# laC avec Terraform / OpenTofu et Ansible

# Terraform / OpenTofu : Qu'est-ce que c'est ?

#### Terraform est:

- un outil open-source d'Infrastructure as Code (IaC)
- créé par HashiCorp en 2014
- permet de définir et de gérer une infrastructure en utilisant un langage déclaratif.
- changement de licence en août 2023 : MPL > BUSL
- OpenTofu : fork de la communauté supporté par la Linux Foundation

# Principales caractéristiques

- Syntaxe déclarative pour définir l'infrastructure
- Prise en charge de multiples fournisseurs et services cloud
- Compatible avec le contrôle de version et la collaboration
- Planification et application des changements de manière prévisible

## Pourquoi utiliser Terraform?

Terraform permet aux équipes DevOps d'automatiser le provisionnement de l'infrastructure, d'améliorer la cohérence et de gérer efficacement des environnements multi-cloud complexes.

Il favorise les pratiques d'infrastructure as code, améliorant l'évolutivité et réduisant les erreurs humaines.

# Autres outils de déploiement

- OpenTofu : Un fork communautaire de Terraform, maintenu par la Linux Foundation
- Pulumi : Outil IaC utilisant des langages de programmation courants (Javascript, Python, Go, C#, YAML)
- AWS Cloud Formation
- Azure Resource Manager
- Google Cloud Deployment Manager

# Ansible : Qu'est-ce que c'est ?

#### Ansible est:

- un outil d'automatisation IT open-source
- simplifie la gestion de configuration
- le déploiement d'applications
- l'orchestration des tâches.

# Caractéristiques clés d'Ansible

- Sans agent (Agentless)
- Utilise SSH pour la communication
- Configuration basée sur YAML
- Facile à apprendre et à utiliser
- Extensible avec des modules

# Composants principaux

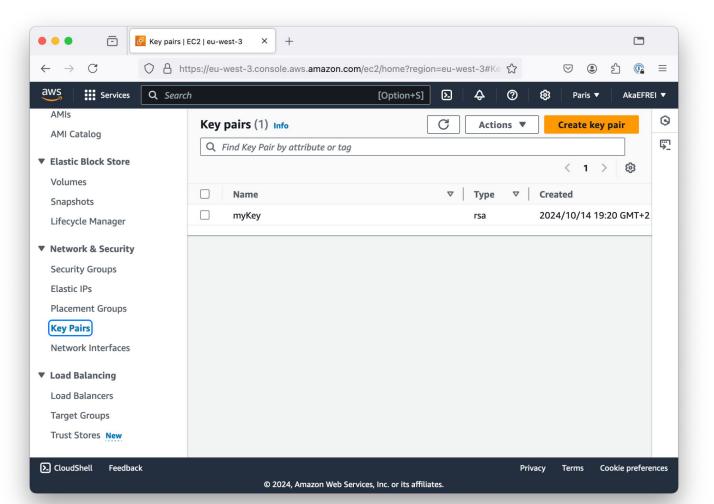
- Playbooks: Scripts d'automatisation
- Inventaire: Liste des hôtes gérés
- Modules: Unités de travail dans Ansible
- Rôles: Réutilisation et organisation du code

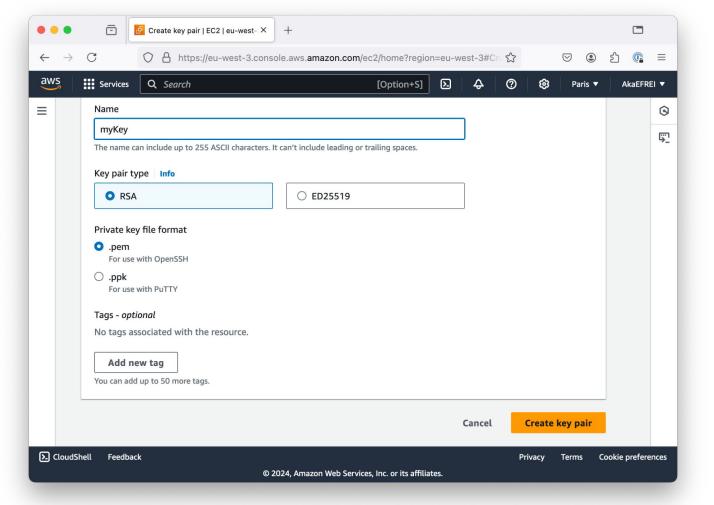
#### Etape 0:

- Créer un compte AWS avec votre adresse EFREI
- Les comptes bénéficient par défaut du "Free Tier" pendant 1 an qui vous permet d'avoir accès un certain nombre de ressources gratuitement chaque mois. Le détail est sur la page.

#### Etape 0.1:

- Créer une clé SSH pour les instances qui sera utilisée pour s'y connecter : <a href="https://eu-west-3.console.aws.amazon.com/ec2/home?region=eu-west-3#Key">https://eu-west-3.console.aws.amazon.com/ec2/home?region=eu-west-3#Key</a> Pairs:





Après cette étape, un fichier "myKey.pem" sera téléchargé automatiquement.

Il vous servira à vous connecter via SSH aux instances que vous allez créer.

ATTENTION aux droits d'accès à ce fichier, pour l'utiliser correctement, vous devrez utiliser la commande suivante :

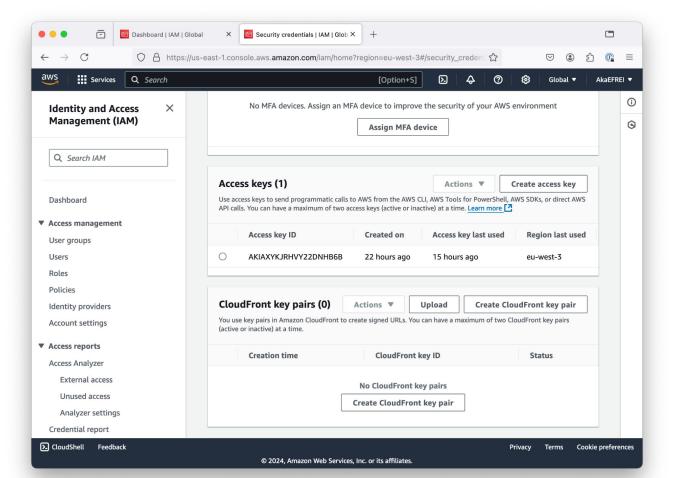
chmod 600 myKey.pem

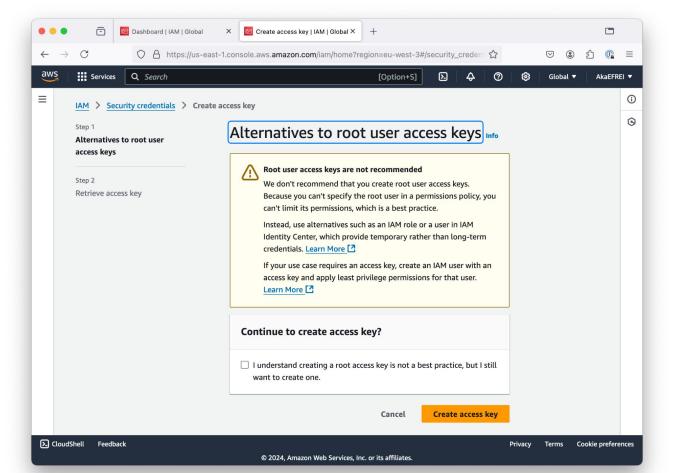
Ce fichier sera utilisé par Ansible plus tard dans la démonstration

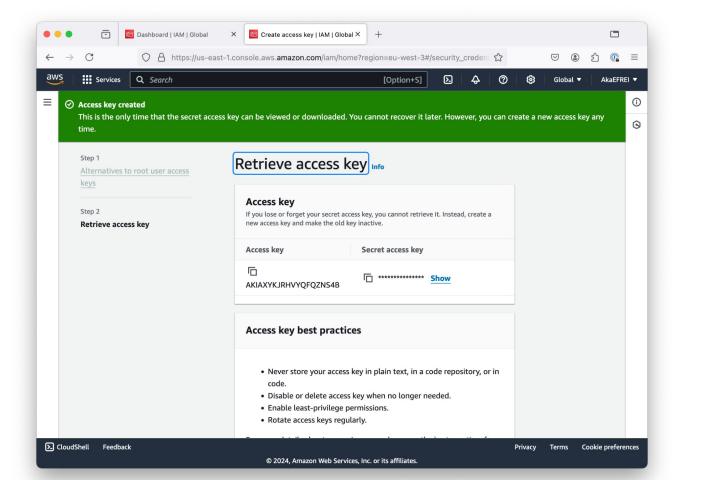
#### Etape 0.2:

 Créer une ACCESS KEY qui permettra à Terraform de se connecter au compte AWS :

https://us-east-1.console.aws.amazon.com/iam/home?region=eu-west-3#/sec urity credentials







Créer un fichier credentials selon le format suivant :

```
[default]
aws_access_key_id = <ACCESSKEY>
aws_secret_access_key = <SECRETKEY>
```

Selon le système que vous utilisez, le fichier est à mettre dans un répertoire différent : <a href="https://docs.aws.amazon.com/sdkref/latest/guide/file-location.html">https://docs.aws.amazon.com/sdkref/latest/guide/file-location.html</a>

Etape 0.3

Installer Terraform (ou OpenTofu) et Ansible

#### Etape 1

On utilise Terraform pour créer un container nommé tutorial à partir de l'image nginx:1.27

#### Les commandes Terraform utilisées :

```
terraform init
terraform plan
terraform apply --auto-approve
terraform destroy --auto-approve
```

Etape 2

On utilise Terraform pour créer 2 containers :

- le premier nommé php-fpm à partir de l'image php:8.3-fpm
- le premier nommé nginx à partir de l'image nginx:1.27

Les 2 containers communiquent à la manière de ce qu'on avait vu précédemment avec Docker : NGINX interroge PHP pour délivrer le contenu de l'exécution de la fonction phpinfo()

Etape 3

On crée maintenant une instance EC2 sur AWS.

La commande en haut du fichier permet de récupérer l'AMI utilisé en paramètre à la création de la ressource

key\_name contient le nom de la clé créée au préalable et qui permettra ensuite de se connecter en SSH par la suite

Le type d'instance t3.micro correspond au type d'instance gratuite dans le cadre de l'utilisation avec AWS Free Tier

Etape 4

On rajoute un output à Terraform qui permet de récupérer l'adresse IP de l'instance créée et qui permettra par la suite de s'y connecter

Etape 5

L'identifiant d'AMI est récupéré directement par une requête de type data dans le fichier Terraform au lieu de spécifier un identifiant d'ami "en dur".

L'AMI changera donc dans le temps au fur et à mesure des mises à jour.

Etape 6

Un nouveau type de ressource est ajouté : un security group nommé "allow\_ssh"

Dans celui-ci, on ouvre le port 22 (SSH) à la connexion depuis l'extérieur et nous permet donc de nous connecter à l'instance avec la commande suivante :

ssh -i myKey.pem ubuntu@<ADRESSE-IP>

#### Etape 7

Dans Terraform, on ajoute maintenant un autre security group nommé "allow\_http\_s" qui ouvre les ports 80 et 443, respectivement HTTP et HTTPS

#### On ajoute maintenant Ansible:

- Le fichier playbook.yml contient la liste des tâches qu'il va effectuer sur les instances
- le fichier inventory.ini contient la liste des instances auxquelles il va se connecter (penser à mettre à jour l'adresse IP à partir de celle qui a été récupérée dans les étapes précédentes)
- La commande: ansible-playbook -i inventory.ini playbook.yml

Etape 8

Nouvelle tâche Ansible:

Remplacer le fichier index.html par défaut par un fichier contenant le message "Bonjour le Monde"

Etape 9

Mise à jour de la tâche Ansible :

Le fichier index.html n'est plus mis à jour depuis Ansible mais par la copie d'un fichier présent localement.

Etape 10

L'inventaire d'Ansible est maintenant généré dynamiquement :

Il se connecte à l'API d'AWS pour récupérer la liste des instances et alimenter automatiquement l'inventaire :

ansible-playbook -i aws\_ec2.yml playbook.yml