aws training and certification

Networking - Configuring Intra Region Amazon VPC peering

Lab Overview

In this lab, you will create Amazon Virtual Private Cloud (Amazon VPC) peering by configuring network routing between two Amazon VPC's located in same AWS Region. Each Amazon VPC contains two Amazon EC2 instances. These EC2 instances will be used to test network connectivity between the Amazon VPCs. You will use Amazon VPC peering to configure network connectivity.

Let's get a quick overview of what is Amazon VPC peering.

Amazon Virtual Private Cloud (Amazon VPC) Peering connection is a networking connection between two VPCs that enables you to route traffic between them using private IPv4 addresses or IPv6 addresses. Instances in either VPC can communicate with each other as if they are within the same network. AWS uses the existing infrastructure of a VPC to create a VPC peering connection; it is neither a gateway nor a VPN connection, and does not rely on a separate piece of physical hardware. You can create a VPC peering connection between your own VPCs, or with a VPC in another AWS account. You can establish peering relationships between VPCs across different AWS Regions (also called inter-Region VPC peering). This allows VPC resources including EC2 instances, Amazon RDS databases and Lambda functions that run in different AWS Regions to communicate with each other using private IP addresses, without requiring gateways, VPN connections, or separate network appliances. The traffic remains in the private IP space. All inter-region traffic is encrypted with no single point of failure, or bandwidth bottleneck. Traffic always stays on the global AWS backbone, and never traverses the public internet, which reduces threats, such as common exploits, and DDoS attacks. Inter-Region VPC Peering provides a simple and cost-effective way to share resources between regions or replicate data for geographic redundancy.

For more information about VPC you can view the AWS Documentation using the following link: https://docs.aws.amazon.com/vpc/latest/peering/what-is-vpc-peering.html

Topics Covered

- Create a second Amazon Virtual Private Cloud (VPC)
- Create a public subnet

- Create an Internet gateway
- Create a Route Table and added a route to the Internet
- Configure the security groups of the two instances in both VPCs to allow network traffic across Amazon VPC peering connection
- Launch an instance for testing network connectivity between VPCs
- Create Intra Regional Amazon VPC peering connections
- Configure network traffic routing across Amazon VPC peering connection
- Test network connectivity across VPC peering connections
- Clean up the resources that were created in the 2 labs

Prerequisites

- To successfully complete this lab, you should have the resources available in your account that were deployed in the previous lab **Networking - Configuring and Deploying Amazon VPC for a Web server**
- You should be familiar with basic navigation of the AWS Management Console.
- Given that you have completed the previous lab, you should now have a basic understanding of Amazon VPC, IP addressing, Security Group, Network ACL, Route Table and Internet Gateway.

Lab environment for VPC peering activity

This lab starts with only one AWS Region, US-EAST-1 (N. Virginia Region).

- VPC-1 (it should already exist from previous lab) and VPC-2 (you will build this VPC in this lab) are in the us-east-1 N. Virginia Region.
- Each VPC has a public subnet with Amazon EC2 resources deployed in each subnet

Task 1: Create a VPC

In this task, you will create second VPC in the AWS Cloud.

1. In the AWS Management Console, choose **US-EAST-1 (N. Virginia Region)**.
2. Choose 

and select **VPC**. (You may also type VPC in the search bar and choose VPC.).

 If you see **New VPC Experience** at the top-left of your screen, ensure **New VPC Experience** is selected. This lab is designed to use the new VPC Console.

3. Choose **Your VPCs** on the left navigation menu.

Note: You will see a default VPC (one is created whenever an AWS account is created) and a VPC named **VPC-1** which we created in the previous lab.

4. Choose **Create VPC** on the right side of the console.
5. In the Create VPC section, enter the following:

   - **Name tag**: Enter VPC-2
   - **IPv4 CIDR block**: Choose a CIDR range in 192.168.0.0 network with mask in such a way that it provides 256 number of IP addresses

Hint: If you want to learn about subnetting, go to the **Appendix** section at the end of this document.

Note: In the first lab we chose a private CIDR range of 10.0.0.0 and in this lab we are choosing a different private CIDR range of 192.168.0.0. We are doing this because you cannot create a VPC peering connection between VPCs that have matching or overlapping IPv4 or IPv6 CIDR blocks. Amazon always assigns your VPC a unique IPv6 CIDR block. If your IPv6 CIDR blocks are unique but your IPv4 blocks are not, you cannot create the peering connection.

This VPC will not have an IPv6 CIDR block, and we will leave it with default tenancy.

6. Choose **Create VPC**

A VPC with the specified CIDR block has been created. Now, let's create the subnets.

Task 2: Create Your Public Subnet

In this task, you will create one public subnet in the Lab VPC. The public subnets will be for internet-facing resources.

7. In the left navigation pane, choose **Subnets**.

Note: You will see subnets for the default VPC and VPC-1. You can ignore them and go to the next step.

8. Choose **Create subnet** and configure it with the following details:

   - **VPC ID**: Select *VPC-2*
   - **Subnet name**: Enter VPC-2 PublicSubnet

- **Availability Zone**: Select the first Availability Zone in the list
- **IPv4 CIDR block**: Choose a CIDR range in 192.168.0.0 network with mask in such a way that it provides 16 number of IP addresses

9. Choose **Create subnet**
10. Select **VPC-2 PublicSubnet**.
11. In the **Actions** menu, select **Edit subnet setting**, then configure:

- Select ☑ **Enable auto-assign public IPv4 address**
- Click **Save**

Enable auto-assign public IPv4 address provides a public IPv4 address for all instances launched into the selected subnet

Note: When you create a VPC, you must specify an IPv4 CIDR block for the VPC. The CIDR block of a subnet can be the same as the CIDR block for the VPC (for a single subnet in the VPC) or a subset of the CIDR block for the VPC (for multiple subnets). If you create more than one subnet in a VPC, the CIDR blocks of the subnets cannot overlap. Even though we named the subnet *VPC-2 PublicSubnet*, they are not yet public. A public subnet must have an internet gateway, which you will create and attach in the lab.

Task 3: Create an Internet Gateway

In this task, you will create an internet gateway so that internet traffic can access the public subnets.

12. In the left navigation pane, choose **Internet Gateways**.

Note: A default internet gateway was created with the default VPC. You can ignore this and proceed with the next step.

13. Choose **Create internet gateway** and configure:

- **Name tag**: Enter `VPC-2 InternetGateway`

14. Choose **Create internet gateway**

Once it's created, you need to attach the internet gateway to your Lab VPC.

15. Choose Actions > **Attach to VPC**.

16. For **VPC**, select *VPC-2*

17. Choose **Attach internet gateway**

The internet gateway is now attached to your Lab VPC. Even though you created an Internet gateway and attached it to your VPC, you still have to tell instance within your public subnet how to get to the Internet.

Task 4: Create a Route Table, Add Routes, And Associate Public Subnets

To use an internet gateway, a subnet's route table must contain a route that directs internet-bound traffic to the internet gateway. This subnet is called a *public subnet*.

In this task, you will create route table, add routes and associate public subnets to your second VPC **VPC-2.**

18. In the left navigation pane, choose **Route Tables**.

Several route tables are displayed, but there is only one route table associated with the Lab VPC. This is the main route table.

19. Choose Create route table and configure:

- **Name tag**: Enter VPC-2 PublicRouteTable
- **VPC**: Select *VPC-2*

20. Choose **Create route table**

21. Select **Routes** tab for *VPC-2 PublicRouteTable*.

22. Choose **Edit routes**

Now, add a route to direct internet-bound traffic (0.0.0.0/0) to the internet gateway.

23. Choose **Add route** and configure:

- **Destination**: Enter 0.0.0.0/0
- **Target**: Select *Internet Gateway* and *VPC-2 InternetGateway*.

24. Choose Save changes

The last step is to associate this new route table with the public subnets.

25. Choose the **Subnet Associations** tab.

26. Choose **Edit subnet associations**

27. Select the rows with *VPC-2 PublicSubnet*.

28. Choose **Save associations**

The *VPC-2 PublicSubnet* is now public because it has a route table entry that sends traffic to the internet via the internet gateway.

Task 5: Create and edit the Security Groups of the instances in VPC-1 and VPC-2

In this task, you will create a security group for the EC2 instance in second VPC **VPC-2** so that the instance in **VPC-1** can access this instance over peering connection. You will also edit the security group of **VPC-1** so that you can SSH to the instance in **VPC-1** for testing the network connectivity across peering connection.

Create a Security Group for the instance in VPC-2

29. In the left navigation pane, choose **Security Groups**.

30. Create another new security group with the following details:

- **Security group name**: Enter VPC-2 SG
- **Description**: Enter Allows ICMP ping from VPC-1 Instance
- **VPC**: Select *VPC-2*

31. For **Inbound rules**, choose **Add rule** and configure it with the following details:

- **Type**: Select *All ICMP - IPv4*
- **Source**: Select *Custom*. For CIDR blocks, provide the private CIDR range of *VPC-1* that was created in previous lab.

Hint: If you followed the instructions of the previous lab, then the VPC-1 CIDR range should be 10.0.0.0/24.

32. Create another **Inbound rule** with the following details:
    - **Type**: Select *SSH*
    - **Source**: Select *Anywhere-IPv4*

33. Create a new tag with the following details:

    - **Key**: Enter Name
    - **Value**: Enter VPC-2 SG

34. Choose Create security group

You have configured the inbound rules to permit ICMP traffic from the EC2 instance in VPC-1 to the EC2 instance in VPC-2.

Edit the Security Group for the instance in VPC-1

35. In the left navigation pane, choose **Security Groups**.
36. Select the Security Group named **VPC-1 SG**
37. Choose Inbound rules and click Edit inbound rules and configure it with the following details:

    - **Type**: Select *SSH*
    - **Source**: Select *Anywhere-IPv4*.

38. Create another **Inbound rule** with the following details:
    - **Type**: Select *All ICMP - IPv4*
    - **Source**: Select *Custom*. For CIDR blocks, provide the private CIDR range of *VPC-2* that was created in this lab

39. Choose Save rules

Task 6: Launch an Instance in your Public Subnet

In this task, you provision an EC2 instance in the public subnet of VPC-2 which we will use to test network connectivity between two VPCs across peering connection.

40. On the Services menu, click **EC2**.

If you see **New EC2 Experience** at the top-left of your screen, ensure **New EC2 Experience** is selected. This lab is designed to use the new EC2 Console.

41. Click **Launch instance** > **Launch instance**.
42. On **Step 1**, click **Select** next to **Amazon Linux 2 AMI**.

You will launch a t2.micro instance. This instance type has 1 vCPU and 1 GiB of memory.

43. On **Step 2**, click **Next: Configure Instance Details**
44. On **Step 3**, configure:

- **Network**: *VPC-2*
- **Subnet**: *VPC-2 PublicSubnet*

45. Click **Next: Add Storage**
46. On **Step 4**, click **Next: Add Tags**
47. On **Step 5**, click **Add Tag** then configure:

- **Key:** Name
- **Value:** VPC-2 Instance

48. Click **Next: Configure Security Group**
49. Configure the following:

- Click ⊙ **Select an existing security group**
- Select **VPC-2 SG**
- Click **Review and Launch**

50. At the **Warning** screen, click **Continue**
51. On **Step 7**, review the settings, then click **Launch**
52. On the **Select an existing key pair or create a new key pair** window, configure the following:

- Select **Choose an existing key pair**

- Select a Key Pair named **VirginiaKeyPair** from drop-down menu that we created in the previous lab
- Select ☑ **I acknowledge that...**
- Click **Launch Instances**

53. Click **View Instances**

This brings you to the **Instances** window where you can watch your new instance launch and view its details.

54. Wait for your web server to fully launch. It should display the following:

- **Instance State:** Running.

You can click the refresh icon to refresh your instances status.

Task 7: Create Intra Region Amazon VPC peering connection between VPC-1 and VPC-2

To establish a VPC peering connection, you do the following:

- The owner of the requester VPC sends a request to the owner of the accepter VPC to create the VPC peering connection. The accepter VPC can be owned by you, or another AWS account, and cannot have a CIDR block that overlaps with the CIDR block of the requester VPC.
- The owner of the accepter VPC accepts the VPC peering connection request to activate the VPC peering connection.

In this task, you will configure an Inter-Region (within the same region) VPC peering connection between VPC-1 and VPC-2 which do not have overlapping CIDR blcoks.

55. On the Services menu, click **VPC**.

56. On the left navigation pane, choose **Peering Connections**.

57. Choose **Create peering connection**. Then configure:

- **Name**: VPC-Peering
- **VPC ID (Requester)**: In the drop down list select **VPC-1**.
- **Account**: My account.
- **Region**: Select ‘This region (us-east-1)'
- **VPC ID (Accepter)**: In the drop down list select **VPC-2**.

58. Choose **Create peering connection**

Note: Even though the peering connection completes successfully. The connection needs to be accepted. Notice the status is *Pending Acceptance*.

59. With the peering connection selected, choose the Actions button, and then choose **Accept request.**
60. Choose Accept request.

Note: The *VPC-Peering* connection status changes to Active.

Task 8: Configure network traffic routing across Amazon VPC peering connection

To enable the flow of traffic between the VPCs using private IP addresses, the owner of each VPC in the VPC peering connection must manually add a route to one or more of their VPC route tables that points to the IP address range of the other VPC (the peer VPC).

In this task, you will configure routing to allow traffic across the peering

Edit the route table in VPC-1

61. On the 

menu, click **VPC**.
62. On the left navigation pane, choose **Route Tables**.
63. Select the route table named **VPC-1 PublicRouteTable**.
64. Choose **Routes** tab.

Note: There are two route entries.

- A destination route of 10.0.0.0/24, with Target as local. This is a default entry, it is automatically created during a VPC launch. This entry enables any resource within the VPC to reach each other. Think of a VLAN or a LAN, any resource within a VLAN/LAN should be able to reach each other.
- A destination route of 0.0.0.0/0, with an Internet Gateway (IGW) gateway as Target. This route was added during the previous lab to enable traffic from the public subnet to reach the internet.

65. Choose **Edit routes**.

66. Choose Add route. Then configure:

- Destination- 192.168.0.0/24 (*This is the VPC-2 CIDR which we add here to enable the instance in VPC-1 public subnet to reach the instance in VPC-2 over the VPC peering connection*).
    - Target- select **Peering Connection** from the drop-down list.
    - Choose VPC-Peering

67. Choose **Save changes**

Note: The *VPC-1 PublicRouteTable* should have 3 route entries at this point.

Edit the route table in VPC-2

68. Select the route table named **VPC-2 PublicRouteTable**.

69. Choose **Routes** tab.

Note: There are two route entries.

- A destination route of 192.168.0.0/24, with Target as local. This is a default entry; it is automatically created during a VPC launch. This entry enables any resource within the VPC to reach each other. Think of a VLAN or a LAN, any resource within a VLAN/LAN should be able to reach each other.
- A destination route of 0.0.0.0/0, with an Internet Gateway (IGW) gateway as Target. This route was added during the previous lab to enable traffic from the public subnet to reach the internet.

70. Choose **Edit routes**.

71. Choose Add route. Then configure:

- Destination- 10.0.0.0/24 (*This is the VPC-1 CIDR which we add here to enable the instance in VPC-2 public subnet to reach the instance in VPC-1 over the VPC peering connection*).
    - Target- select **Peering Connection** from the drop-down list.
    - Choose VPC-Peering

72. Choose **Save changes**

Note: The *VPC-2 PublicRouteTable* should have 3 route entries at this point.

Task 9: Test network connectivity across VPC peering connections

This lab uses EC2 Instance Connect to give you access to the EC2 instances through the Amazon EC2 console (browser-based client). Amazon EC2 Instance Connect provides a simple and secure way to connect to your Linux instances using Secure Shell (SSH).

In this task, you will use ICMP ping to test connectivity between the EC2 Instances across the VPC peering connections. The Security Groups for each EC2 have been configured to allow ICMP traffic. Access an EC2 in one VPC, and ping the private IP address of an EC2 in another VPC. Each VPC has two EC2s.

The IP addresses in the examples within this task can be different when compared to your lab EC2 instances IP addresses.

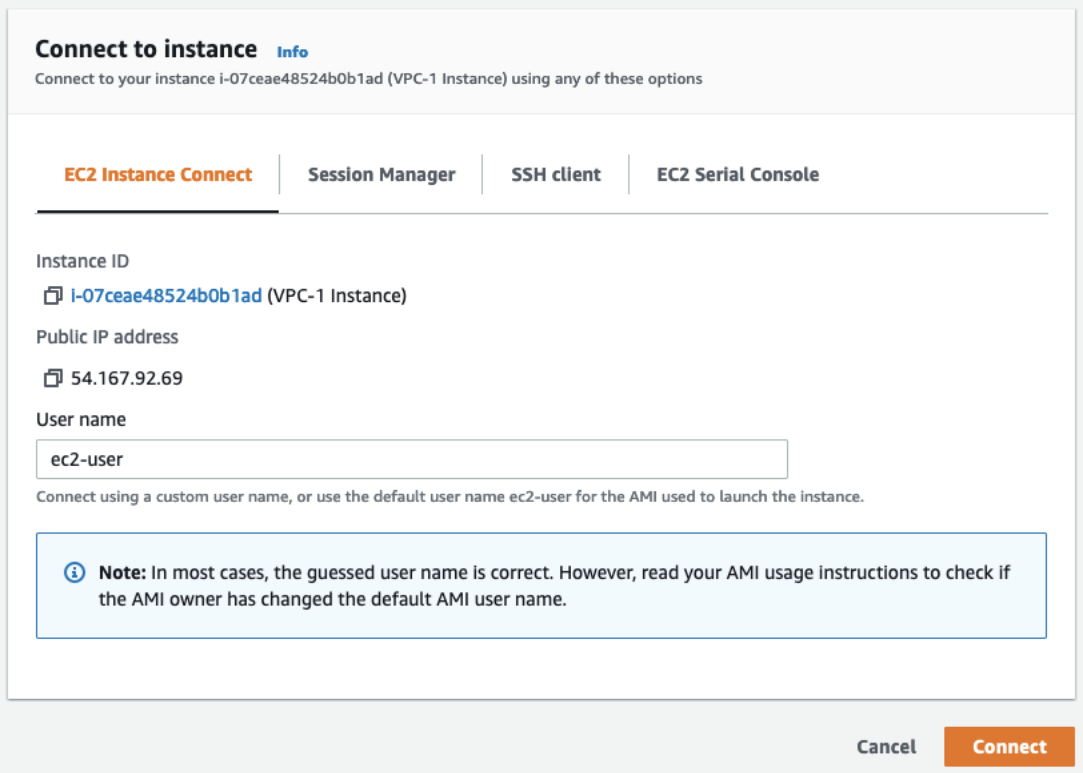
73. In the **us-east-1 (N. Virginia)** Region AWS Management Console, on the **Services** menu, choose **EC2**.
74. On the left navigation pane, choose **Instances**. There are 2 EC2 instances, namely:

    - VPC-1 Instance
    - VPC-2 Instance

75. Select instance named **VPC-2 Instance** and copy the Private IPv4 address of this instance in a text editor.

NOTE: In addition to the Private IPv4 address, this EC2 has a Public IPv4 address. This is because it is launched in public subnet.

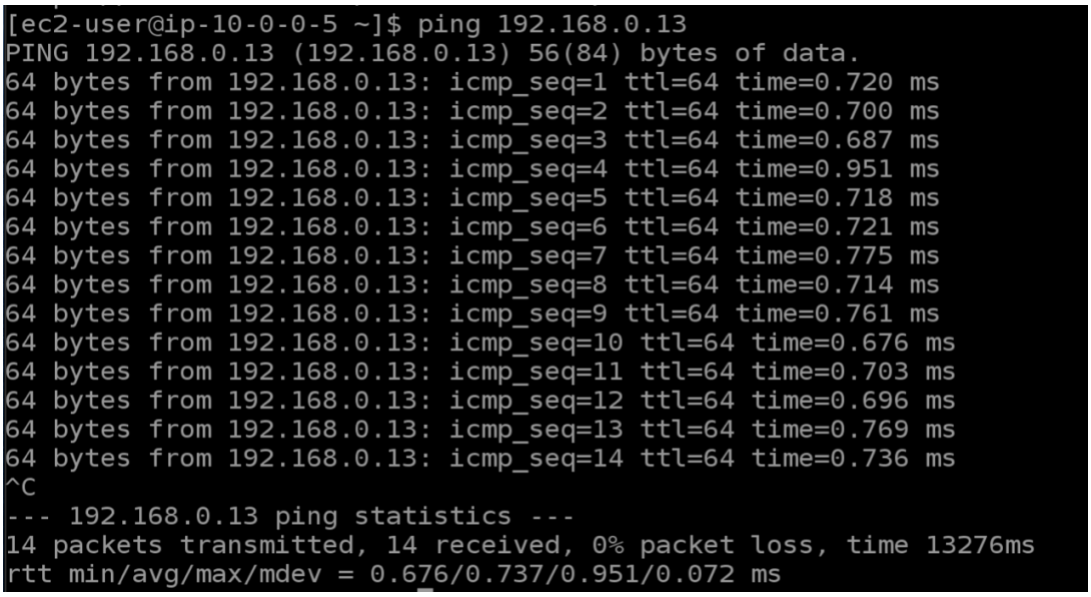
76. Select instance named **VPC-1 Instance**
77. With **EC2 Instance Connect** tab selected, choose **Connect** *(Located near the top section of the navigation panel)*.



Note: A new browser window opens, click **Connect**

78. Choose inside the browser-based shell
79. Test connectivity to *VPC-2 Instance* using ICMP ping.
80. Ping to the Private IPv4 address that you copied in the text editor in Step 73 using the below command

    - ping <Private IPv4 address>
    - To stop the ping command, press *Ctrl+C* on your keyboard.
    - The ICMP ping command should complete, similar to image below



- If they do not complete:
    - Review Task 7 and Task 8 and make sure they are completed. (Verify the peering connection configuration and route table entries)

81. You can repeat the **Steps 73-78** by copying the Private IPv4 address of **VPC-1 Instance** and then connect to **VPC-2 Instance** and ping the Private IPv4 of **VPC-1 Instance**.

Task 10: Clean Up

While there is no additional charge for creating and using an Amazon Virtual Private Cloud (VPC) itself, you can pay for optional VPC capabilities with usage-based charges. Usage charges for other Amazon Web Services, including Amazon EC2, still apply at published rates for those resources, including data transfer charges. To avoid incurring unnecessary charges, use the AWS Management Console to delete the VPCs and other resources that you created while running these labs.

In this task, you will delete all the resources that we created in this lab and the previous lab.

Terminate all Instances

82. In the **us-east-1 (N. Virginia)** Region AWS Management Console, on the **Services** menu, choose **EC2**.
83. On the left navigation pane, choose **Instances**. There are 2 EC2 instances, namely:

- VPC-1 Instance
- VPC-2 Instance

84. Select both the instances.
85. In the **Instance State** menu *(Located near the top section of the navigation panel)*, select **Terminate**.
86. Wait for your web server to terminate. It should display the following:

- **Instance State:** Terminated

You can click the refresh icon to refresh your instances status.

You have now successfully deleted both the instances, **VPC-1 Instance** and **VPC-2 Instance** that were launched in the two VPCs.

Delete VPC-1 and VPC-2

In the previous step, you deleted the instances which were underlying dependencies in the VPCs. Now in this task, you will finally delete the VPC which will also delete its components like Subnets, Route Tables, Security Groups, Network ACLs and Internet Gateways.

87. In the **us-east-1 (N. Virginia)** Region AWS Management Console, on the **Services** menu, choose **VPC**.
88. On the left navigation pane, choose **Your VPCs**. There are 2 VPCs, namely:

- VPC-1
- VPC-2

89. Select both the VPCs.
90. In the **Actions** menu *(Located near the top section of the navigation panel)*, select **Terminate**
91. A dialogue box will pop-up. To confirm deletion, type *delete* in the field.
92. To confirm deletion, type *delete* in the field.

You can click the refresh icon to refresh your VPC console. You should no longer see the two VPCs, **VPC-1** and **VPC-2**

Conclusion

 Congratulations! You now have successfully:

- Create a second Amazon Virtual Private Cloud (VPC)
- Create a public subnet
- Create an Internet gateway
- Create a Route Table and added a route to the Internet
- Configure the security groups of the two instances in both VPCs to allow network traffic across Amazon VPC peering connection
- Launch an instance for testing network connectivity between VPCs
- Create Intra Regional Amazon VPC peering connections
- Configure network traffic routing across Amazon VPC peering connection
- Test network connectivity across VPC peering connections
- Clean up the resources that were created in the 2 labs

APPENDIX

This section will help you understand subnetting in Amazon VPC.

VPC and subnet sizing for IPv4

Subnetting allows you to create multiple logical networks that exist within a single Class A, B, or C network. If you do not subnet, you are only able to use one network from your Class A, B, or C network, which is unrealistic.

Each data link on a network must have a unique network ID, with every node on that link being a member of the same network. If you break a major network (Class A, B, or C) into smaller subnetworks, it allows you to create a network of interconnecting subnetworks. Each data link on this network would then have a unique network/subnetwork ID. Any device, or gateway, that connects n networks/subnetworks has n distinct IP addresses, one for each network / subnetwork that it interconnects.

When you create a VPC, you must specify an IPv4 CIDR block for the VPC. The allowed block size is between a /16 netmask (65,536 IP addresses) and /28 netmask (16 IP addresses).

When you create a VPC, we recommend that you specify a CIDR block from the private IPv4 address ranges as specified in RFC 1918:

RFC 1918 range	Example CIDR block
10.0.0.0 - 10.255.255.255 (10/8 prefix)	Your VPC must be /16 or smaller, for example, 10.0.0.0/16.

172.16.0.0 - 172.31.255.255 (172.16/12 prefix)	Your VPC must be /16 or smaller, for example, 172.31.0.0/16.
192.168.0.0 - 192.168.255.255 (192.168/16 prefix)	Your VPC can be smaller, for example 192.168.0.0/20.

The CIDR block of a subnet can be the same as the CIDR block for the VPC (for a single subnet in the VPC), or a subset of the CIDR block for the VPC (for multiple subnets). The allowed block size is between a /28 netmask and /16 netmask. If you create more than one subnet in a VPC, the CIDR blocks of the subnets cannot overlap.

For example, if you create a VPC with CIDR block 172.16.0.0/24, it supports 256 IP addresses. You can break this CIDR block into multiple smaller subnets, each supporting 16 IP addresses. One subnet uses CIDR block 172.16.0.0/28 (for addresses 172.16.0.0 - 172.16.0.15) and the other uses CIDR block 172.16.0.16/28 (for addresses 172.16.0.16 - 172.16.0.31).

By extending the mask to be 255.255.255.240, you have taken four bits (indicated by "sub") from the original host portion of the address and used them to make subnets.

```
172.16.0.0 -          10101100.00010000.00000000.00000000
255.255.255.240 -     11111111.11111111.11111111.11110000
                      ----------------------------------------|sub|------
```

172.16.0.0	255.255.255.240	host address range 1 to 15
172.16.0.16	255.255.255.240	host address range 16 to 31
172.16.0.32	255.255.255.240	host address range 32 to 47
172.16.0.48	255.255.255.240	host address range 48 to 63

There are tools available on the internet to help you calculate and create IPv4 subnet CIDR blocks. You can find tools that suit your needs by searching for terms such as 'subnet calculator' or 'CIDR calculator'. Your network engineering group can also help you determine the CIDR blocks to specify for your subnets.

The first four IP addresses and the last IP address in each subnet CIDR block are not available for you to use, and cannot be assigned to an instance. For example, in a subnet with CIDR block 172.16.0.0/24, the following five IP addresses are reserved:

- 172.16.0.0: Network address.
- 172.16.0.1: Reserved by AWS for the VPC router.
- 172.16.0.2: Reserved by AWS. The IP address of the DNS server is the base of the VPC network range plus two. For VPCs with multiple CIDR blocks, the IP address of the DNS server is located in the primary CIDR. We also reserve the base of each subnet range plus two for all CIDR blocks in the VPC. For more information, see Amazon DNS server.
- 172.16.0.3: Reserved by AWS for future use.
- 172.16.0.255: Network broadcast address. We do not support broadcast in a VPC, therefore we reserve this address.

Additional Resources

- VPC Peering Basics
- VPC Peering Limitations
- VPC Peering Connections
- VPC Peering Configurations
- Amazon DNS server