

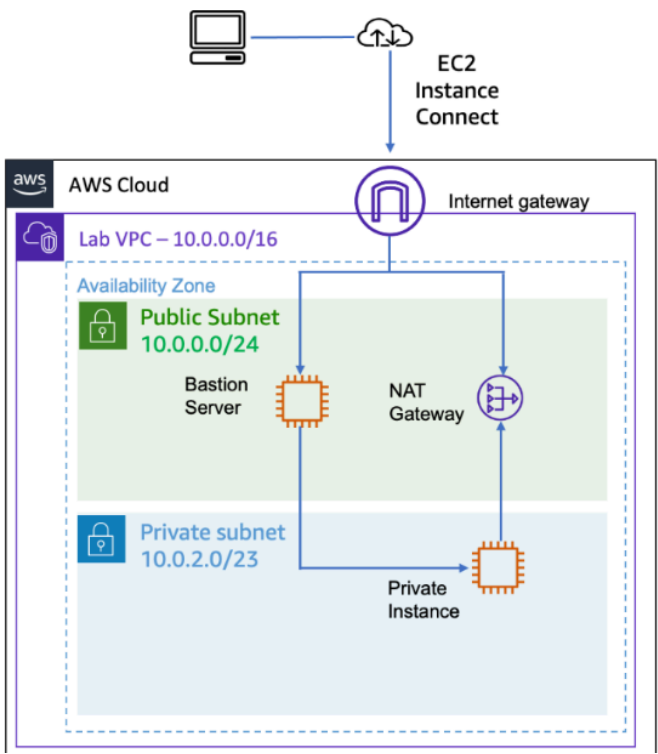
180-[JAWS]-Lab - Configuring an Amazon VPC

Configuring a VPC

Lab overview

Amazon Virtual Private Cloud (Amazon VPC) gives you the ability to provision a logically isolated section of the Amazon Web Services (AWS) Cloud where you can launch AWS resources in a virtual network that you define. You have complete control over your virtual networking environment, including selecting your IP address ranges, creating subnets, and configuring route tables and network gateways.

In this lab, you build a virtual private cloud (VPC) and other network components required to deploy resources, such as an Amazon Elastic Compute Cloud (Amazon EC2) instance.



Public Route Table

Destination	Target
10.0.0.0/16	Local
0.0.0.0/0	Internet Gateway

Private Route Table

Destination	Target
10.0.0.0/16	Local
0.0.0.0/0	NAT Gateway

Objectives

By the end of this lab, you should be able to do the following:

- Create a VPC with a private and public subnet, an internet gateway, and a NAT gateway.
- Configure route tables associated with subnets to local and internet-bound traffic by using an internet gateway and a NAT gateway.
- Launch a bastion server in a public subnet.
- Use a bastion server to log in to an instance in a private subnet.

If you have time, you can complete the optional challenge section in which you create an Amazon EC2 instance in a private subnet and connect to it through the bastion server.

Duration

This lab will require approximately **45 minutes** to complete.

Accessing the AWS Management Console

1. At the top of these instructions, choose **Start Lab** to launch your lab.

A **Start Lab** panel opens displaying the lab status.

2. Wait until the message "Lab status: ready" appears, and then choose **X** to close the **Start Lab** panel.

3. At the top of these instructions, choose **AWS** to open the AWS Management Console on a new browser tab. The system automatically signs you in.

Tip: If a new browser tab does not open, a banner or icon at the top of your browser will indicate that your browser is preventing the site from opening pop-up windows. Choose the banner or icon, and choose **Allow pop-ups**.

4. Arrange the AWS Management Console so that it appears alongside these instructions. Ideally, you should be able to see both browser tabs at the same time to follow the lab steps.

Task 1: Creating a VPC

In this task, you create a new VPC.

5. On the **AWS Management Console**, in the **Search** bar, enter and choose **VPC** to go to the **VPC Management Console**.

6. In the left navigation pane, for **Virtual private cloud**, choose **Your VPCs**.

■ In every Region, a default VPC with a Classless Inter-Domain Routing (CIDR) block of 172.31.0.0/16 has already been created for you. Even if you haven't created anything in your account yet, you will see some pre-existing VPC resources already there.

7. Choose **Create VPC** and configure the following options:

- **Resources to create:** Choose **VPC only**.
- **Name tag:** Enter **Lab VPC**.
- **IPv4 CIDR block:** Choose **IPv4 CIDR manual input**.
- **IPv4 CIDR:** Enter **10.0.0.0/16**.
- **IPv6 CIDR block:** Choose **No IPv6 CIDR block**.
- **Tenancy:** Choose **Default**.
- **Tags:** Leave the suggested tags as is.

8. Choose **Create VPC**.

At the top of the page, a message displays similar to the following: "You successfully created vpc-NNNNNNNNNNNN / Lab VPC."

9. Choose **Actions**, and choose **Edit VPC settings**.

10. In the **DNS settings** section, select **Enable DNS hostnames**.

11. Choose **Save**.

EC2 instances launched into the VPC now automatically receive a public IPv4 Domain Name System (DNS) hostname.

Task 2: Creating subnets

In this task, you create a public subnet and a private subnet.

Task 2.1: Creating a public subnet

12. In the left navigation pane, for **Virtual private cloud**, choose **Subnets**.

13. Choose **Create subnet** and configure the following options:

- **VPC ID:** Choose **Lab VPC**.
- **Subnet name:** Enter `Public Subnet`.
- **Availability Zone:** Choose the first Availability Zone in the list. Do not choose **No preference**.
- **IPv4 CIDR block:** Enter `10.0.0.0/24`.

14. Choose **Create subnet**.

You now configure the public subnet to automatically assign a public IP address for all EC2 instances that are launched within it.

15. Select **Public Subnet**.

16. Choose **Actions**, and then choose **Edit subnet settings**.

17. In the **Auto-assign IP settings** section, select **Enable auto-assign public IPv4 address**.

18. Choose **Save**.

● Even though this subnet has been named **Public Subnet**, it is not yet public. A public subnet must have an internet gateway, which you attach in a task later in the lab.

Task 2.2: Creating a private subnet

In this task, you create the private subnet, which is used for resources that are to remain isolated from the internet.

19. To create the private subnet, repeat the steps from the previous task, and choose the following options:

- **VPC ID:** Choose **Lab VPC**.
- **Subnet name:** Enter `Private Subnet`.
- **Availability Zone:** Choose the first Availability Zone on the list. Do not choose **No preference**.
- **IPv4 CIDR block:** Enter `10.0.2.0/23`.

20. Choose **Create subnet**.

The CIDR block of 10.0.2.0/23 includes all IP addresses that start with 10.0.2.x and 10.0.3.x. This range is twice as large as the public subnet because most resources should be kept in private subnets unless they specifically need to be accessible from the internet.

Your VPC now has two subnets. However, the VPC is totally isolated and cannot communicate with resources outside the VPC.

Next, you configure the public subnet to connect to the internet through an internet gateway.

Task 3: Creating an internet gateway

In this task, you create an internet gateway for your VPC. You need an internet gateway to establish outside connectivity to EC2 instances in VPCs.

21. In the left navigation pane, for **Virtual private cloud**, choose **Internet gateways**.

22. Choose **Create internet gateway**, and then for **Name tag**, enter `Lab IGW`.

23. Choose **Create internet gateway**.

24. Choose **Actions**, then choose **Attach to a VPC**.

Your public subnet now has a connection to the internet. However, to route traffic to the internet, you must also configure the public subnet's route table so that it uses the internet gateway.

Task 4: Configuring route tables

In this task, you do the following:

- Create a public route table for internet-bound traffic.
- Add a route to the route table to direct internet-bound traffic to the internet gateway.
- Associate the public subnet with the new route table.

25. In the left navigation pane, for **Virtual private cloud**, choose **Route tables**.

Several route tables are listed.

26. Select the route table that includes **Lab VPC** in the **VPC** column.

Tip: If you cannot see the VPC column, scroll to the right.

27. In the **Name** column, choose the edit icon, enter `Private Route Table` for **Edit Name**, and then choose **Save**.

28. Choose the **Routes** tab.

There is currently only one route. It shows that all traffic destined for 10.0.0.0/16 (which is the range of the Lab VPC) will be routed locally. This option allows all subnets within a VPC to communicate with each other.

You now create a new public route table to send public traffic to the internet gateway.

29. Choose **Create route table** and configure the following options:

- **Name - optional:** Enter `Public Route Table`.
- **VPC:** Choose **Lab VPC**.

30. Choose **Create route table**.

31. After the route table is created, in the **Routes** tab, choose **Edit routes**.

Note: You now add a route to direct internet-bound traffic (0.0.0.0/0) to the internet gateway.

32. Choose **Add route** and then configure the following options:

- **Destination:** Enter `0.0.0.0/0`.
- **Target:** Choose **Internet Gateway**, and then choose **Lab IGW** from the list.

33. Choose **Save changes**.

The final step is to associate this new route table with the public subnet.

34. Choose the **Subnet associations** tab.

35. Choose **Edit subnet associations**.

36. Select **Public Subnet**.

37. Choose **Save associations**.

The public subnet is now public because it has a route table entry that sends traffic to the internet through the internet gateway.

In the previous tasks, you created a VPC and attached an internet gateway. Then you created subnets and a route table and associated a public route table to the public subnet. You now launch resources in the subnets as required.

Task 5: Launching a bastion server in the public subnet


A bastion server (also known as a jump box) is an EC2 instance in a public subnet that is securely configured to provide access to resources in a private subnet. Systems operators can connect to the bastion server and then jump into resources in the private subnet.

In this task, you launch an EC2 instance bastion server in the public subnet that you created earlier.

38. On the AWS Management Console, in the **Search** bar, enter and choose `EC2` to go to the **EC2 Management Console**.



39. In the left navigation pane, choose **Instances**.

40. Choose **Launch instances** and configure the following options:

- In the **Name and tags** section, enter `Bastion Server`.
- In the **Application and OS Images (Amazon Machine Image)** section, configure the following options:
 - **Quick Start:** Choose **Amazon Linux**.
 - **Amazon Machine Image (AMI):** Choose **Amazon Linux 2023 AMI**.
- In the **Instance type** section, choose **t3.micro**.
- In the **Key pair (login)** section, choose **Proceed without a key pair (Not recommended)**.
 You use EC2 Instance Connect to access the shell running on the EC2 instance, so a key pair is not needed in the lab.

41. In the **Network settings** section, choose **Edit** and configure the following options:

- **VPC - required:** Choose **Lab VPC**.
- **Subnet:** Choose **Public Subnet**.
- **Auto-assign public IP:** Choose **Enable**.
- **Firewall (security groups):** Choose **Create security group**.

- **Security group name - required:** Enter `Bastion Security Group`.
 - **Description - required:** Enter `Allow SSH`.
 - **Inbound security groups rules:**
 - **Type:** Choose `ssh`.
 - **Source type:** Choose `Anywhere`.
42. Choose **Launch instance**.
43. To display the launched instance, choose **View all instances**.
-  The EC2 instance named **Bastion Server** is initially in a *Pending* state. The **Instance state** then changes to  *Running* to indicate that the instance has finished booting.
- The bastion server will be launched in the public subnet.
- Continue to the next task. You do not need to wait for the instance to be running.

Task 6: Creating a NAT gateway

In this task, you launch a NAT gateway in the public subnet and configure the private route table to facilitate communication between resources in the private subnet and the internet.

44. On the AWS Management Console, in the **Search** bar, enter `NAT gateways`, choose the **Features** list, and choose **NAT gateways**.
45. Choose **Create NAT gateway** and configure the following options:
- **Name:** Enter `Lab NAT gateway`.
 - **Subnet:** From the dropdown list, choose **Public Subnet**.
46. Choose `Allocate Elastic IP`.
47. Choose **Create a NAT gateway**.
- You now configure the private subnet to send internet-bound traffic to the NAT gateway.
48. In the left navigation pane, choose **Route tables**, and then select **Private Route Table**.
49. Choose the **Routes** tab.
- The route table is currently showing only a single entry, which routes traffic locally within the VPC. You add an additional route to send internet-bound traffic through the NAT gateway.
50. Choose `Edit routes`.
51. Choose `Add route` and configure the following options:
- **Destination:** Enter `0.0.0.0/0`.
 - **Target:** Choose **NAT Gateway**, and then choose **nat-** from the list.

53. Choose **Save changes**.

Resources in the private subnet that wish to communicate with the internet now have their network traffic directed to the NAT gateway, which forwards the request to the internet. Responses flow through the NAT gateway back to the private subnet.

Optional challenge: Testing the private subnet

💡 This challenge is optional and is provided in case you have lab time remaining.

In this optional challenge, you launch an EC2 instance in the private subnet and confirm that it can communicate with the internet.

Launching an instance in the private subnet

In this optional task, you launch an EC2 instance in the private subnet.

54. Follow the instructions that you used to launch the bastion server, and configure the following options:

- In the **Name and tags** section, enter `Private Instance`.
- In the **Application and OS Images (Amazon Machine Image)** section, configure the following options:
 - **Quick Start:** Choose **Amazon Linux**.
 - **Amazon Machine Image (AMI):** Choose **Amazon Linux 2023 AMI**.
- In the **Instance type** section, choose **t3.micro**.
- In the **Key pair (login)** section, choose **Proceed without a key pair (Not recommended)**.
- In the **Network settings** section, choose `Edit` and configure the following options:
 - **VPC - required:** Choose **Lab VPC**.
 - **Subnet:** Choose **Private Subnet** (not the public subnet).
 - **Firewall (security groups):** Choose **Create security group**.
 - **Security group name - required:** Enter `Private Instance SG`.
 - **Description - required:** Enter `Allow SSH from Bastion`.
- **Inbound security groups rules:**
 - **Type:** Choose **ssh**.
 - **Source type:** Choose **Custom**.
 - **Source:** Choose **10.0.0.0/16**.

55. Expand the **Advanced Details** section, and for **User data - optional**, paste the following script:

```
#!/bin/bash
# Turn on password authentication for lab challenge
echo 'lab-password' | passwd ec2-user --stdin
sed -i 's|[*]PasswordAuthentication no|PasswordAuthentication yes|g' /etc/ssh/sshd_config
systemctl restart sshd.service
```

💡 This script permit login by using a password. It is included to help make the lab steps shorter but is not recommended for normal instance deployments.

56. Choose **Launch instance**.

57. To display the launched instance, choose **View all instances**.

Logging in to the bastion server

The instance that you just launched is in the private subnet, so it is not possible to directly log in to the instance. Instead, you first log in to the bastion server in the public subnet and then log in to the private instance from the bastion server.

58. On the **AWS Management Console**, in the **Search** bar, enter and choose **EC2** to open the **EC2 Management Console**.

59. In the navigation pane, choose **Instances**.

60. From the list of instances, select the **Bastion Server** instance.

61. Choose **Connect**.

62. On the **EC2 Instance Connect** tab, choose **Connect**.

Note: If you prefer to use an SSH client to connect to the EC2 instance, see the guidance to [Connect to Your Linux Instance](#).

Logging in to the private instance

You should now be logged in to the bastion server, which is located in the public subnet. Keep this terminal window open for use later.

You now connect to the private instance, which is placed in the private subnet.

63. In the Amazon EC2 console, choose **Instances**, and select **Private Instance** (and clear any other instances).

64. Copy the **Private IPv4 addresses** (shown in the lower half of the page) to your clipboard.

This IP address is a private IP address starting with 10.0.2.x or 10.0.3.x. This address is not reachable directly from the internet, which is why you first logged in to the bastion server. You now log in to the private instance.

65. Return to the terminal window, and run the following command. In the command, replace *PRIVATE-IP* with the IP address that you just copied to your clipboard:

```
ssh PRIVATE-IP
```

The command that you run should look similar to the following: `ssh 10.0.2.123`

66. If you are prompted with the message "Are you sure you want to continue connecting", enter **yes**

67. When prompted for a password, enter **tab-password**.

You should now be connected to the private instance. You accomplished this task by first connecting to the bastion server (in the public subnet) and then connecting to the private instance (in the private subnet).

Testing the NAT gateway

The final part of this challenge is to confirm that the private instance can access the internet.

You do this by running the ping command.

68. Run the following command:

```
ping -c 3 amazon.com
```

You should see results similar to the following:

```
PING amazon.com (176.32.98.166) 56(84) bytes of data.  
64 bytes from 176.32.98.166 (176.32.98.166): icmp_seq=1 ttl=222 time=79.2 ms  
64 bytes from 176.32.98.166 (176.32.98.166): icmp_seq=2 ttl=222 time=79.2 ms  
64 bytes from 176.32.98.166 (176.32.98.166): icmp_seq=3 ttl=222 time=79.0 ms
```

This output indicates that the private instance successfully communicated with amazon.com on the internet.

The private instance is in the private subnet, and the only way that this is possible in the current scenario is by going through the NAT gateway.

This output confirms that your network configuration was successful.

Conclusion

Congratulations! You now have successfully done the following:

- Created a VPC with a private and public subnet, an internet gateway, and a NAT gateway
- Configured route tables associated with subnets to local and internet-bound traffic by using an internet gateway and a NAT gateway
- Launched a bastion server in a public subnet
- Used a bastion server to log in to an instance in a private subnet

Lab complete

Congratulations! You have completed the lab.

70. At the top of this page, choose **End Lab** and then choose **Yes** to confirm that you want to end the lab.

A panel appears indicating that "You may close this message box now. Lab resources are terminating."

71. To close the **End Lab** panel, choose the **X** in the upper-right corner.