Deployment 4

Brayan Molina

October 28th, 2022

Table of Contents

Objectives:

This deployment will build on the information used in previous deployments of the flask application, Url-shortner. The objective of this deployment will be the following:

- Demonstrate ability to deploy an application into a VPC using IaC.

Documentation:

Jenkins Server Configuration and Installation:

- EC2 instance on AWS configured on default VPC with the following:
  - Port 22 – SSH
  - Port 80 – HTTP
  - Port 8080 – Jenkins
  - Jenkins script used for default installation
- Terraform installed on Jenkins server following commands provided by HashiCorp.
- Details for user configured on Amazon's IAM will be provided to Jenkins through Jenkins credentials manager. This will allow Jenkins to recognize the AWS access key and secret key needed to deploy our application using Terraform.

Pipeline Build in Jenkins:

Jenkinsfile:

- Build and Test stages are configuring virtual environment and running flask application, as well as testing function found inside the application source code.
- Credentials provided to Jenkins will be stored as environment variables active within the scope of the step.
- Dir step will be used to change to Terraform directory containing configuration for our resources.
- Terraform will be used to provision an EC2 instance that will deploy our application using Gunicorn.
- Terraform will be needed for the Init, Plan, Apply, and Destroy stages in the Jenkinsfile.

Modification to pipeline:

- The following lines were added to the build stage of the Jenkinsfile:

  ```
  python3 -m pip install mypy
  mypy –show-error-codes application.py >> test3/bin/linterrors.txt
  ```

- The output from mypy will be placed in a file called linterrors.txt; path of file could be changed if needed.

Terraform Deployment:

- To deploy the application, configuration of the following resources is needed:
  - VPC – VPC with a single availability zone.
  - Subnet – Public subnet to create EC2 instance serving application.
  - Internet Gateway – To receive and deliver traffic outside of the VPC.
  - Route Table – Routes traffic to the correct resource
  - Instance – EC2 instance serving url-shortner
  - Configuration of availability zone that subnets will be placed in

```
variable "aws_access_key" {}
variable "aws_secret_key" {}

provider "aws" {
  access_key = var.aws_access_key
  secret_key = var.aws_secret_key
  region = "us-east-1"

}

resource "aws_vpc" "test-vpc" {
  cidr_block           = "172.28.0.0/16"
  enable_dns_hostnames = "true"

  tags = {
    "Name" : "Deployment4VPC"
  }
}

resource "aws_subnet" "public1" {
  cidr_block = "172.28.0.0/18"
  vpc_id                  = aws_vpc.test-vpc.id
  map_public_ip_on_launch = "true"
  availability_zone       = data.aws_availability_zones.available.names[0]
}


resource "aws_internet_gateway" "gw_1" {
  vpc_id = aws_vpc.test-vpc.id
}

resource "aws_route_table" "route_table1" {
  vpc_id = aws_vpc.test-vpc.id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.gw_1.id
  }
}

resource "aws_route_table_association" "route-subnet1" {
  subnet_id      = aws_subnet.public1.id
  route_table_id = aws_route_table.route_table1.id
}


resource "aws_instance" "web_server01" {
  ami = "ami-08c40ec9ead489470"
  instance_type = "t2.micro"
  key_name = "ssh1"
  vpc_security_group_ids = [aws_security_group.web_ssh.id]
  subnet_id = aws_subnet.public1.id

  user_data = "${file("deploy.sh")}"

  tags = {
    "Name" : "Webserver001"
  }

}

data "aws_availability_zones" "available" {
  state = "available"
}


output "instance_ip" {
  value = aws_instance.web_server01.public_ip

}
```

Key Takeaways:

- Run the terraform file inside the workspace of the Jenkins server.
- The user_data block contains the deployment commands needed to host our application on Gunicorn. Gunicorn will host application on default port 8000.

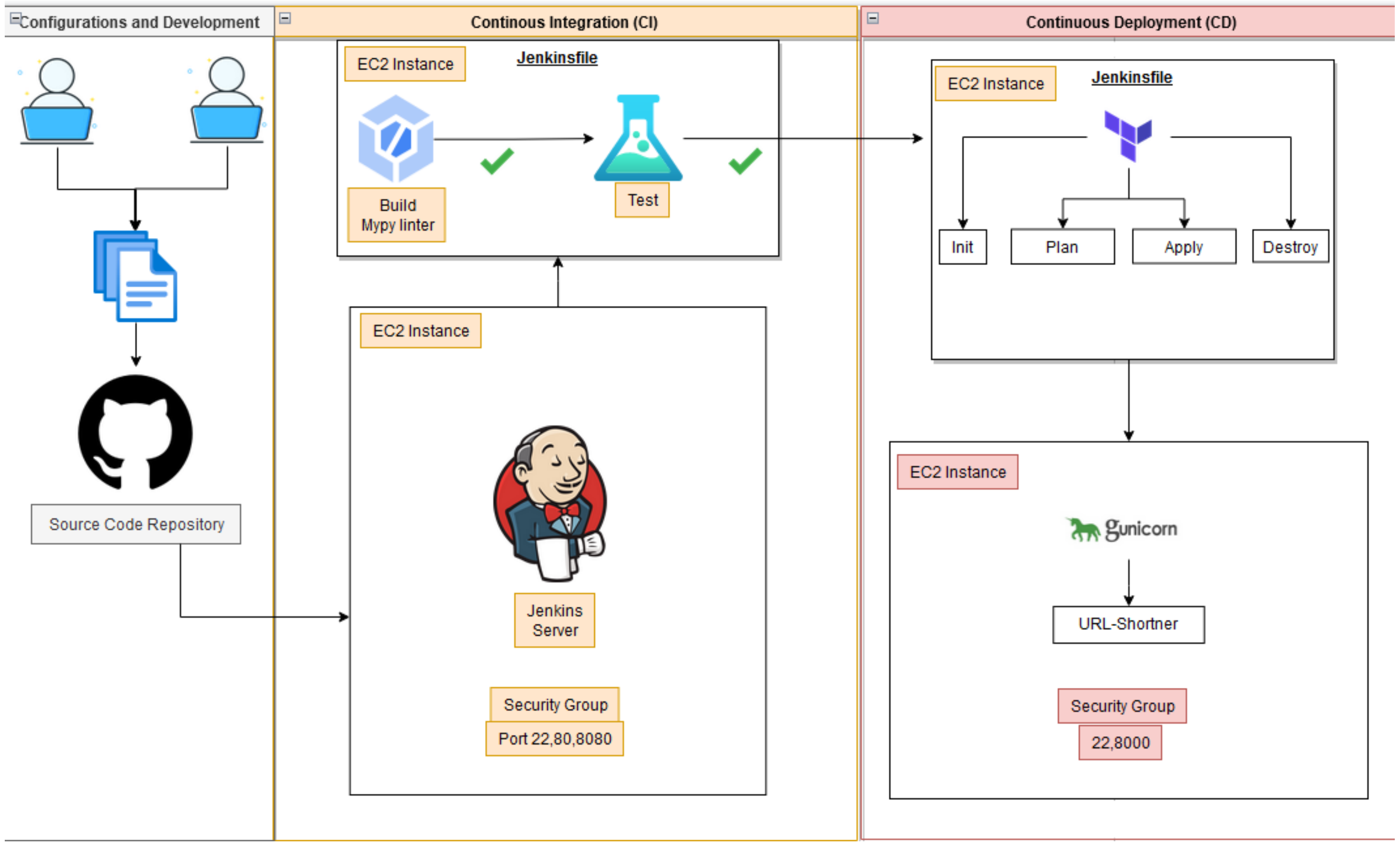
Observations and Errors:

Error	*Cause*	*Solution*
The virtual environment was not created successfully because ensurepip is not available. On Debian/Ubuntu systems, you need to install the python3-venv package using the following command.	Virtual Environment command not installed in the Jenkins server	Run apt install python3.10-venv to install venv into your server.
configuring Terraform AWS Provider: error validating provider credentials	The request signature we calculated does not match the signature you provided.	Correctly input the AWS user credentials. Excess spaces at the end could lead to this error.
creating EC2 Instance: InvalidKeyPair.NotFound: The key pair 'Cali' does not exist	Cali Key does not exist for the AWS user	Correctly input the name of your public key generated on your AWS account. This will be specific to your account.
data resource "aws_availability_zones" "available" has not been declared	Declare availability zone in Terraform	Include data related to Availability zone subnets will be in.
main.tf line 28: There is no closing brace for this block before the end of the file	Braces are incorrectly paired in the Jenkinsfile	Ensure that the number of braces are correct in the Jenkinsfile.

Improvements:

- Create redundancy in a singular VPC would be ideal in a real world case. The redundancy will allow resiliency in our systems.
- Implementing Nginx as our web server and a reverse proxy for our application.
- Placing our application and Gunicorn in a private subnet to introduce security and limit access from clients directly connecting to the application. A jump host or Nginx web server in a public subnet will be needed to implement this solution.

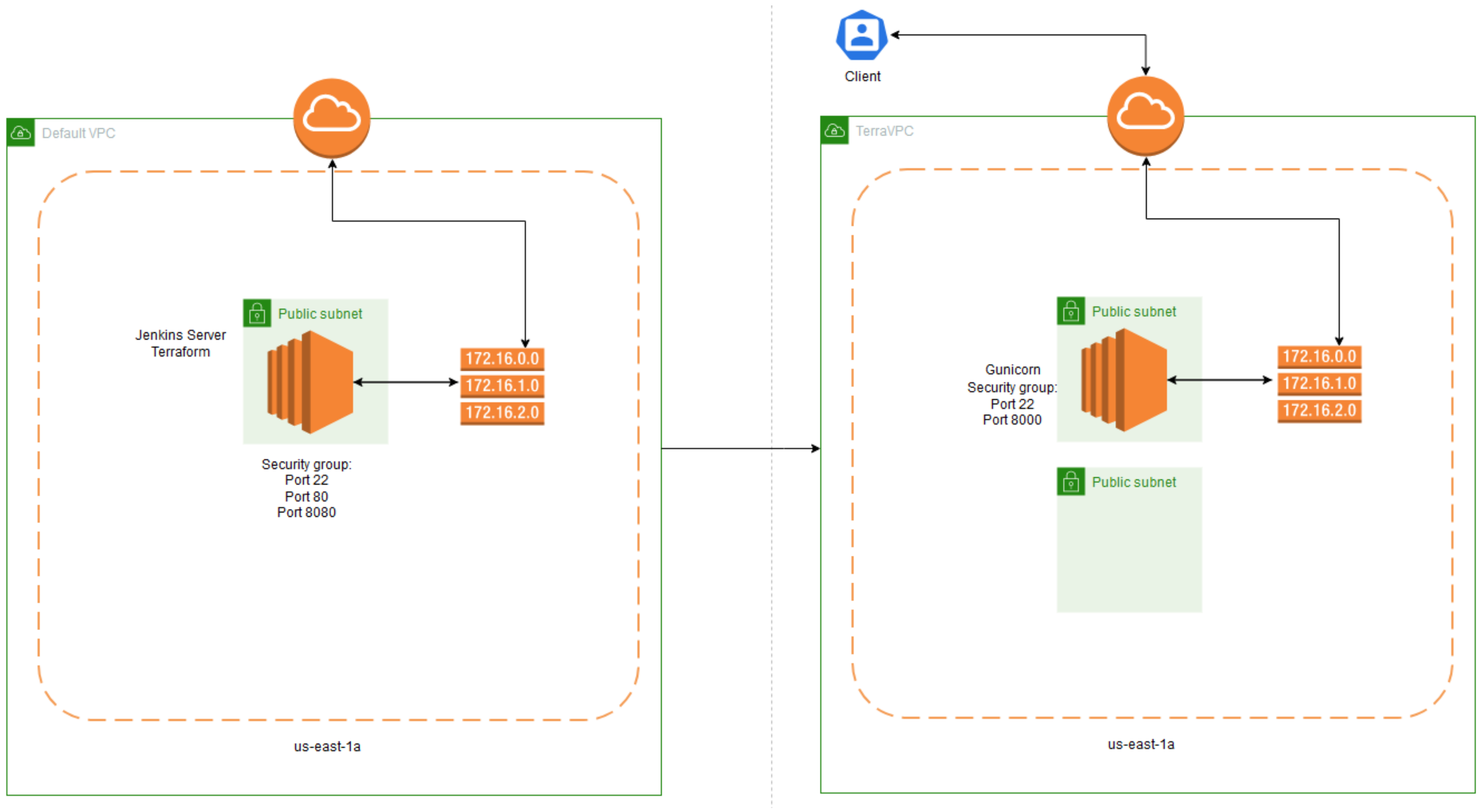
CI/CD Pipeline:



Key takeaways:

- Terraform will deploy the application based on the configuration provided in the main.tf file and the security group in the SG.tf file.
- The deploy.sh script will install the requirements needed for our flask application and Gunicorn to deploy the application onto the EC2 instance provisioned by Terraform.
- The linter was added as two separate commands in the build stage of the Jenkinsfile. If the linter fails in the build stage, Jenkins will return the build stage as having failed.

VPC Diagram:



Key Takeaways:

- Single availability zones for the subnets in the VPC's.
- The client will be sending requests to the EC2 instance with Gunicorn hosting the application.
- Client will connect to the application on port 8000.
- Terraform will provision the entire infrastructure on AWS from the Jenkins server.

Sources:

1. https://www.terraform.io/downloads
2. https://www.jenkins.io/doc/pipeline/steps/credentials-binding/
3. https://registry.terraform.io/modules/terraform-aws-modules/ec2-instance/aws/latest