Managing your cloud infrastructure using CI/CD with Terraform and Ansible.

Abubakar Siddiq Ango
GitLab BV.
@sarki247, abuango.me
Me

- Based in Bauchi, Nigeria
- Support Engineer with GitLab BV
- Organizer, GDG Bauchi & DigitalOcean Bauchi
- Executive Director, Uplift Nigeria (uplift.ng)
- @sarki247 (Twitter & Instagram), @abuango (Everywhere else)
Automation!!!

We love automation and want to automate everything!
Why we need Automation

- Standards
- Best practices
- Consistency
- Efficiency
DevOps is more than Automation

DevOps requires cultural norms and an architecture that allows for the shared goals to be achieved throughout the IT value stream. This goes far beyond just automation.

Source: https://dzone.com/articles/the-difference-between-ci-pipelines-and-devops-ass
Infrastructure as Code - IaC

Using the same tools developers use, we can treat infrastructure as code allowing us to:

- Collaborate
- Test
- Version
- Document the state of Infrastructure

You can manage an entire data centre & quickly recover from any incident by simply redeploying the code or reverting to an earlier commit.
IaC - Use Cases

- Versioned Infrastructure Deployment
- Rapid Onboarding of new staff
- Rapid Disaster Recovery
- Customer Support
- Cost Control
- Quality Assurance
Ansible & Terraform

Terraform is a great tool for building infrastructure in the cloud. Ansible is an agentless (and serverless) configuration management tool. A common use case is to build servers with Terraform, and have Ansible configure them.
Ansible

- Procedural
- Configuration Management
- Mutable Infrastructure
- Flexible
- Stateless
- Agentless
- Composed of Hosts, Playbooks, Roles, Tasks
- galaxy.ansible.com
- Modules
Ansible

- digital_ocean:
  state: present
  command: droplet
  name: mydroplet
  api_token: XXX
  size_id: 2gb
  region_id: ams2
  image_id: fedora-19-x64
  wait_timeout: 500
  register: my_droplet

- debug:
  msg: "ID is {{ my_droplet.droplet.id }}"

- debug:
  msg: "IP is {{ my_droplet.droplet.ip_address }}"

---

- hosts: webservers
  vars:
    http_port: 80
    max_clients: 200
    remote_user: root
  tasks:
    - name: ensure apache is at the latest version
      yum: name=httpd state=latest
    - name: write the apache config file
      template: src=/srv/httpd.j2
      dest=/etc/httpd.conf
    - name: ensure apache is running (and enable it at boot)
      service: name=httpd state=started enabled=yes
  handlers:
    - name: restart apache
      service: name=httpd state=restarted
Ansible Demo

- Provision a droplet and install a webserver
- Using third party roles to make a HA (High Availability) deployment

Source: https://gitlab.com/do-lagos-meetup/ansible-demo
Terraform

- Declarative Syntax
- Infrastructure Orchestration
- Stateful
- Immutable Infrastructure
- Providers & Provisioners as Plugins
- Client-only
resource "digitalocean_droplet" "web" {
  image    = "ubuntu-14-04-x64"
  name     = "web-1"
  region   = "nyc2"
  size     = "512mb"
}

output "Public ip web" {
  value = "${digitalocean_droplet.web.ipv4_address}"}

Terraform Provisioners

- Local-exec: Run commands on terraform host
- Remote-exec: Run commands on the provisioned server
  - Authentication with Username/Password
  - Authentication with SSH Key
Terraform Demo

- Deploy a web server
- Deploy a High Availability Server

Source: https://gitlab.com/do-lagos-meetup/terraform-demo
CI/CD

**Continuous Integration**: Automated Validation

**Continuous Delivery**: Rapid automated releases

**Continuous Deployment**: Rapid automated deployments to production

#AutoDevOps #IdeaToProduction
CI/CD + IaC

- Code
  - Scripts: Ansible, Terraform, etc.

- Git
  - git push
  - Trigger CI
  - If Test Fails, Notify

- Test
  - Create PR/MR
  - Make Changes if needed

- Review

- Deploy
  - Merge & Trigger CD
  - AWS / GCP / DO
Terraform + Ansible

No first-class Ansible provisioner for Terraform but you can use `local-exec` Provisioner to either echo IPs to an inventory or run `ansible-playbook` directly.

```hcl
provisioner "local-exec" {
  command = "echo ${digitalocean_droplet.web.public_ip} >> web_ips.txt"
}
```

Or

```hcl
provisioner "local-exec" {
  command = "sleep 120; ANSIBLE_HOST_KEY_CHECKING=False ansible-playbook -u root --private-key ./deployer.pem -i '${digitalocean_droplet.web.public_ip},' master.yml"
}
```
Đọ̣my / Nagode / Daalụ