SYSTEM PROVISIONING AND CONFIGURATION MANAGEMENT

(ASSIGNMENT 1)

Abhinav Sharma

Terraform scripts to perform following tasks on AWS cloud Platform

1. Creating two T2 micro ec2 instances
2. Creating a VPN on AWS
3. Creating a S3 bucket

What is Terraform?

Terraform is an open source tool for infrastructure provisioning created by HashiCorp. It provides Infrastructure as code allowing you to automate and manage your infrastructure, platform and your services that run on the platform. Terraform can manage existing and popular service providers(aws, azure, GCP etc).

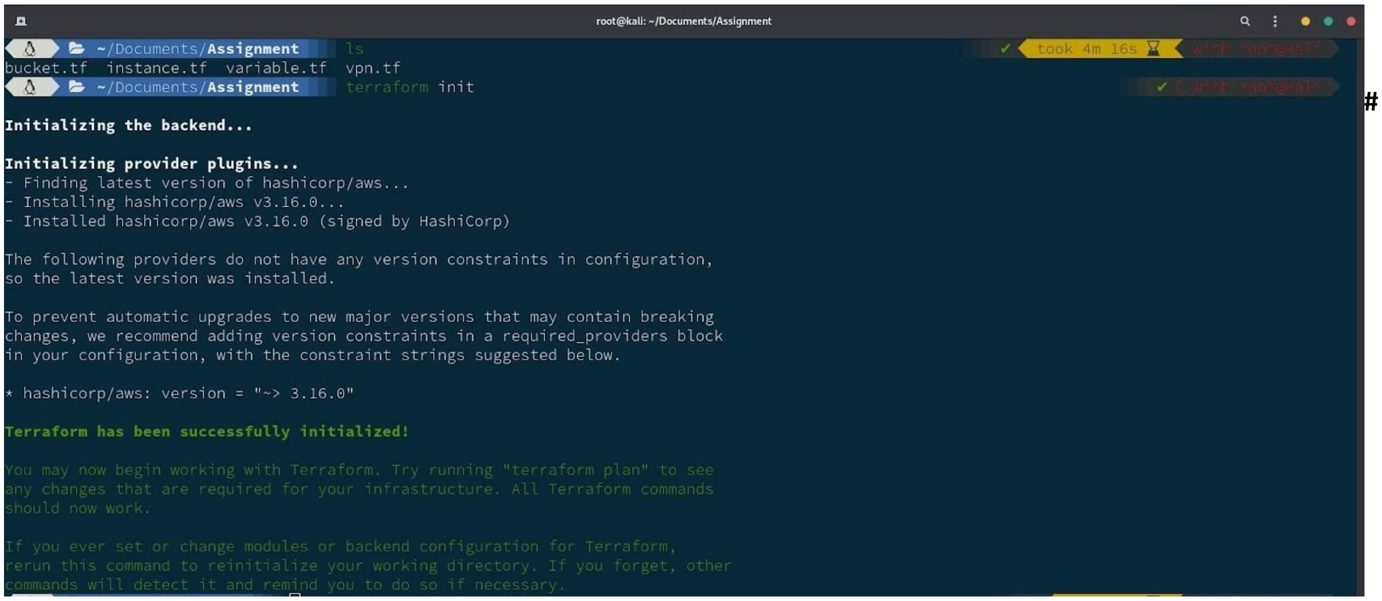
You do not have to prepare infrastructure like private network space, ec2 server instances, installing docker and other tools and security. Terraform does all that for you by preparing the whole infrastructure using terraform scripts. Thus, it is a software tool that provides Infrastructure as code.

Terraform is declarative which means you define what you want.

Run the following command

terraform init

Initializes working directory containing terraform configuration files. It is safe to run this command multiple times

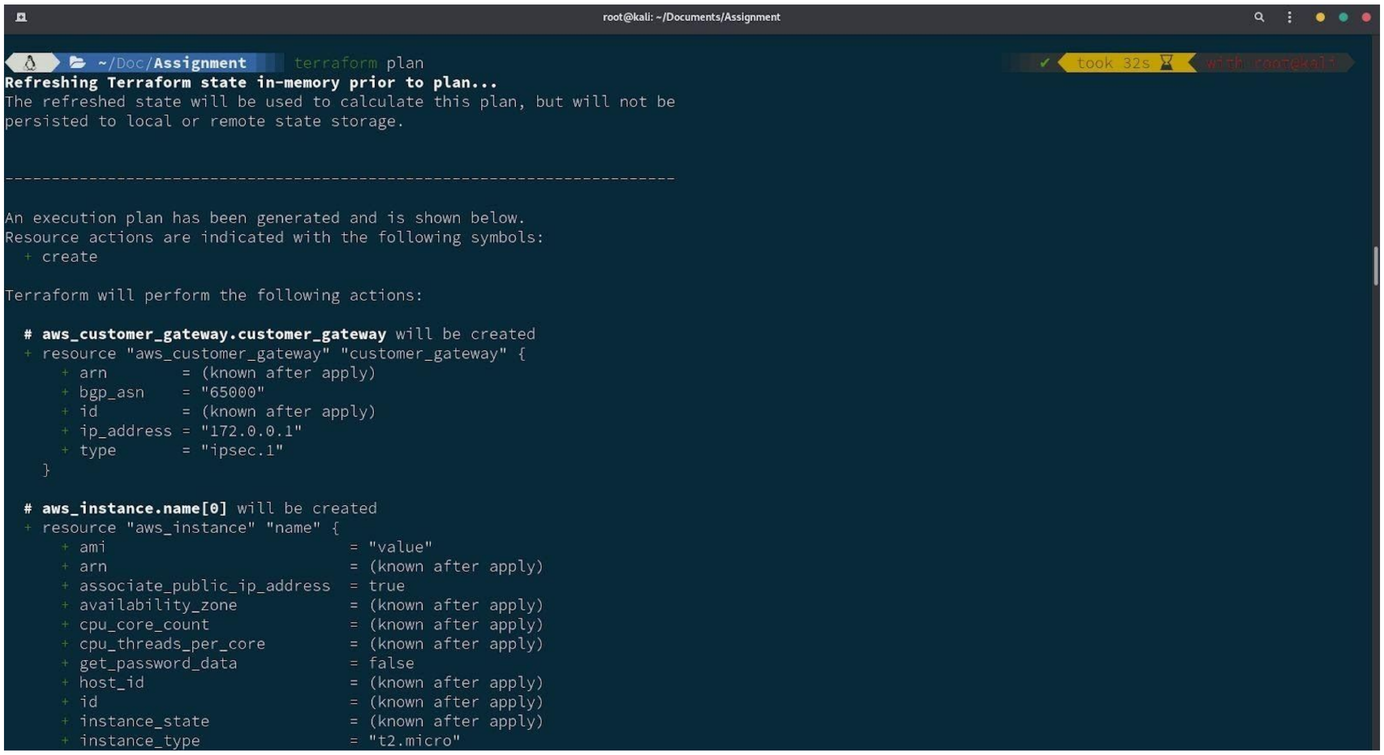


```
root@kali: ~/Documents/Assignment

 ~/Documents/Assignment  ls                                              ✔  took 4m 16s   with root@kali
bucket.tf  instance.tf  variable.tf  vpn.tf
 ~/Documents/Assignment  terraform init                                            ✔  with root@kali

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v3.16.0...
- Installed hashicorp/aws v3.16.0 (signed by HashiCorp)

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking
changes, we recommend adding version constraints in a required_providers block
in your configuration, with the constraint strings suggested below.

* hashicorp/aws: version = "~> 3.16.0"

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.
```

#

terraform plan

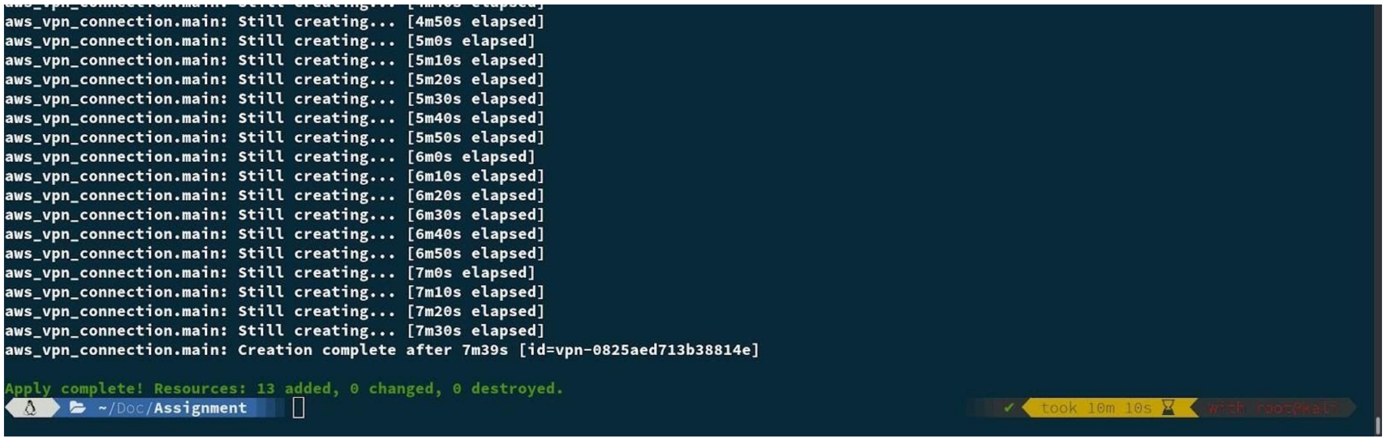
To create execution plan that helps you check whether execution plan matches your Expectations



```
root@kali: ~/Documents/Assignment

 ~/Doc/Assignment  terraform plan                                         ✔  took 32s   with root@kali
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.


------------------------------------------------------------------------

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_customer_gateway.customer_gateway will be created
  + resource "aws_customer_gateway" "customer_gateway" {
      + arn        = (known after apply)
      + bgp_asn    = "65000"
      + id         = (known after apply)
      + ip_address = "172.0.0.1"
      + type       = "ipsec.1"
    }

  # aws_instance.name[0] will be created
  + resource "aws_instance" "name" {
      + ami                          = "value"
      + arn                          = (known after apply)
      + associate_public_ip_address  = true
      + availability_zone            = (known after apply)
      + cpu_core_count               = (known after apply)
      + cpu_threads_per_core         = (known after apply)
      + get_password_data            = false
      + host_id                      = (known after apply)
      + id                           = (known after apply)
      + instance_state               = (known after apply)
      + instance_type                = "t2.micro"
```

# terraform apply -auto-approve

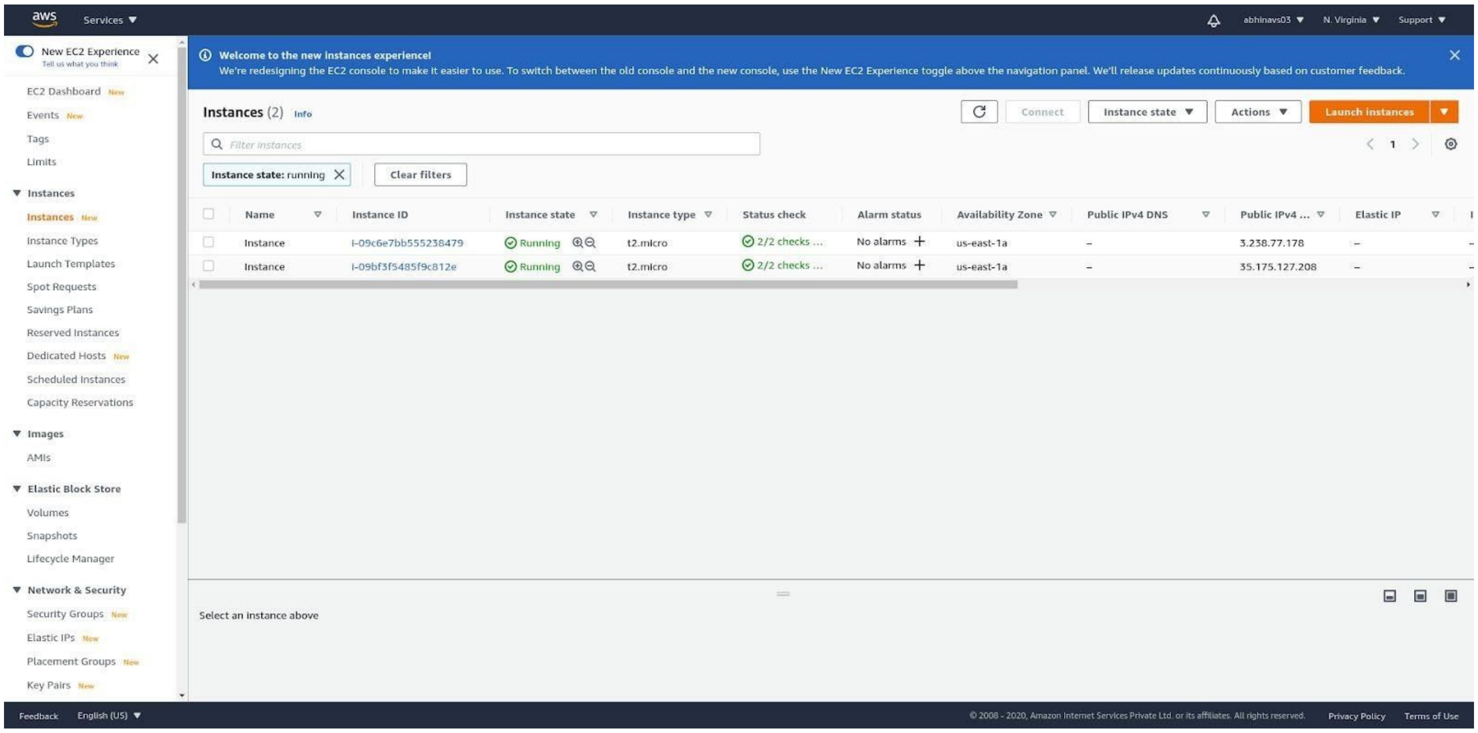
To apply the changes to reach the desired state of the configuration



```
aws_vpn_connection.main: Still creating... [4m50s elapsed]
aws_vpn_connection.main: Still creating... [5m0s elapsed]
aws_vpn_connection.main: Still creating... [5m10s elapsed]
aws_vpn_connection.main: Still creating... [5m20s elapsed]
aws_vpn_connection.main: Still creating... [5m30s elapsed]
aws_vpn_connection.main: Still creating... [5m40s elapsed]
aws_vpn_connection.main: Still creating... [5m50s elapsed]
aws_vpn_connection.main: Still creating... [6m0s elapsed]
aws_vpn_connection.main: Still creating... [6m10s elapsed]
aws_vpn_connection.main: Still creating... [6m20s elapsed]
aws_vpn_connection.main: Still creating... [6m30s elapsed]
aws_vpn_connection.main: Still creating... [6m40s elapsed]
aws_vpn_connection.main: Still creating... [6m50s elapsed]
aws_vpn_connection.main: Still creating... [7m0s elapsed]
aws_vpn_connection.main: Still creating... [7m10s elapsed]
aws_vpn_connection.main: Still creating... [7m20s elapsed]
aws_vpn_connection.main: Still creating... [7m30s elapsed]
aws_vpn_connection.main: Creation complete after 7m39s [id=vpn-0825aed713b38814e]

Apply complete! Resources: 13 added, 0 changed, 0 destroyed.
  ~/Doc/Assignment                                             ✓  took 10m 10s
```

Now you can check the instances, VPN and S3 bucket have been created on your AWS cloud.

One t2-micro ec2-instance is created in Mumbai region and the other in N. Virginia region.



Welcome to the new instances experience!
We're redesigning the EC2 console to make it easier to use. To switch between the old console and the new console, use the New EC2 Experience toggle above the navigation panel. We'll release updates continuously based on customer feedback.

Instances (2) Info

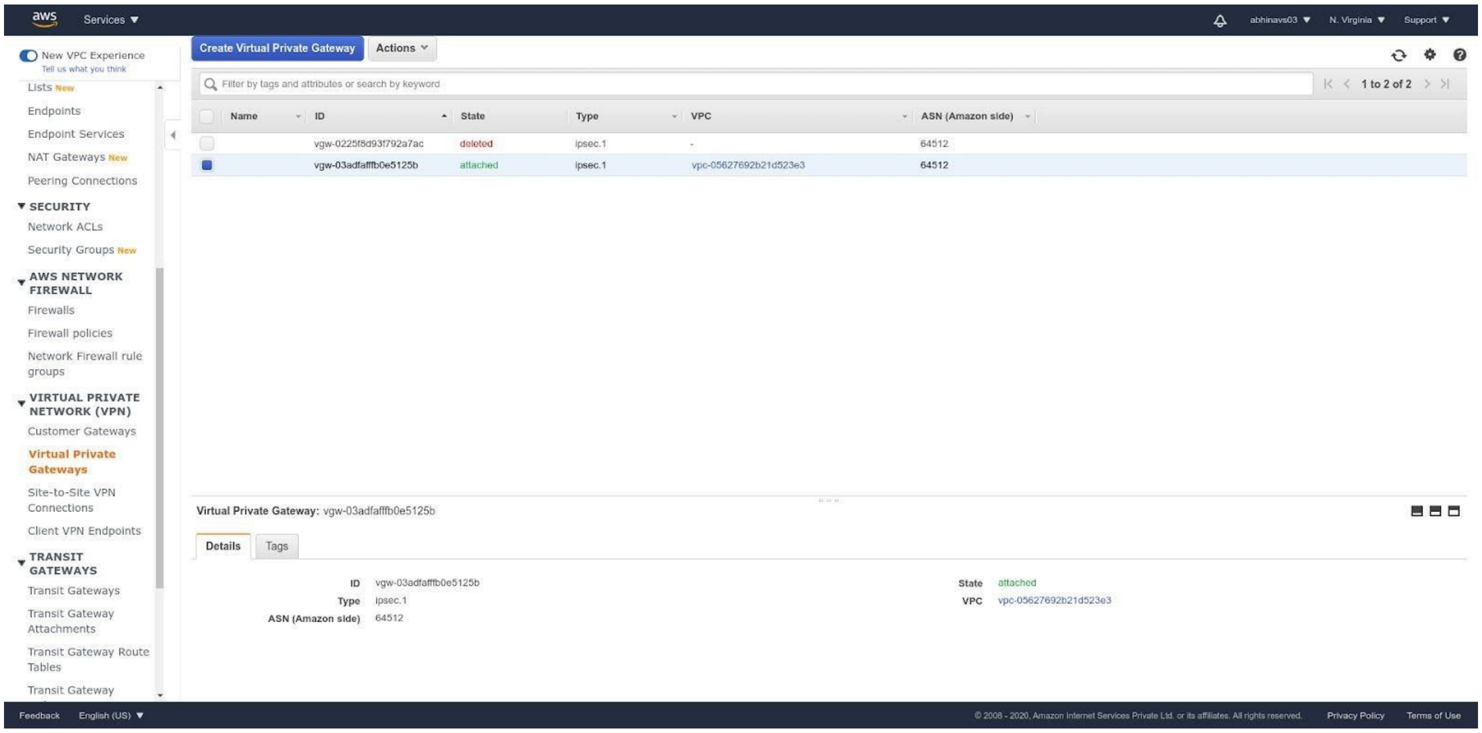
Instance state: running

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
Instance	i-09c6e7bb555238479	Running	t2.micro	2/2 checks ...	No alarms	us-east-1a	–	3.238.77.178	–
Instance	i-09bf3f5485f9c812e	Running	t2.micro	2/2 checks ...	No alarms	us-east-1a	–	35.175.127.208	–

Select an instance above

© 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.

VPN



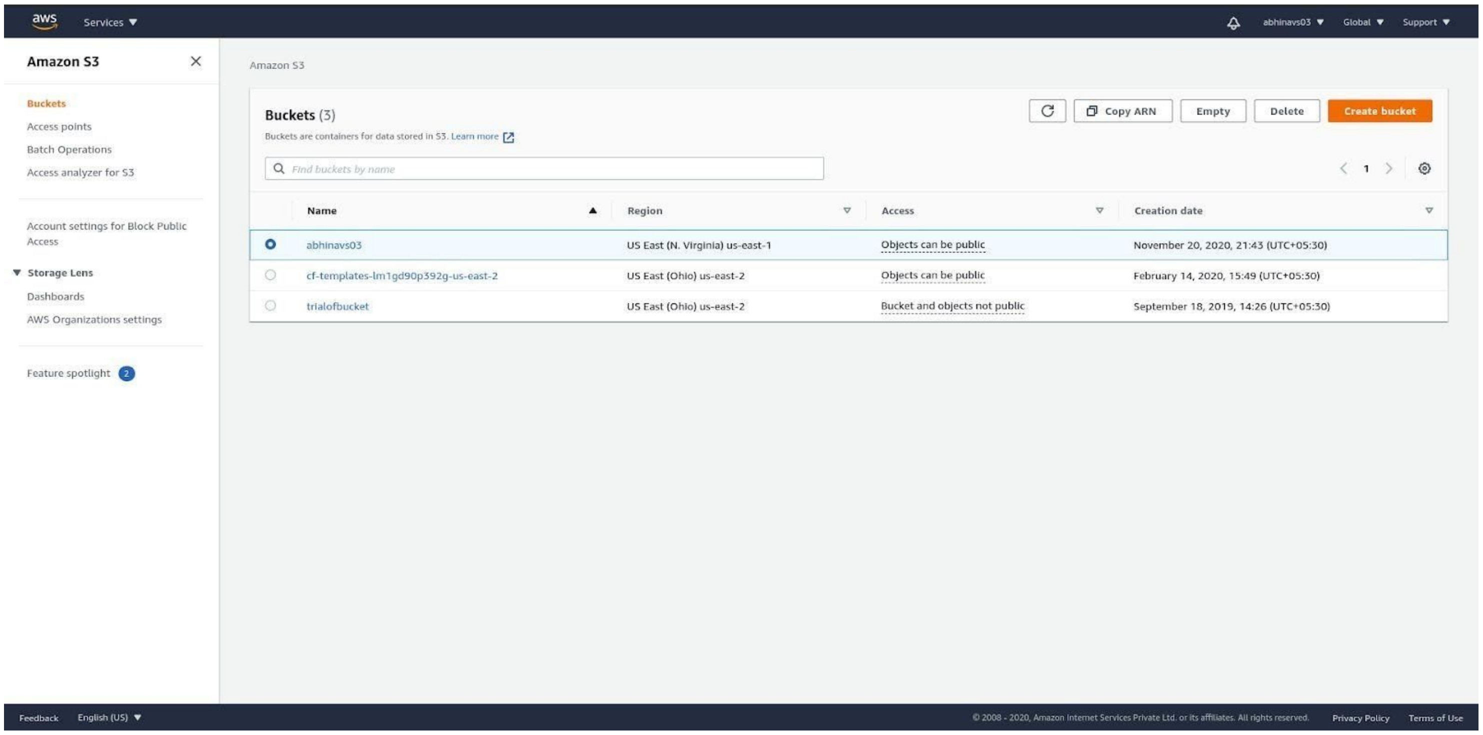
Name	ID	State	Type	VPC	ASN (Amazon side)
	vgw-0225f8d93f792a7ac	deleted	ipsec.1	-	64512
	vgw-03adfafffb0e5125b	attached	ipsec.1	vpc-05627692b21d523e3	64512

Virtual Private Gateway: vgw-03adfafffb0e5125b

Details | Tags

ID	vgw-03adfafffb0e5125b	State	attached
Type	ipsec.1	VPC	vpc-05627692b21d523e3
ASN (Amazon side)	64512		

S3 bucket



Buckets (3)

Buckets are containers for data stored in S3. Learn more

Name	Region	Access	Creation date
abhinavs03	US East (N. Virginia) us-east-1	Objects can be public	November 20, 2020, 21:43 (UTC+05:30)
cf-templates-lm1gd90p392g-us-east-2	US East (Ohio) us-east-2	Objects can be public	February 14, 2020, 15:49 (UTC+05:30)
trialofbucket	US East (Ohio) us-east-2	Bucket and objects not public	September 18, 2019, 14:26 (UTC+05:30)