Linux Academy
Hands-on Lab

VPC Peering Lab

Contents

Lab Connection Information

- Labs may take up to five minutes to build
- Access to an AWS Console is provided on the Hands-on Lab page, along with your login credentials
- Ensure you are using the N. Virginia region
- Labs will automatically end once the alloted amount of time finishes

Related Courses

AWS Certified Advanced Networking Specialty

Related Videos

VPC Refresh

VPC Peering

Need Help?

Linux Academy Community

... and you can always send in a support ticket on our website to talk to an instructor!

In this lab, we use three different VPCs to demonstration VPC pairing and how AWS handles certain routing scenarios.

Lab Setup

Log in to the AWS Console using the credentials provided on the Hands-on Lab page.

Navigate to the **EC2 Dashboard**. We have two running instances. Switch to the **VPC Dashboard** to see our multiple different VPCs – Site A, B, and C. We want to peer all of these sites, with Site B as the headquarters, able to access Site B and Site C. Note that Site A and C cannot be peered together because their CIDR ranges overlap, both using 10.111.0.0/16.

Click **Peering Connections** on the left menu. **Create Peering Connection**. Give the connection a **name tag** of *B-A*. Set the **VPC (Requester)** to *Site B*. Leave **Account** set to *My account*, and select *Site A* for the **VPC (Accepter)**. Press **Create Peering Connection**.

Press **Create Peering Connection** again, this time setting the **name tag** to *B-C*, with a **VPC (Requester)** of *Site B* and a **VPC (Accepter)** of *Site C*. **Create Peering Connection**.

Now, on the **VPC Peering** page, right click on each connection and press **Accept Request**, **Yes, Accept**.

Configuring the Site B Routes

We want to ensure our route tables are configured properly before continuing. From the **EC2 Dashboard**, view the **Instances** and select the private instance. Note the private IP in the 10.111.1.0 range.

Return to the **VPC Dashboard** and view the **Route Tables**. Select **Site A** and click on the **Routes** tab. We have a route to 10.111.0.0. Click on the **Subnet Associations** tab and confirm the 10.111.1.0 subnet is listed. For **Site B**, the route is 10.222.0.0 and the subnet is 10.222.1.0. For **Site C**, the route is 10.111.0.0 and the subnet is 10.111.1.0. As we can see, Site A and C have the same routing and subnet information.

Since we can't change the IP address, we want to instead see what subnet and IP address that the instance uses. From the **EC2 Dashboard** check the VPC the private instance is in, then compare it to the VPC information in our **VPC Dashboard**. The instance is in Site C, so we want to use our B-C connection to work with the instance.

Select Site B, and click the **Routes** tab. Press **Edit**, **Add another route**. Set the **Destination** to the private IP of the instance with a /32 and the **Target** to the *B-C* route. *Add another route* with a Destination of *10.111.1.0/24* and a Target of *B-A*. AWS will take the highest mask and route to it first. **Save**.

Configuring the Site A and C Routes

Now, we want to log in to our public instance. From the **EC2 Dashboard**, copy the IP address on the instance with public routing, and SSH into it using your workstation terminal:

```
[user@workstation] ssh linuxacademy@PUBLICIP
```

The password is *123456*.

Try to ping the private instance:

```
[linuxacademy@ip] ping PRIVATEIP
```

This fails because our routing is only one way. We need to have routes going in both directions for our VPCs to properly peer.

Go back to the **VPC Dashboard**. Select Site C and press the **Routes** tab. Click **Edit**, **Add another route**, then add route with the **Destination** of *10.222.0.0/16* and a **Target** of *B-C*. **Save**. Select Site A and follow the same instructions, this time adding a route with the **Destination** of *10.222.0.0/16* and a **Target** of *B-A*.

Return to the terminal. Try to ping the IP address once more:

```
[linuxacademy@ip] ping PRIVATEIP
```

It still does not work. Now, we want to confirm our security settings. Go to **Network ACLs** on the left menu of the VPC Dashboard. Check the inbound and outbound rules of all sites. All three should allow all, both ways, so this is not the issue.

Go to the **Security Groups** page. View the **Inbound Rules** for the Site A security group. Currently, it is only allowing connections with within its own VPC. Press **Edit**, **Add another rule**, and set an *All Traffic* rule type with a **Source** of *10.222.0.0/16*. **Save**. Do the same for Site C.

Select Site B. Click **Inbound Rules**, **Edit**, then **Add another rule**, this time allowing *All Traffic* from the *10.111.0.0/16* **Source**. **Save**.

Once more, return to the terminal and try to ping the private instance:

```
[linuxacademy@ip] ping PRIVATEIP
PING 10.111.1.107 (10.111.1.107) 56(84) bytes of data.
64 bytes from 10.111.1.107: icmp_seq=1 ttl=255 time=0.640 ms
64 bytes from 10.111.1.107: icmp_seq=2 ttl=255 time=0.687 ms
```

Success! This lab is now complete.