TERRAFORM

|  |  |
| --- | --- |
| exp of vpc subnet etc | <https://docs.aws.amazon.com/vpc/latest/userguide/how-it-works.html> |
| Terraform Crash Course 1:10. (Ref for below) | https://www.youtube.com/watch?v=b1P2AH9bjpI |
| 8 Terraform Best Practices that will improve your TF workflow immediately | https://www.youtube.com/watch?v=gxPykhPxRW0 |
| An **Elastic IP address** | https://aws.amazon.com/premiumsupport/knowledge-center/intro-elastic-ip-addresses/ |

|  |
| --- |
| Download terraform on terminal brew install terraform  Terraform -v  Other than mac sudo mv terraform/user/local/bin  Create a directory with in it create a folder  In visual studio code go to the above folder create a file .tf ext for file 20:54  Hasicorp config lang |

CMD (used in our terminal in the dir in the folder where we stored our .tf file.)

# comment

Terraform plan (dry run

Terraform state list (**shows the resource addresses for every resource**)

Terraform state show (state show command displays detailed state data about one resource.)

Terraform apply ( actual working if already used once it check any new updates and apply if nothing leaves the same)

Creating specific resource 2:03

Terraform destroy (will destroy all the infra

Terraform destroy -target 2:0 specify the recourse u want to delete

Instead destroy # unwanted & apply

Can create ec2 vpc-subnet

(provider-aws , resourse-ami, tags-name)

Never delete terraform.tfstate file(is track of happenings

Terraform destroy - -auto-approve

Terraform hit enter will giv u. the list of cmd 1:49

Out put 1:55:46 (enter in the resourses where u want the out put ,only 1 req in each out put)🡪terraform apply(better terraform refresh)

To see use terraform output

Defining variable in terra form 2:04:20

Creating otherf iles2 :09:58

Saying terraform to look for specific file 2:11

PRACTICE

Create an Ec2 instance deploy on a custom vpc & a custom subnet & assign a public ip address so that we can ssh to it &connect to it to make changes and automatically set up web server to run on it and we can end web traffic 1:10

In AWS

1. Create a key pair(services 🡪Ec2 section🡪NT wrk &sec🡪Keypairs🡪create new🡪name🡪pem(mac r lin) 🡪can. Be converted to ppk🡪creat key🡪automatically dwnlds.

In browser

1. Search for terraform aws vpc 🡪one with hashi corp Create vpc 1

https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/vpc

1. Terr – aws – internet gateway🡪 creating internet gate way2 <https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/internet_gateway>

|  |
| --- |
| An internet gateway is a horizontally scaled, redundant, and highly available VPC component that allows communication between your VPC and the internet.An internet gateway supports IPv4 and IPv6 traffic. It does not cause availability risks or bandwidth constraints on your network traffic. |

1. Create a custom route table(terraform aws route table) 3

<https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/route_table>

|  |
| --- |
| A route table contains a set of rules, called routes, that are used to determine where network traffic from your subnet or gateway is directed. To put it simply, a route table **tells network packets which way they need to go to get to their destination**  **When u copy** egress\_only\_gateway\_id = to gateway\_id = (default route of ipv6&4 go to internet gateway so we can go to internet ) |

1. creat a subnet 4

https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/subnet

1. terraform aws route table associate 5 associate subnet with route table

<https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/route_table_association>

1. terraform aws security group 6

<https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/security_group>

|  |
| --- |
| Egress in the world of networking implies traffic that exits an entity or a network boundary, while Ingress is traffic that enters the boundary of a network. While in service provider types of the network this is pretty clear, in the case of datacenter or cloud it is slightly different. In the cloud, Egress still means traffic that’s leaving from inside the private network out to the public internet, but Ingress means something slightly different.  "0.0.0.0/0" allows any incoming if u specify ip it will allow that only  -1 any protocol |

1. terraform aws network interface 7

<https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/network_interface>

|  |
| --- |
| "10.0.0.50" is the add resrv by aws(reserves some addd) u cange to one u like "10.0.1.50"  Pass in the sec grp detail u created.  We can attach a device (we skipped here)   1. attachment { 2. instance = aws\_instance.test.id 3. device\_index = 1 4. }   This created a private ip in nxt step u create public ip so anybody on internet can access it.  <https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/eip> (prioide elstic IP)  -avilability zone hard coded & interface in same so that both can be connected.(serv & subnet same zone)  Depends\_on=should be in []so that u can pass many dependencies in alist like vpn, subnet etc  Net work interface block https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/instance |

APT stands **for Advanced Package Tool**. It is a set of core tools inside Debian and Ubuntu Linux system. You can use the apt command to install apps, delete or remove apps, keep Ubuntu/Debian server up to date and more. Apt work with dpkg command to install and update the system.

Terraform aws eip 8

https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/eip

1. provider "aws"
2. region = "us-east-1"
3. Acess\_key. = "xxx"
4. Secreate\_key = "xxxx"
5. #Create vpc 1
6. resource "aws\_vpc" "main"
7. cidr = "10.0.0.0/16"
8. tags = {
9. Name = "production"
10. }
11. }
12. #creating internet gate way 2
13. resource "aws\_internet\_gateway" "gw" {
14. vpc\_id = aws\_vpc.main.id
15. }
16. #create a custom route table 3
17. resource "aws\_route\_tables" "Prod-route-table" {
18. vpc\_id = aws\_vpc.main.id
20. Route{
21. Cidr\_block = "0.0.0.0/0"
22. Gateway\_id = aws\_internet\_gateway.gw.id
23. }
24. route {
25. ipv6\_cidr\_block = "::/0"
26. gateway\_id = aws\_internet\_gateway.gw.id
27. }
28. tags = {
29. Name = "prod"
30. }
31. #create a sub net 4
32. resource "aws\_subnet" "subnet-1" {
33. vpc\_id = aws\_vpc.main.id
34. cidr\_block = "10.0.1.0/24"
35. Availability\_zone = "us-east-1a"
36. Tags = {
37. Name = "prod-subnet"
38. }
39. }
40. # associate subnet with route table 5
41. resource "aws\_route\_table\_association" "a" {
42. subnet\_id = aws\_subnet.subnet-1.id
43. route\_table\_id = aws\_route\_tables.Prod-route-table.id
44. }
45. #create a security group 6
46. resource "aws\_security\_group" "allow\_web" {
47. name = "allow\_web\_traffic "
48. description = "Allow web inbound traffic"
49. vpc\_id = aws\_vpc.main.id
50. ingress {
51. description = "HTTPS from VPC"
52. from\_port = 443
53. to\_port = 443
54. protocol = "tcp"
55. cidr\_blocks = ["0.0.0.0/0"]
56. ipv6\_cidr\_blocks = [aws\_vpc.main.ipv6\_cidr\_block]
57. }
58. ingress {
59. description = "HTTPS from VPC"
60. from\_port = 80
61. to\_port = 80
62. protocol = "tcp"
63. cidr\_blocks = ["0.0.0.0/0"]
64. }
65. ingress {
66. description = "SSH from VPC"
67. from\_port = 22
68. to\_port = 22
69. protocol = "tcp"
70. cidr\_blocks = ["0.0.0.0/0"]
71. }
72. egress {
73. from\_port = 0
74. to\_port = 0
75. protocol = "-1"
76. cidr\_blocks = ["0.0.0.0/0"]
77. ipv6\_cidr\_blocks = ["::/0"]
78. }
79. tags = {
80. Name = "allow\_web"
81. }
82. #create a network interface 7
83. resource "aws\_network\_interface" "web-server-nic" {
84. subnet\_id = aws\_subnet.subnet-1.id
85. private\_ips = ["10.0.1.50"]
86. security\_groups = [aws\_security\_group.allow\_web.id]
88. }
89. #assign EIP to the network interface created in above step 8
90. resource "aws\_eip" "one" {
91. vpc = true
92. network\_interface = aws\_network\_interface.web-server-nic.id
93. associate\_with\_private\_ip = "10.0.1.50"
94. Depends\_on= [aws\_internet\_gateway.gw]
95. }
96. #creates a n ubantu server & install/enable apache 2. 9
97. Resourses "aws\_instance" "web-server-instance"
98. ami = "ami-image no for ubantu in aws"
99. instance\_type= "t2.micro"
100. availability\_zone = "keep it one u given ibn subnet"
101. Key \_name ="main-key"
102. network\_interface{
103. device\_index = 0
104. Network\_interface\_id = aws\_network\_interface.web-server-nic.id
105. }
106. user\_data = <<-EOF
107. #!/bin/bash
108. Sudo apt update -y
109. Sudo apt install apache2 -y
110. Sudo systemctl start apache2
111. Sudo bash -c ‘echo ur first web server > /var/www/html/idex.htyml’
112. EOF
113. Tags = {
114. Name = "web-server"
115. }
116. }

Save

Go to terminal cd. To the folder in the directory terraform apply🡪yes🡪go to instance and check