LIVE

KNOWLEDGE SHARING SESSION

Using Ansible with Terraform

Gineesh Madapparambath

24 July 2021, 8.00 PM SGT/MYT, 5.30 PM IST

techbeatly

Agenda

- Terraform or Ansible ?
- Infrastructure Management - CMO and FMO
- Terraform Provisioners
- Using Ansible as Terraform Provisioners
- See it in action
- Discussion / Q&A

Ansible

Ansible is an open source automation tool with simple automation language that can perfectly describe IT application environments in Ansible Playbooks.

Terraform

Terraform is an infrastructure as code (IaC) tool that allows you to build, change, and version infrastructure safely and efficiently.

~~Terraform vs Ansible~~ Terraform + Ansible

- Not mutually replaceable 100%
- Use it as a combination instead of comparing

CMO - Current Mode of Operation

Everything Manual

- Get details about cluster/region, Network, Storage etc
- Create VPC (Virtual Private Cloud) or Project
- Create Security Groups, Network Policies
- Create Servers (VM, instance, droplet)
- Configure servers with additional disks
- Install and configure softwares and packages
- Create users and groups
- Changes ? Login to the cloud console, modify the configurations
- Changes ? Login to the servers, modify the configurations.

FMO - Future Mode of Operation

Everything (*) Automated

- Develop your Infrastructure Code
- Deploy as Dev, Staging, Test it, test it, test it
- Deploy to production
- Deploy to DR (Disaster Recovery) on demand
- Changes ? Modify your code and redeploy !

** see supported modules and resources*

techbeatly

Terraform Provisioners

Provisioners can be used to model specific actions on the local machine or on a remote machine in order to prepare servers or other infrastructure objects for service.

```
resource "aws_instance" "web" {
  # ...

  provisioner "local-exec" {
    command = "echo The server's IP address is ${self.private_ip}"
  }
}
```

Destroy Time Provisioners

Provisioner will run when the resource it is defined within is destroyed.

```
resource "aws_instance" "web" {
  # ...

  provisioner "local-exec" {
    when    = destroy
    command = "echo 'Destroy-time provisioner'"
  }
}
```

What if provisioner failed ?

- **continue** - Ignore the error and continue with creation or destruction.
- **fail** - Raise an error and stop applying (default)

```
resource "aws_instance" "web" {
  # ...

  provisioner "local-exec" {
    command    = "echo The server's IP address is ${self.private_ip}"
    on_failure = continue
  }
}
```

How to connect to the target machine ?

- Connect using ssh/winrm, username-password or ssh keys

```
# Copies the file as the root user using SSH
provisioner "file" {
  source      = "conf/myapp.conf"
  destination = "/etc/myapp.conf"

  connection {
    type     = "ssh"
    user     = "root"
    password = "${var.root_password}"
    host     = "${var.host}"
  }
}
```


terraform.io/docs/language/resources/provisioners/connection.html

How to use remote-exec

- **inline**
- **script**
- **scripts**

```
resource "aws_instance" "web" {
  # ...

  provisioner "remote-exec" {
    inline = [
      "puppet apply",
      "consul join ${aws_instance.web.private_ip}",
    ]
  }
}
```

techbeatly

terraform.io/docs/language/resources/provisioners/remote-exec.html

Let’s combine Ansible with Terraform

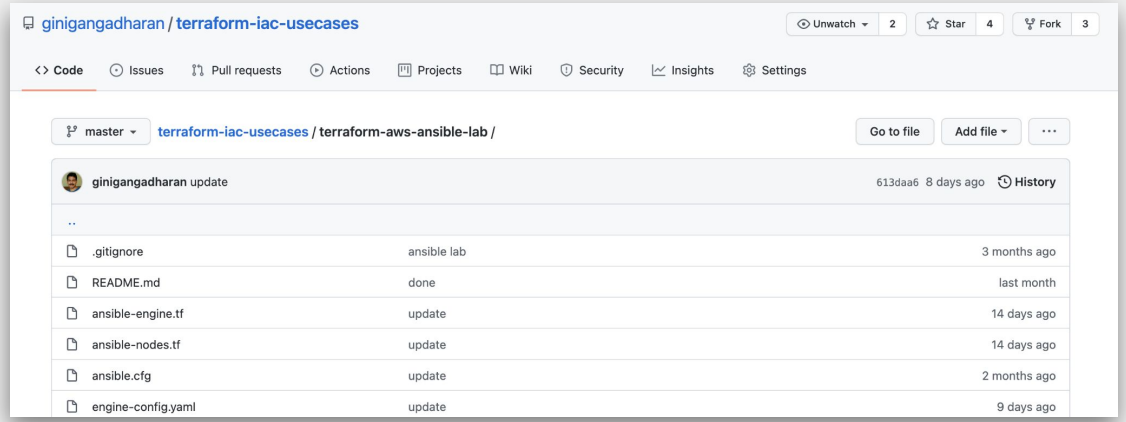
- Calling an Ansible playbook inside an ec2 instance

```
provisioner "remote-exec" {
  inline = [
    "sleep 120; ansible-playbook engine-config.yaml"
  ]

  connection {
    type = "ssh"
    user = "ec2-user"
    private_key = file(pathexpand(var.ssh_key_pair))
    host = self.public_ip
  }
}
```

Let’s explore the combination

- Terraform configuration with multiple provisioners



techbeatly

github.com/ginigangadharan/terraform-iac-usecases/

Questions & Feedbacks

Ansible FREE Course : techbeatly.com/ansible-course
Ansible Real Life : techbeatly.com/ansible-real-life

SCAN ME

feedback

techbeatly.com /techbeatly t.me/techbeatly

techbeatly