gruppen-arbeit.md

Ansible: Gruppenaufgabe

Szenario

- Dein Team hat eine geniale Millionen Euro Geschäftsidee für das nächste Unicorn Startup.
- Um Investoren einfacher überzeugen zu können, beschließt ihr eine Website zu erstellen.
- Das verwendete Setup soll einfach zu warten sein, daher soll alles im Sinne von Infrastructure as Code aufgebaut sein

Anforderungen

1. Webserver installiert (nginx, apache, ...)
2. Webroot Folder erzeugt (z.B. `/var/www/html`)

Überlegungen

Schritte

1. Vorbereitungen

1.1 AWS Profile

- Stelle sicher, dass dein AWS CLI Profile richtig konfiguriert ist.

```
aws configure list-profiles
```

- Falls dein techstarter Profil nicht existiert, nutze `aws configure sso`

1.2 Ansible CLI

- Stelle sicher, dass die Ansible CLI installiert ist

```
ansible --version
```

- Sollte ansible nicht anstalliert sein, folge dieser Anleitung:
  - Link
  - TLDR: Unter Linux nutze `python3 -m pip install --user ansible`

1.3 Projekt Ordner

- Erstelle einen neuen Ordner und öffne deinen Texteditor

```
mkdir ansible-intro && cd ansible-intro && code .
```

1.4 SSH Key-Pair

- Erstelle ein neues SSH Key-Pair.
- Merke dir den Pfad zu beiden Keys (Public + Private)

```
ssh-keygen -t rsa
```

1.5. AWS EC2 - Terraform

- Kopiere die Datei `infrastructure.tf` in dein Repo und passe diese entsprechend an:
  - AWS Profile, z.B. `techstarter`
  - Pfad zu deinem **Public SSH Key**
- Erstelle die Infrastruktur

```
terraform init
terraform plan
terraform apply
```

- Speichere dir die IP Addresse, die am Ende des `terraform apply` Commands erscheint ab
- Verbinde dich via SSH mit der EC2 Instanz

```
ssh -i </pfad/zu/deinem/ssh-private-key> ec2-user@<server_ip>
```

2. Ansible Setup

2.1 Inventory

- Erstelle die host Datei um deine Server Infrastruktur für Ansible zu definieren

```
touch hosts
```

- Füge folgenden Inhalt in die neue Datei und passe folgende Felder an:
  - EC2_PUBLIC_IP
  - Pfad zu dinem Private Key

```
host1 ansible_host=DEINE_EC2_PUBLIC_IP ansible_user=ec2-user ansible_port=22 ansible_ssh_private_key_file=/pf
```

- Um dein Inventory zu testen, führe das ping Modul aus

```
ansible -i hosts all -m ping
```

2.2 Playbook

- Jetzt können wir beginnen das Playbook zu konfigurieren, welche die EC2 Instanz versioniert
- Erstelle die Datei `playbook.yaml` und füge folgenden Inhalt ein:

```yaml
---
- name: Intro to Ansible Playbooks
  hosts: all

  tasks:
  - name: Add the user 'foo'
    ansible.builtin.user:
      name: foo
    become: yes
    become_method: sudo
  - name: upgrade all packages
    yum:
      name: '*'
      state: latest
    become: yes
```

- Diese Playbook erstellt bis jetzt einen neuen User `foo` (mit sudo Rechten) und installiert alle Paketupdates

```
ansible-playbook -i hosts playbook.yaml
```

2.3 Installation der Anwendung

- Ergänze in den folgenden Schritten die `playbook.yaml` und füge weitere Tasks hinzu

2.3.1 Texteditor

- Schaue dir folgenden Teil der Dokumentation genau an: Link
- **Installiere folgende Pakete**:
  - vim
  - nano

2.3.2 Web Root und index.hmtl

- Informiere dich über den von deinem Webserver verwendeten Web Root (Ordner wo html Datein liegen)
- Erstelle diesen Ordner mittels Ansible
- Schreibe eine index.html Datei in deinem Repo und kopiere diese mit Ansible auf deinen Webserver

2.3.3 Web Server

- Installiere ebenfalls über Ansible einen der folgenden Web Server:
  - nginx
  - Apache
- Erstelle in deinem Repository die Konfigationsdatei für den Web Server für den du dich entschieden hast
- Stelle sicher, dass dein Web Root Verzeichnis in deiner Konfiguration hinterlegt ist.
- Kopiere diese Datei auf den Server an den richtigen Ort und nutze dafür ebenfalls Ansible.

3. Test

Rufe die IP Addresse jetzt über deinen Browser auf, **stelle sicher, dass du HTTP und nicht HTTPS verwendest!!!**

Wird dir deine index.html angezeigt?

4. Freiwillig

Für alle, die noch tiefer einsteigen wollen:

Um den Automationsaufwand so minimal wie möglich zu erhalten, soll das Prinzip `GitOps` verwendet werden. - Es gibt ein zentrales Git Repo mit der Konfiguration der Infrastruktur - Der Server führt eine lokale Kopie dieses Repos - In regelmäßigen Zeitschritten wird das Repo geupdatet und die Konfiguration ausgeführt

4.1 Git Repo

- Erstelle ein öffentliches git Repository mit den aktuellen Datein
- Stelle sicher, dass keine Secrets in diesem Repo gespeichert sind!

4.2 Update Script

- Erstelle ein Bash Script in dem Repo was foglende Schritte ausführt:
  i. Überprüfen ob lokale Kopie des Repos existiert 1.1 Falls nicht, erstellen 1.2 Falls ja, updaten (git pull)
  ii. Die aktuelle Version der `playbook.yaml` auf dem localhost ausführen `ansible-playbook`
  iii. Logs in eine persistierte Datei schreiben

4.3 Cron Job via Ansible definieren

Ändere das Playbook erneut ab:

- Das gerade erstellte Script soll an eine persistierte Stelle im OS geschrieben werden und ausführbar gemacht werden
- Ein Cron Job soll einmal pro Stunde laufen und das erstellte Script ausführen