Dynamic Web Implementation & Management by Terraform

1. [Configuration AWS to interact with Terraform]

* IAM > Security Credentials -> Issue Access Key & Secret Access Key
* Install the AWS CLI on Windows
* Locates in the local directory and configures the Access keys to AWS
* After the configuration[~/.aws/credentials], the new environment connects with HTTPS Address(GitHub)

1. [main.tf]

**Structures >>**

1. Essential Network Configuration [main.tf]

* Provider: aws with region = “eu-west-3”
* VPC / IGW / 2 Public Subnet[AZ1, AZ2] / 1 Route Table Public
* 2 Route Table Association with Public Subnet[AZ1, AZ2]
* 4 Private Subnet: Web, DB server [AZ1, AZ2]
* 2 EIP for 2 NAT gateway and Attach each EIP to NAT gateway[AZ1, AZ2]
* 2 Route Table Private for Availability Zone
* 4 Route Table Association with Private Subnet[AZ1, AZ2]

\*\* All resource “aws\_route\_table” Route to 0.0.0.0/0 -> All traffic[cidr\_block]

Route to IGW or NAT gateway

1. Security Group Configuration [main.tf]

* Create 2 Start Security Groups on VPC:

[1] bastion\_ssh(22) [2] load\_balancer(80,443)

* Inbound Traffic via TCP -> aws\_vpc\_security\_group\_ingress\_rule
* Outbound Traffic via “-1” -> aws\_vpc\_security\_group\_egress\_rule
* Outbound traffic doesn’t need to be configured using TCP protocol.
* Create 3 Security Groups on VPC

[3] web\_server(22,80,443) [4] db\_migration(22) [5] rds\_instance(22,3306)

* Inbound Traffic via TCP -> aws\_vpc\_security\_group\_ingress\_rule
* Inbound Traffic configuration using referenced\_security\_group\_id
* Admin can access Webserver VIA [bastion\_ssh: 22] & [load\_balancer: 80,443]
* Admin can conduct DB\_migration VIA [bastion\_ssh: 22]
* Admin can access RDS VIA [Webserver’s DB] & [After DB migration]: 3306
* Outbound Traffic via “-1” -> aws\_vpc\_security\_group\_egress\_rule

1. DB center: RDS instance Configuration: [main.tf/ data.tf]

* aws\_db\_subnet\_group to place the RDS onto the DB subnets[AZ1, AZ2].
* Automation: S3 Create ->
* [Data Migration: sql file using ]
* aws\_db\_instance
* Because this Dynamic website is implemented using LAMP Server.
* After Data Migration, the database is accessed via MySQL.
* Web Server is accessed via Bastion
* resource “aws\_internet\_gateway” => attaching IGW to VPC
* resource “aws\_subnet” => creating 2 Public Subnets
* Added attributes:
* availability\_zone: Public subnets are placed in two Availability Zones.
* map\_public\_ip\_on\_launch: The Public subnet needs A Public IPv4 address.
* resource “aws\_route\_table” => Routing Traffic to IGW
* The public route table should route to EVERYWHERE[0.0.0.0/0] through IGW.
* resource “aws\_route\_table\_association”

=> Place the Route table in each subnet

* resource “aws\_subnet” => creating 4 Private Subnets
* To manage the DB instance and the Web Instance separately
* resource “aws\_eip” => creating the 2 Public IPs for NATs
* resource “aws\_nat\_gateway” => creating 2 NAT gateway
* The NAT Gateway is placed in the public subnet[1,2] and has each EIP.
* Plus, it is dependent on the IGW gateway to communicate with WWW.
* resource “aws\_route\_table” => creating Route table in each NAT
* Thus, each route table routes to the [NAT/IGW] Gateway. And the NAT Gateway is attached to the IGW gateway, which is attached to VPC. [NAT->IGW->VPC]
* resource “aws\_route\_table\_association”

=> Place the 4 Route tables in 4 Subnets

* AZ1 Route table is placed on the AZ1 subnets[DB1, Web1].
* AZ2 Route table is placed on the AZ2 subnets[DB2, Web2].
* **EVERY AZ**, 1 Route Table Routing to-> 1 NAT Gateway

1. variables.tf
2. outputs.tf

* To manage the manifest information of the Resources, this outputs.tf includes the essential information to monitor and check.

1. Organise the Current Workload: output “current\_env” / output “current\_vpc”
2. Divide the Availability Zones & Check each Subnet CIDR & EIP

* output “public\_subnets”
* output “private\_az1\_subnets” // output “eip1\_az1”
* output “private\_az2\_subnets” // output “eip2\_az2”

1. Terraform APPLY

* terraform init -> terraform plan -> terraform apply
* Error 1: [Wrong address CIDR]
* The subnet CIDR has an error due to assigning an Invalid Network address.

Subnet Mask /27

🡪 (32-27) => 2^5(host bits) = every 32 addresses per subnet

Subnet Mask /28

🡪 (32-28) => 2^4(host bits) = every 16 addresses per subnet

=> 192.168.56.0, 16, 32, 64 … // = > 192.168.56.0, 32, 64, 96, …

* Error 2: [IAM User Authorisation]

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

* Add the Policies to IAM User[admin\_dwa]:

AmazonEC2FullAccess / AmazonVPCFullAccess

* Success > The output contents are identified the manifest information.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명