetName=vnet2-subnet1
AdminUser=azureuser
AdminPassword=Azure123456!
az vm create --resource-group $ResourceGroup --name $VmName --image UbuntuLTS --vnet-name $VnetName --subnet $SubnetName --admin-username $AdminUser --admin-password $AdminPassword --nsg ""
```

Create a route in the spoke vnet2

From the previous lab, we have a route table **udr-to-fw** created in region West US 2. We will use the same route table and associate to subnet vnet2-subnet1, to add a default route to the firewall in vnet2 as well.

Associate the route table to the subnet.

1. On the Route Tables page, select **udr-to-fw** and then select **Subnets**.
2. Select **Associate**.
3. Select **Choose a virtual network**.
4. Select **vnet2**.
5. Select **vnet2-subnet1**.
6. Select **OK**.

Add firewall rule to allow ICMP traffic

1. From the Azure portal, go to the Firewalls page and click on firewall **vnet-hub-fw**.
2. Under Settings, click on Rules → **Network rule collection** → **Add network rule collection**.
3. Add a rule with the following values:
   Name: allow-icmp
   Priority: 200
   Action: Allow

Under Rules → IP Addresses:
Name: allow-icmp
Protocol: ICMP
Source Addresses: 10.0.0.0/8
Destination addresses: 10.0.0.0/8
Destination Ports: *

Verify reachability between the two spokes

1. Connect via serial console to vm vnet1-vm-mgmt1.
2. Ping the private IP address of vm vnet2-vm1.
3. Verify the pings work successfully.

Conclusion

Note that these two vnets are not directly peered. We enabled the flow between the two vnets that are not directly peered. We are leveraging Azure Firewall to control flow across multiple spoke virtual networks.