DB instance

Hello! And welcome back to the Terraform and Ansible for AWS course on Linux Academy! My name is Derek, and in this lesson, we’re going to start deploying our database and compute resources.

If you notice, we haven’t done anything with S3 yet. I’ll be saving that for last as it involves several sections of our script, so it’s best to just do it later.

First up, we need to get our Database deployed. So let’s add it!

resource "aws\_db\_instance" "db" {

allocated\_storage = 10

engine = "mysql"

engine\_version = "5.6.27"

instance\_class = "${var.db\_instance\_class}"

name = "${var.dbname}"

username = "${var.dbuser}"

password = "${var.dbpass}"

db\_subnet\_group\_name = "${aws\_db\_subnet\_group.rds\_subnetgroup.name}" vpc\_security\_group\_ids = ["${aws\_security\_group.RDS.id}"]

}

Ok, now that we’ve done, that, we need to make sure we add these variables to our variables.tf and terraform.tfvars files. Again, if you wish to have terraform ask you to enter the variables with each run, you can just add them to your variables.tf file followed by squiggly brackets. Otherwise, add them to both your variables.tf and terraform.tfvars files with the values you wish to use.

Here is my variables.tf file currently. As you can see, I have my localip and database fields.

This is my terraform.tfvars file, as you can see, everything is filled out as well. Of course, these are just examples and a production environment should have MUCH more secure passwords.

Ok, now that we have created the database, let’s create our key pair to use for our instances.

Resource “aws\_key\_pair” “auth” {

Key\_name = “${var.key\_name}”

Public\_key = “${file(var.public\_key\_path)}”

}

Ok, let’s talk about this. First, we created the aws key pair resource and gave it an ID of “auth”. Then we gave it a name with a variable, that we need to add to our variables.tf file. I’ll name mine “kryptonite”, you can use whatever you like.

Now, we need to specify our public key we created earlier. If you recall from the previoius lesson, we created a key in home, .ssh, called kryptonite. I will enter this path as the variable here.

I will specify this key location in the terraform.tf file here:

So, what this is doing is importing the contents of the public key file, uploading to amazon, and creating a new key based on this information. Now, remember, this will NOT upload the private key to your instances, only the public! So, if you need to connect to one of your private instances from your public instance as a “bastion host”, you will need to use SSH –A to forward the key agent or you will need to copy the private key to your host. I don’t ever recommend storing your private key on a host, so your first option is your most secure.