Terraform Configuration

Raúl Estrada

Octubre 2020

IaC using Terraform

One of the tools that can help deploy infrastructure on AWS is HashiCorp's Terraform (https://www.terraform.io). HashiCorp is that genius bunch that gave us Vagrant, Packer, and Consul. I would recommend you look up their website if you have not already.

Using **Terraform** (**TF**), we will be able to write a template describing an environment, perform a **dry run** to see what is about to happen and whether it is expected, deploy the template, and make any late adjustments where necessary-all of this without leaving the shell prompt.

Configuration

Firstly, you will need to have a copy of TF (https://www.terraform.io/downloads.html) on your machine and available on the CLI. You should be able to query the currently installed version, which in my case is 0.6.15:

Copy

```
$ terraform --version
Terraform v0.6.15
```

Since TF makes use of the AWS APIs, it requires a set of authentication keys and some level of access to your AWS account. In order to deploy the examples in this chapter you could create a new **Identity and Access Management** (**IAM**) user with the following permissions:

Copy

```
    "autoscaling:CreateAutoScalingGroup",
    "autoscaling:CreateLaunchConfiguration",
    "autoscaling:DeleteLaunchConfiguration",
    "autoscaling:Describe*",
    "autoscaling:UpdateAutoScalingGroup",
    "ec2:AllocateAddress",
    "ec2:AssociateAddress",
    "ec2:AssociateRouteTable",
    "ec2:AttachInternetGateway",
    "ec2:AuthorizeSecurityGroupEgress",
    "ec2:AuthorizeSecurityGroupIngress",
    "ec2:CreateInternetGateway",
    "ec2:CreateNatGateway",
    "ec2:CreateRoute",
    "ec2:CreateRouteTable",
    "ec2:CreateSecurityGroup",
    "ec2:CreateSubnet",
    "ec2:CreateTags",
    "ec2:CreateVpc",
    "ec2:Describe*",
```

Copy

```
    "ec2:CreateNatGateway",
    "ec2:CreateRoute",
    "ec2:CreateRouteTable",
    "ec2:CreateSecurityGroup",
    "ec2:CreateSubnet",
    "ec2:CreateTags",
    "ec2:CreateVpc",
    "ec2:Describe*",
    "ec2:ModifySubnetAttribute",
    "ec2:RevokeSecurityGroupEgress",
    "elasticloadbalancing:AddTags",
    "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer",
    "elasticloadbalancing:AttachLoadBalancerToSubnets",
    "elasticloadbalancing:CreateLoadBalancer",
    "elasticloadbalancing:CreateLoadBalancerListeners",
    "elasticloadbalancing:Describe*",
    "elasticloadbalancing:ModifyLoadBalancerAttributes",
    "rds:CreateDBInstance",
    "rds:CreateDBSubnetGroup",
    "rds:Describe*"
```

One way to make the credentials of the IAM user available to TF is by exporting the following environment variables:

Copy

```
$ export AWS_ACCESS_KEY_ID='user_access_key'
$ export AWS_SECRET_ACCESS_KEY='user_secret_access_key'
```

This should be sufficient to get us started.