Linux Academy

Hands-on Lab

Challenge Lab: Troubleshooting Connectivity Issues #2

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 Solutions Architect - Associate

Related Videos

VPC Troubleshooting Scenarios

Need Help?

Linux Academy Community

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

For this challenge lab, we provide you with a broken environment that is in need of fixing. Attempt to troubleshoot the environment on your own, after reading the parameters in the scenario, then compare your answer with the solution.

The Challenge Scenario

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

Take this time to review the AWS architecture. An EC2 instance, called *Web Server*, needs to have its packages updated on the command line using `yum`. You must fix the AWS environment's connectivity issues to allow for this.

The Web Server instance has been provisioned in a private subnet and placed behind a Bastion Host and NAT gateway.

To access the Bastion Host to test its connectivity, SSH into the server using the **Access Credentials** provided on the Hands-on Lab page:

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

From inside the Bastion Host, once the AWS environment has been troubleshot, we can then use the web server **Access Credentials**:

```
[linuxacademy@bastion] ssh linuxacademy@WEBPRIVATEIP
```

And update the server:

```
[linuxacademy@webserver] sudo yum update
```

The Solution

Before we focus on the issues of the environment, we need to understand how the environment is intended to be set up. The Bastion Host and NAT gateway are in a public subnet, while the Web Server instance is in a private subnet without any public IP address.

To update the Web Server, we need to access it through the Bastion Host, while the server itself needs access to the NAT gateway to download updates.

Navigate to the **EC2 Dashboard** and view both servers, the Web Server and Bastion Host. They are in different subnets and Availability Zones. Make note of the subnet IDs.

Now, in another tab, open the **VPC Dashboard** and go the **Subnets** page. There are four listed subnets, two of which are associated with our EC2 instances. Select the private subnet, and view the **Route Table**; it has no routes to anything except internal routes. View the public subnet; this has a route to the NAT gateway.

From your terminal, check to see if we can log in to the Bastion Host:

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

This fails, indicating that there is an issue with the environment. We will want to check the internet gateway, route tables, NACL, and security groups.

Return to the **VPC Dashboard** on the AWS Console. Click **Internet Gateway**. We do have an internet gateway, and it is attached to the VPC. Go to **Route Tables**, select the route table associated with the public subnet, and confirm that the internet gateway is listed under **Routes**.

Go to the **EC2 Dashboard** and select the Web Server instance; view the associated security group. Here we can see the issue: Access is only allowed over port 80, but SSH uses port 22. Click on the security group to access it, then press the **Inbound** tab. **Edit** and click **Add Rule** to add a rule with the **Type** *SSH* with a CIDR of *0.0.0.0/0*. **Save**.

Return to the terminal and try to connect:

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

This still fails. The only thing left to check is the NACL.

From the **VPC Dashboard**, click **Network ACLs**. Select the NACL associated with our subnets, then go to the **Inbound Rules** tab. All traffic from all ranges is allowed. Now check the **Outbound Rules**; here, we can see everything is denied. Press **Edit**, **Add Rule**. Set the **Rule #** to *100*, the **Type** to *ALL Traffic*, and the **Destination** to *0.0.0.0/0*. **Save**.

Once more, try to log in from the terminal:

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

We can successfully connect!

Now, try to log in to the private instance:

```
[linuxacademy@bastion] ssh linuxacademy@WEBPRIVATEIP
```

Since this is successfully, try to update the packages:

```
[linuxacademy@webserver] sudo yum update
```

This fails, indicating an issue with the communication between the private instance the NAT gateway. We want to check that the NAT gateway is there, and that there is a route to the subnet to the NAT gateway.

Return to the **VPC Dashboard** and click **NAT Gateways**. As we can see, the gateway is there and has a public IP assigned to it. However, we still need see if it is associated with the correct route table. Make of the associated subnet, then go to the **Subnets** page. Select the associated subnet, and click **Route Tables**. This subnet does have a route to the internet gateway. We than want to confirm the associate between the public subnet 1 we just looked at and subnet 3. Select **Subnet 3** and view the **Route Table**; this has no routes.

Switch to the **Route Tables** page and select the route table associated with subnet 3. Click **Routes**, **Edit**, then **Add another route**. Set the **Destination** to *0.0.0.0/0* and the **Target** to the NAT gateway. **Save**.

Return to the terminal and try to update the server:

```
[linuxacademy@webserver] sudo yum update
```

Success! This lab is now complete.