



NextWork.org

Create S3 Buckets with Terraform



tahirgroot@gmail.com

Windows

Binary download

386

Version: 1.10.2

[Download](#)

AMD64

Version: 1.10.2

[Download](#)



TA

tahirgroot@gmail.com

NextWork Student

NextWork.org

Introducing Today's Project!

In this project, I will demonstrate how to use terrafrom to launch an S3 bucket. The goal is to install + setup Terraform, troubleshoot any errors (i.e. Installing the CLI), and sucessfully plan + apply terraform config to launch s3 bucket

Tools and concepts

Services I used were Terraform, Amazon S3 AWS CLI, AWS IAM Key concepts I learnt include infrastructure as code, creating configuration files, modularity of code, using providers, plugins, state files, lock files, terraform concepts

Project reflection

This project took me approximately one hour The most challenging part was modifying s3 using terraform documentation.

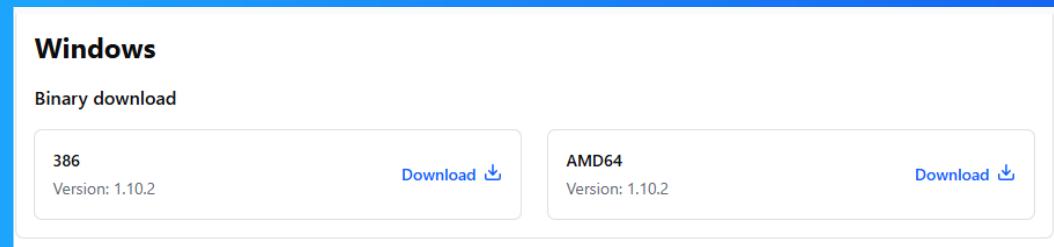
n/a

Introducing Terraform

Terraform is a tool for managing IT resources using code. I use Terraform to process a configuration file, I've prepared the details the desired state of the infrastructure. Terraform then builds to setup that desired state.

Terraform is one of the most popular tools used for infrastructure as code (IaC), which is a way to manage IT infrastructure. Instead of manually managing resources using the console/text commands in CLI, IAC automate that process with code.

Terraform uses configuration files to understand the desired state of my infrastructure main.tf is that i use in terraform to describe the desired state.

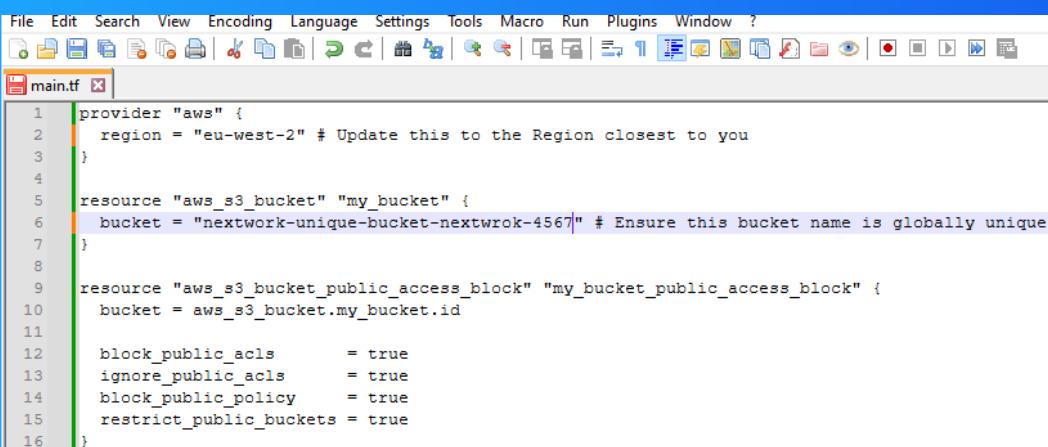


Configuration files

The configuration is structured in blocks instead of a single block of code. The advantage of doing this is modularity is the ability to workon/update different parts of main.tf without affecting other parts. great for teams collaborating to

My main.tf configuration has three blocks

The first block indicates that we are using AWS for Ifrastructure The second block provisions an Amazon S3 bucket The third block manages the buckets permissions I.e. blocking all public access



```
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
main.tf
1 provider "aws" {
2   region = "eu-west-2" # Update this to the Region closest to you
3 }
4
5 resource "aws_s3_bucket" "my_bucket" {
6   bucket = "nextwork-unique-bucket-nextwrok-4567" # Ensure this bucket name is globally unique
7 }
8
9 resource "aws_s3_bucket_public_access_block" "my_bucket_public_access_block" {
10   bucket = aws_s3_bucket.my_bucket.id
11
12   block_public_acls      = true
13   ignore_public_acls    = true
14   block_public_policy   = true
15   restrict_public_buckets = true
16 }
```

Customizing my S3 Bucket

For my project extension, I visited the official Terraform documentation to look for ways I can customize my bucket's configurations. The documentation shows example config all the available parameter for a resource and the rules for using it.

I chose to customise my bucket by adding tags because that lets me identify the project that i launched it for later on. When I launch my bucket, I can verify my customization by visting the 'tags' panel in the s3 bucket.

```
resource "aws_s3_bucket" "my_bucket" {
  bucket = "nextwork-unique-bucket-nextwrok-4567" # Ensure this bucket name is globally unique

  tags = {
    Project = "Create an S3 bucket with Terraform"
  }
}
```

Terraform commands

I ran 'terraform init' to initialize terraform, this means getting it set up the backend to store state files, and install the necessary plugins i.e. the AWS Provider plugin

Next, I ran 'terraform plan' to compare the resources/infrastructure in the main.tf file, and compare that against infrastructure current state. Then share back the execution plan i.e. how running main.tf will impact our infrastructure.

```
PS C:\Users\UseR\OneDrive\Desktop\nextwork_terraform> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.81.0...
- Installed hashicorp/aws v5.81.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

AWS CLI and Access Keys

When I tried to plan my Terraform configuration, I received an error message that says no valid credentials found because my local terminal was not setup with aws credinitals yet. this needs to be setup use aws provider plugin

To resolve my error, first I installed AWS CLI, which is AWS coomand line tool for interacting with AWS infrastructure. instead of using the AWS Management Console, the CLI lets me run text commands to do the same thing from local pc

I set up AWS access keys so that we have a way to 'log into my AWS account over the cli in my local computer. once the access key ID and secret access key are passed through 'aws configure' our terminal now has the neccessary AWS credentials.

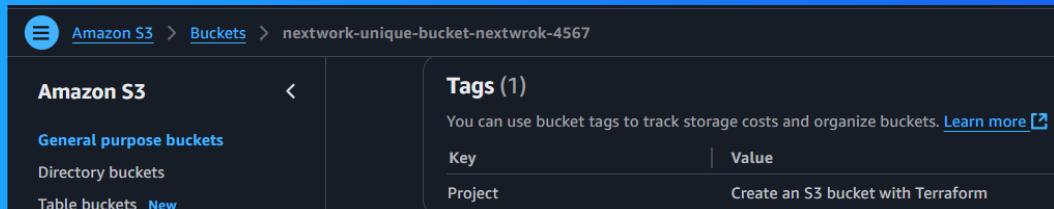
```
PS C:\Users\UseR\OneDrive\Desktop\Nextwork_terraform> terraform plan
Planning failed. Terraform encountered an error while generating this plan.

Error: No valid credential sources found
  with provider["registry.terraform.io/hashicorp/aws"],
  on main.tf
  1: provider "aws" {
```

Lanching the S3 Bucket

I ran 'terraform apply' to get terraform to apply the changes/updates it showed us when we ran 'terraform plan'. Running 'terraform apply' will affect my AWS account by creating two resources -an s3 bucket and its set of permission settings

The sequence of running terraform init, plan, and apply is crucial because I need to initialize terraform first before we get to make any comparisons between main.tf and current infrastructure and connect to aws in the first place

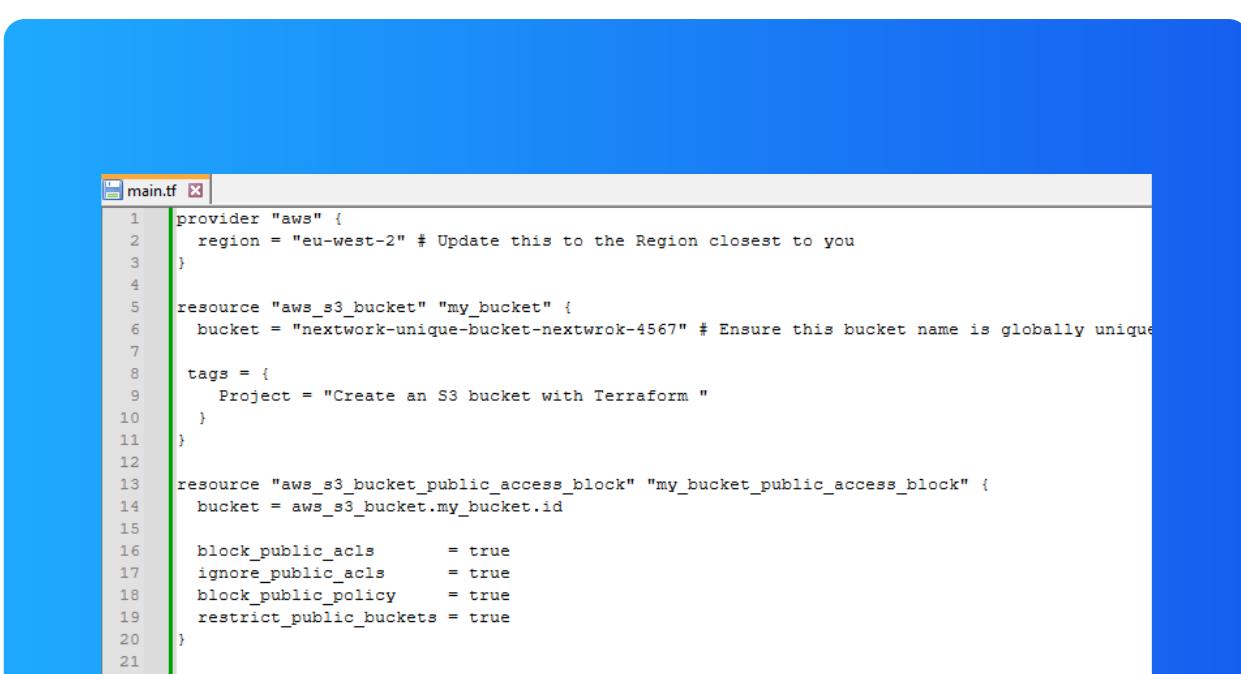


Uploading an S3 Object

I created a new resource block to upload an image called image.png inot the s3 bucket the image will be uploaded straight away without being nested in a subfolder in the bucket.

We need to run terraform apply again because i have updated terraform config file, which means there are changes that terraform needs to review and compare with my current infrastructure again. This time only new resource is created in the image

To validate that I've updated my configuration successfully, I checked the s3 bucket and confirmed a new image is inside i also downloaded the image back to me and confirmed that it was the correct image that was initally uploaded



```
main.tf
1 provider "aws" {
2   region = "eu-west-2" # Update this to the Region closest to you
3 }
4
5 resource "aws_s3_bucket" "my_bucket" {
6   bucket = "nextwork-unique-bucket-nextwrok-4567" # Ensure this bucket name is globally unique
7
8   tags = {
9     Project = "Create an S3 bucket with Terraform "
10    }
11  }
12
13 resource "aws_s3_bucket_public_access_block" "my_bucket_public_access_block" {
14   bucket = aws_s3_bucket.my_bucket.id
15
16   block_public_acls      = true
17   ignore_public_acls    = true
18   block_public_policy   = true
19   restrict_public_buckets = true
20 }
21
22 resource "aws_s3_object" "image" {
```



NextWork.org

Everyone should be in a job they love.

Check out nextwork.org for
more projects

