Craig ELlrod

senior cloud security architect

Deploying NGINX with TERRAFORM

Linux

Table of Contents

[Introduction 2](#_Toc146393859)

[Prerequisites 2](#_Toc146393860)

[Step-by-Step Prerequisites 2](#_Toc146393861)

[How it works 4](#_Toc146393862)

[Step-by-Step 5](#_Toc146393863)

[Step 1 – Run the AWS configure 5](#_Toc146393864)

[Step 2 – Add the userdata to the EC2 instance user data 5](#_Toc146393865)

[Step 3 – Add your AWS Key pair 5](#_Toc146393866)

[Step 4 – Add user data and keypair to the aws\_instance resource block 6](#_Toc146393867)

[Step 5 – Modify the ami id to the region 7](#_Toc146393868)

[Step 6 – Launch the EC2 instance 7](#_Toc146393869)

[Step 7 – SSH and validate nginx install 8](#_Toc146393870)

[Step 8 – Tear down (optional) 8](#_Toc146393871)

# Introduction

This REDlocust field guide will present a step-by-step method to deploy NGINX within a TERRAFORM script for Linux.

## Prerequisites

An Amazon Web Services account.

<https://aws.amazon.com/free/>

Terraform getting started. Version 15 or higher is preferred.

<https://learn.hashicorp.com/collections/terraform/aws-get-started>

A Linux host – For example Linux Ubuntu – running as a virtual machine in VirtualBox, with Docker installed.

<https://virtualbox.org>

<https://ubuntu.com/download>

<https://learn.hashicorp.com/tutorials/terraform/install-cli>

The github for this tutorial.

﻿ ﻿<https://github.com/r3dlocust/tutorial_tf_linux>

Note: A Terraform step-by-step tutorial is located here.

<https://learn.hashicorp.com/collections/terraform/aws-get-started>

Note2: The last “Store Remote State” step of the tutorial is not required.

## Step-by-Step Prerequisites

1. Install VirtualBox:

<https://www.youtube.com/watch?v=8mns5yqMfZk>

1. Install Ubuntu on VirtualBox:

<https://www.youtube.com/watch?v=x5MhydijWmc>

<https://linuxize.com/post/how-to-install-virtualbox-guest-additions-in-ubuntu/>

<https://download.virtualbox.org/virtualbox/6.1.0_BETA2/VBoxGuestAdditions_6.1.0_BETA2.iso>

<https://superuser.com/questions/1261903/auto-captute-keyboard-virtual-box>

**\*\*\* CREATE SNAPSHOT IN VIRTUALBOX \*\*\***

1. Install Visual Studio Code:

<https://www.youtube.com/watch?v=Y1fei1mzP7Q>

<https://code.visualstudio.com/docs/setup/linux>

**\*\*\* CREATE SNAPSHOT IN VIRTUALBOX \*\*\***

1. Install the Terraform Extension in VS Code:

<https://www.youtube.com/watch?v=PfI0fDr5yDc>

<https://www.terraform.io/cli/install/apt>

**\*\*\* CREATE SNAPSHOT IN VIRTUALBOX \*\*\***

1. Install the AWS CLI Toolkit:

<https://www.youtube.com/watch?v=IkTWpJVpkuk>

<https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>

**\*\*\* CREATE SNAPSHOT IN VIRTUALBOX \*\*\***

1. Install Terraform on Ubuntu 20.04:

<https://www.youtube.com/watch?v=apc-7t0nH70>

<https://www.terraform.io/cli/install/apt>

**\*\*\* CREATE SNAPSHOT IN VIRTUALBOX \*\*\***

1. Create a KeyPair in AWS Console:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html#having-ec2-create-your-key-pair>

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html#retrieving-the-public-key>

\*Note: Open a terminal, and enter these commands

ssh-keygen -y -f <keypair name>.pem

\*\*\*highlight and copy the output\*\*\*

nano ~/.ssh/<keypair name>.pub

Right click, paste

CTRL-o

CTRL-x

cp \*.pem ~/.ssh/

cp \*.pub ~/.ssh/

chmod 400 \*.pem \*.pub

**\*\*\* CREATE SNAPSHOT IN VIRTUALBOX \*\*\***

1. Optional - Install Chrome Browser:

<https://www.youtube.com/watch?v=k_iuTefIUMY>

<https://linuxways.net/ubuntu/how-to-install-google-chrome-on-ubuntu-20-04-lts/>

**\*\*\* CREATE SNAPSHOT IN VIRTUALBOX \*\*\***

## How it works

Installing the Ubuntu in AWS with terraform involves these simple steps.

* Insert the userdata script into the EC2 instance user data 🡪 main.tf terraform file.
* Add your AWS keypair 🡪 main.tf terraform file.
* Launch the EC2 instance through terraform automation 🡪 init; plan; apply.
* Tear down (optional).

# Step-by-Step

## Step 1 – Run the AWS configure

Run aws configure on the Ubuntu build machine

﻿Aws configure

AWS Access Key ID []

AWS Secret Access Key []

Default region []

Default output []

## Step 2 – Add the userdata to the EC2 instance user data

To get in the practice of installing custom commands through AWS EC2 userdata, use the following.

﻿locals {

user\_data = <<EOF

#!/bin/bash

sudo apt update

sudo apt install nginx

sudo systemctl start nginx

EOF

}

## Step 3 – Add your AWS Key pair

In order to ssh into your Linux instance, you need to specify the AWS keypair to the EC2 instance at launch. Add the following terraform resource block to the main.tf terraform file. Replace the “public key name” with the name of your AWS public keypair filename.

To create an AWS keypair, follow the AWS instructions.

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html>

Once the private key .pem file has been downloaded and moved to the ~/.ssh directory, you need to create the public key from that file, by following the “Retrieve the public key for your key pair” section in the above link.

At this point you should have a .pub (public) key and a .pem (private) key in the ~/.ssh directory. Add the following terraform resource block to the main.tf terraform file. Replace the “public key name” with the name of your AWS public keypair filename.

﻿ ﻿resource "aws\_key\_pair" "terraform\_pub\_key" {

public\_key = file("~/.ssh/<public key name>.pub")

}

## Step 4 – Add user data and keypair to the aws\_instance resource block

Once the user\_data and keypair resources are added to the main.tf terraform, those two parameters need to be added to the aws\_instance resource block of the main.tf terraform file. Add the following to the aws\_instance resource block in the main.tf terraform file.

﻿ user\_data\_base64 = base64encode(local.user\_data)

key\_name = aws\_key\_pair.terraform\_pub\_key.key\_name

For example, the final code block for the main.tf terraform should look similar to the following. The “new” pieces to the main.tf terraform file are highlighted in orange.

Note: Theoretically, you could copy and paste the below code block into the main.tf terraform file, and run it, of course replacing your license key and keypair file name.

﻿terraform {

required\_providers {

aws = {

source = "hashicorp/aws"

version = "~> 5.0"

}

}

}

# Configure the AWS Provider

provider "aws" {

region = "us-east-1"

}

locals {

user\_data = <<EOF

#!/bin/bash

sudo apt update

sudo apt install nginx

EOF

}

resource "aws\_instance" "app\_server" {

ami = "ami-0db6c6238a40c0681"

instance\_type = "t2.micro"

user\_data\_base64 = base64encode(local.user\_data)

key\_name = aws\_key\_pair.terraform\_pub\_key.key\_name

tags = {

Name = var.instance\_name

}

}

resource "aws\_key\_pair" "terraform\_pub\_key" {

public\_key = file("~/.ssh/<public key name>.pub")

}

variable "instance\_name" {

description = "Value of the Name tag for the EC2 instance"

type = string

default = "ExampleAppServerInstance"

}

output "instance\_id" {

description = "ID of the EC2 instance"

value = aws\_instance.app\_server.id

}

﻿

output "instance\_public\_ip" {

description = "Public IP address of the EC2 instance"

value = aws\_instance.app\_server.public\_ip

}

## Step 5 – Modify the ami id to the region

You will need to lookup the latest AMI instance id from the following website: <https://cloud-images.ubuntu.com/locator/ec2/>

Filter on your region, Version = 20.04 LTS, Arch = amd64. (The arm64 version won’t work).

Substitute the existing ami-0db6c6238a40c0681 with the AMI-ID from the webpage.

Note: Save the file.

## Step 6 – Launch the EC2 instance

As you learned in the terraform tutorial, to launch the EC2 instance enter the following commands in the terminal or CLI.

$terraform init

$terraform plan

$terraform apply

Note: If you log in to the AWS console, (<https://console.aws.amazon.com/console/home>) and navigate to EC2 instances you will see the instance spinning up.

## Step 7 – SSH and validate nginx install

If you want to ssh into the EC2 instance you can perform the ssh commands as listed in <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstancesLinux.html>.

For example, for an Ubuntu 20.04 EC2 instance:

ssh -i ~/.ssh/<private key name>.pem ubuntu@<public ip address>  
sudo systemctl status nginx

Note: The public ip address of the EC2 instance can be found in the AWS console and was also output as a part of the terraform automation.

Note2: Each distribution of Linux has a different username in AWS. In the above example, for Ubuntu, the username is ubuntu. For Amazon Linux 2, the username is ec2-user. Usernames for the different Linux versions can be found here: <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/managing-users.html>.

## Step 8 – Tear down (optional)

If you want to reverse the automation that terraform performed, simply enter the following commands.

﻿$ terraform destroy