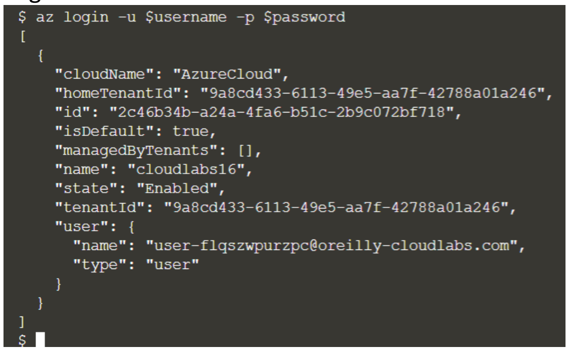
In this lab, you will learn how to create a new subnet in an Azure VNet using the Azure CLI.

Learning Objectives
In this lab, you will learn how to do the following:

Log in to the Azure CLI
Create a new VNet subnet
Confirm that the new subnet was successfully created
Clean up the subnet

Login to azure cli :



```
$ az login -u $username -p $password
[
  {
    "cloudName": "AzureCloud",
    "homeTenantId": "9a8cd433-6113-49e5-aa7f-42788a01a246",
    "id": "2c46b34b-a24a-4fa6-b51c-2b9c072bf718",
    "isDefault": true,
    "managedByTenants": [],
    "name": "cloudlabs16",
    "state": "Enabled",
    "tenantId": "9a8cd433-6113-49e5-aa7f-42788a01a246",
    "user": {
      "name": "user-flqszwpurzpc@oreilly-cloudlabs.com",
      "type": "user"
    }
  }
]
$
```

In the next step, we will elaborate on subnets and create a second subnet in your VNet using the Azure CLI.

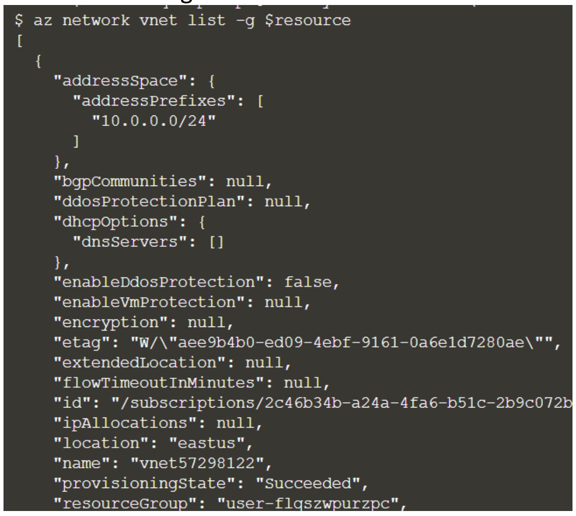
Understanding Azure Virtual Network Subnets
Azure Virtual Network is the foundation of private networks in Microsoft Azure. Many Azure services support deployment to an Azure Virtual Network. This enables the resources to communicate with each other and other networks in a more secure way.

VNets can be segmented into smaller subnetworks using virtual subnets. Azure resources can then get deployed into these subnets or allow traffic to/from them. You can protect subnets by assigning Azure network security groups (NSGs) to them.

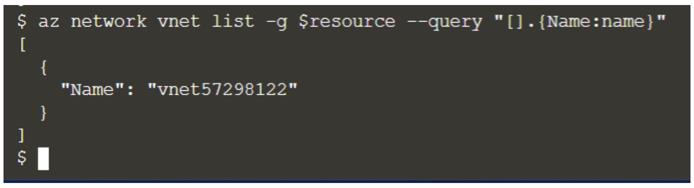
Similar to other computer networks, you need to specify an address range (a.k.a. address space) when creating a VNet. You also have the option to divide your VNet to different address spaces using subnets.

Note: You can create multiple subnets within the same VNet. Just make sure the subnet address spaces don't conflict. Similar to Azure VNets, you can use the CIDR (Classless Inter-Domain Routing) format to define address spaces for the net subnets.

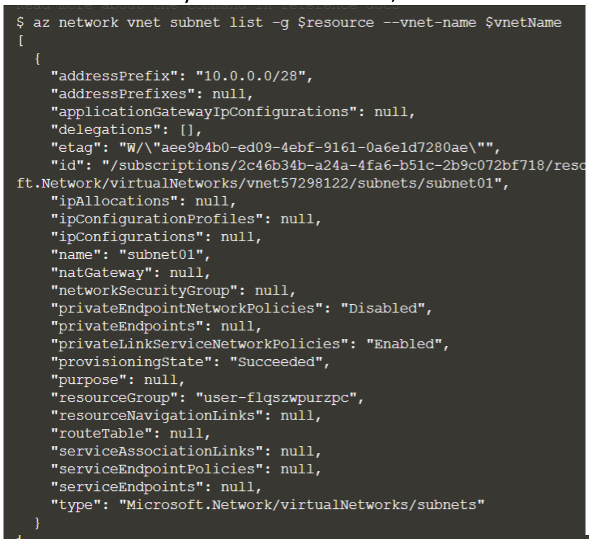
Use the following command to confirm that a VNet is successfully created for you:

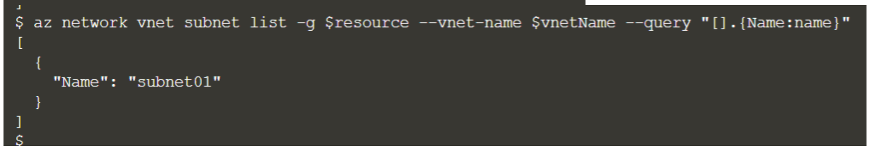


```
$ az network vnet list -g $resource
[
  {
    "addressSpace": {
      "addressPrefixes": [
        "10.0.0.0/24"
      ]
    },
    "bgpCommunities": null,
    "ddosProtectionPlan": null,
    "dhcpOptions": {
      "dnsServers": []
    },
    "enableDdosProtection": false,
    "enableVmProtection": null,
    "encryption": null,
    "etag": "W/\"aee9b4b0-ed09-4ebf-9161-0a6e1d7280ae\"",
    "extendedLocation": null,
    "flowTimeoutInMinutes": null,
    "id": "/subscriptions/2c46b34b-a24a-4fa6-b51c-2b9c072b
    "ipAllocations": null,
    "location": "eastus",
    "name": "vnet57298122",
    "provisioningState": "Succeeded",
    "resourceGroup": "user-flqszwpurzpc",
```


```
$ az network vnet list -g $resource --query "[].{Name:name}"
[
  {
    "Name": "vnet57298122"
  }
]
$
```

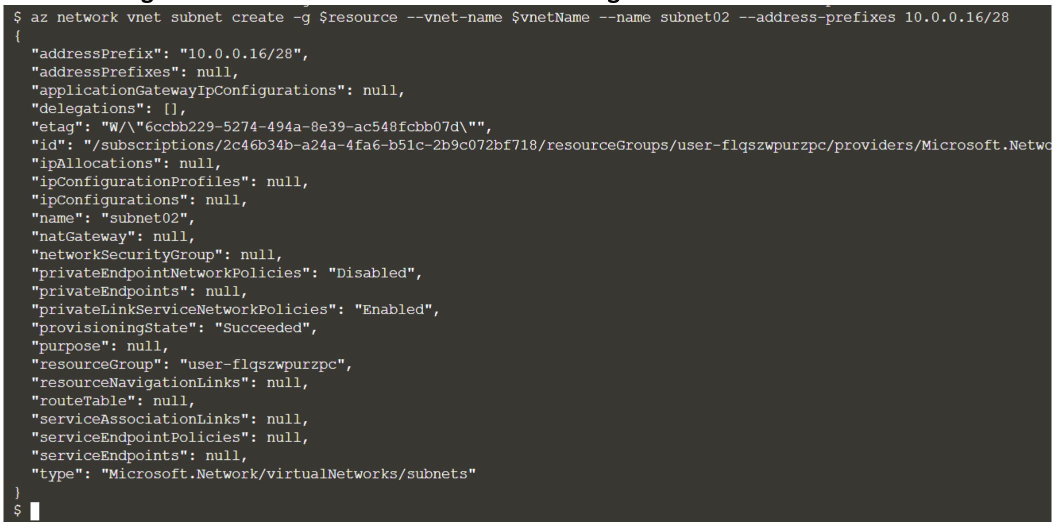
This VNet already contains a subnet, named subnet01. You can confirm this by running the following query:


```
$ az network vnet subnet list -g $resource --vnet-name $vnetName
[
  {
    "addressPrefix": "10.0.0.0/28",
    "addressPrefixes": null,
    "applicationGatewayIpConfigurations": null,
    "delegations": [],
    "etag": "W/\"aee9b4b0-ed09-4ebf-9161-0a6e1d7280ae\"",
    "id": "/subscriptions/2c46b34b-a24a-4fa6-b51c-2b9c072bf718/resc
ft.Network/virtualNetworks/vnet57298122/subnets/subnet01",
    "ipAllocations": null,
    "ipConfigurationProfiles": null,
    "ipConfigurations": null,
    "name": "subnet01",
    "natGateway": null,
    "networkSecurityGroup": null,
    "privateEndpointNetworkPolicies": "Disabled",
    "privateEndpoints": null,
    "privateLinkServiceNetworkPolicies": "Enabled",
    "provisioningState": "Succeeded",
    "purpose": null,
    "resourceGroup": "user-flqszwpurzpc",
    "resourceNavigationLinks": null,
    "routeTable": null,
    "serviceAssociationLinks": null,
    "serviceEndpointPolicies": null,
    "serviceEndpoints": null,
    "type": "Microsoft.Network/virtualNetworks/subnets"
  }
```


```
$ az network vnet subnet list -g $resource --vnet-name $vnetName --query "[].{Name:name}"
[
  {
    "Name": "subnet01"
  }
]
$
```

Now, let's create the second Azure subnet in our VNet using the Azure CLI.

Creating a New Subnet in Our Azure Virtual Network
The following command creates a subnet in the existing VNet:


```
$ az network vnet subnet create -g $resource --vnet-name $vnetName --name subnet02 --address-prefixes 10.0.0.16/28
{
  "addressPrefix": "10.0.0.16/28",
  "addressPrefixes": null,
  "applicationGatewayIpConfigurations": null,
  "delegations": [],
  "etag": "W/\"6ccbb229-5274-494a-8e39-ac548fcbb07d\"",
  "id": "/subscriptions/2c46b34b-a24a-4fa6-b51c-2b9c072bf718/resourceGroups/user-flqszwpurzpc/providers/Microsoft.Netwo
  "ipAllocations": null,
  "ipConfigurationProfiles": null,
  "ipConfigurations": null,
  "name": "subnet02",
  "natGateway": null,
  "networkSecurityGroup": null,
  "privateEndpointNetworkPolicies": "Disabled",
  "privateEndpoints": null,
  "privateLinkServiceNetworkPolicies": "Enabled",
  "provisioningState": "Succeeded",
  "purpose": null,
  "resourceGroup": "user-flqszwpurzpc",
  "resourceNavigationLinks": null,
  "routeTable": null,
  "serviceAssociationLinks": null,
  "serviceEndpointPolicies": null,
  "serviceEndpoints": null,
  "type": "Microsoft.Network/virtualNetworks/subnets"
}
$
```

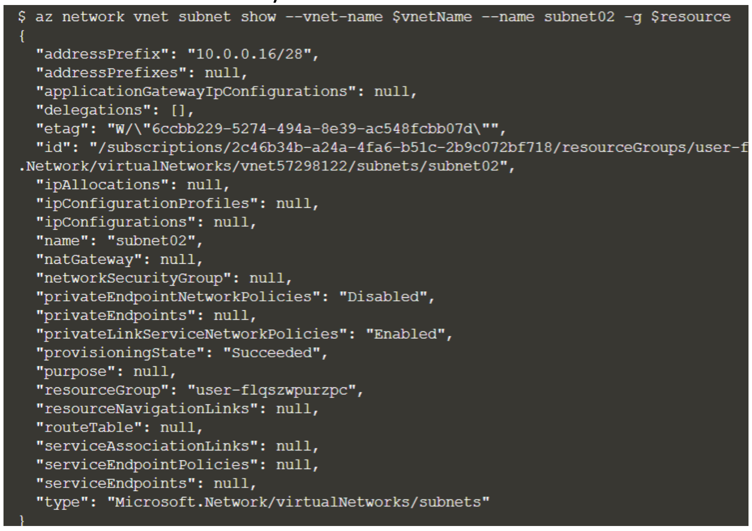
The command creates a new subnet named subnet02, with the CIDR address of 10.0.0.16/28 (15 addresses).

Note: Our VNet already contains subnet01 with the 10.0.0.0/28 address range (IPs between 10.0.0.0 and 10.0.0.15). To avoid conflicts, we configured subnet02 to use the 10.0.0.16/28 CIDR range (meaning IP ranges between 10.0.0.16 and 10.0.0.31).

Here are a few parameters for this command:

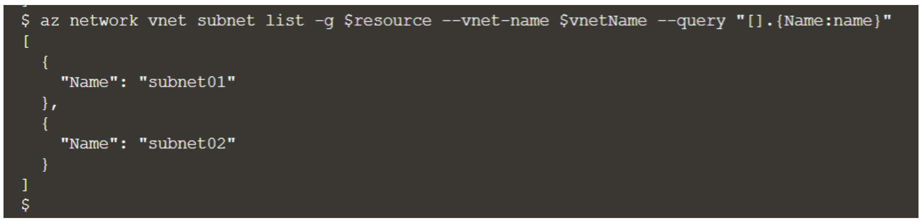
--resource-group: Name of the VNet Resource Group
--name: Name of the new subnet
--vnet-name: Name of the existing parent VNet
--address-prefix: List of the subnet IP addresses in CIDR (address space)

Confirm the Subnet is Created
It's time to see if we have correctly created our subnet. Use the following command to confirm that the new subnet is created within your VNet:



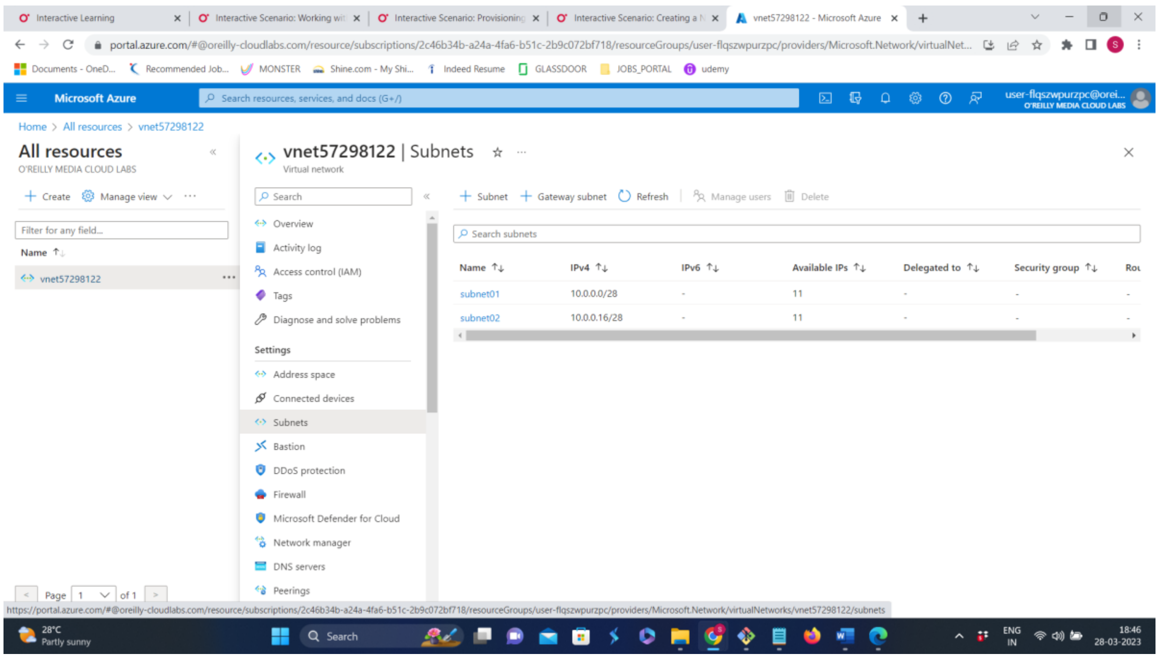
```
$ az network vnet subnet show --vnet-name $vnetName --name subnet02 -g $resource
{
  "addressPrefix": "10.0.0.16/28",
  "addressPrefixes": null,
  "applicationGatewayIpConfigurations": null,
  "delegations": [],
  "etag": "W/\"6ccbb229-5274-494a-8e39-ac548fcbb07d\"",
  "id": "/subscriptions/2c46b34b-a24a-4fa6-b51c-2b9c072bf718/resourceGroups/user-f
.Network/virtualNetworks/vnet57298122/subnets/subnet02",
  "ipAllocations": null,
  "ipConfigurationProfiles": null,
  "ipConfigurations": null,
  "name": "subnet02",
  "natGateway": null,
  "networkSecurityGroup": null,
  "privateEndpointNetworkPolicies": "Disabled",
  "privateEndpoints": null,
  "privateLinkServiceNetworkPolicies": "Enabled",
  "provisioningState": "Succeeded",
  "purpose": null,
  "resourceGroup": "user-flqszwpurzpc",
  "resourceNavigationLinks": null,
  "routeTable": null,
  "serviceAssociationLinks": null,
  "serviceEndpointPolicies": null,
  "serviceEndpoints": null,
  "type": "Microsoft.Network/virtualNetworks/subnets"
}
```

Running the following command gives you all the subnets within our VNet. Confirm that you see both subnet01 and subnet02 in the result:



```
$ az network vnet subnet list -g $resource --vnet-name $vnetName --query "[].{Name:name}"
[
  {
    "Name": "subnet01"
  },
  {
    "Name": "subnet02"
  }
]
$
```

You can find your Azure VNet and its subnets in the Azure portal.



portal.azure.com/#@oreilly-cloudlabs.com/resource/subscriptions/2c46b34b-a24a-4fa6-b51c-2b9c072bf718/resourceGroups/user-flqszwpurzpc/providers/Microsoft.Network/virtualNet...

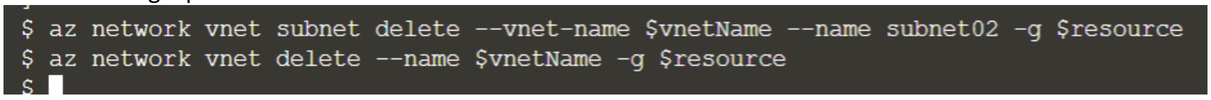
Home > All resources > vnet57298122

vnet57298122 | Subnets
Virtual network

Name	IPv4	IPv6	Available IPs	Delegated to	Security group	Rou
subnet01	10.0.0.0/28	-	11	-	-	-
subnet02	10.0.0.16/28	-	11	-	-	-

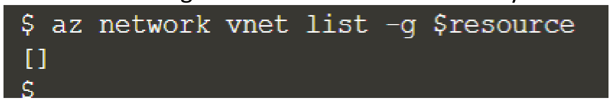
Both subnet are created .

Delete the Azure Virtual Network Subnet
Use the following command to clean up the subnet from your VNet:
Now, use the following command to clean up the parent VNet as well:
Note: Deleting a parent VNet will delete all the child subnets as well.



```
$ az network vnet subnet delete --vnet-name $vnetName --name subnet02 -g $resource
$ az network vnet delete --name $vnetName -g $resource
$
```

Use the following command to list all VNets in your allocated resource group:



```
$ az network vnet list -g $resource
[]
$
```