The environment and the first steps

Hello and welcome back to the Terraform and Ansible for AWS course on Linux Academy! In this lesson, we’re going to take a look at our infrastructure after it has been fully deployed so you can get an idea of how we will plan our script. You may think that you typically won’t have this luxury, but there are plenty of sample environments out there that can help you get started.

This is my AWS console after a fresh Terraform Apply

1. IAM: First up, let’s check out the IAM role that needs to be created for the EC2 instances to access the S3 buckets. As you can see, s3\_access was created for this purpose.

Here you can see the policy that was created which is an allow all. Luckily, our primary instances will be in a private subnet, so this ability isn’t as concerning as it could be if you had an instance open to the world.

1. Next Up, we’ve got the VPC that we created. This is a custom VPC that we will create from scratch.

As you can see, there are several subnets that all have their specific purpose. This allows utmost granularity and security control for our environment.

Here we have our route tables. There is a public and a private. The private is primarily for the RDS instances and the public is for the rest.

Here is our Internet Gateway which is attached to the public route table. This will allow our public route table to access the internet. If any of this is confusing to you, I strongly suggest you check out the AWS courses on Linux Academy before you proceed.

Finally, we have the Security groups. These are also pretty self-explanatory. The thing to take note of here is the sg\_public, which only allows SSH access from my (your) IP address, which we took note of in a previous lesson. The sg\_private and sg\_rds security groups both only allow traffic from the VPC.

1. Now we move on to the EC2 section. There has been a good bit of activity here.

Under instances, you can see we have 2 autoscaling group instances and one dev instance. You will notice the asg-instances do not have public IP addresses. This is by design as the Load Balancer handles the public traffic from them.

Under AMIs, you can see we have an AMI with a random name here, this is to ensure we don’t have a collision with an existing AMI.

Under “key Pairs”, you can see a key named “my\_key” has been created. This is a custom key that was made using the SSH key created in an earlier lesson.

Here is the load balancer. As you can see under instances, both instances are in separate Availability Zones and are currently InService serving traffic.

Here is our Health Check settings, which are set very low for this example to ensure the servers are in service quickly, but this is not necessarily best practice.

The listener is on port 80. I urge you to explore attaching an SSL certificate to this and protecting your site appropriately, but this is outside the objectives for this course.

Next up, we have launch configuration. This launch configuration has a custom ID set by Terraform in order to assure we have no collisions. You can see here that it deploys instsances with the proper key and IAM instance profile. Also, you can see that userdata is set to send a cron job that automatically retrieves data from the code bucket every 5 minutes to keep them updated with changes propagated from the dev server using an Ansible playbook.

Here is our Autoscaling group which also has a custom name. As you can see, it is attached to the launch configuration we created and its instances are attached to the loadbalancer we created. You can see the availability zones, the autoscaling policies, and everything else you would normally see in an Auto Scaling Group

1. Finally, we have the Route53 records! As you can see, The NS records use the designated name servers we retrieved from an earlier lesson, there is a CNAME pointing to the database, and an A record for the dev server and the load balancer.

Great! Now that we have a rough idea of what Terraform and Ansible can do, I hope you are as excited as I am to open up your favorite text editor and start coding!