

TERRAFORM: LES BASES POUR DEVOPS

Par Dirane TAFEN

Plan

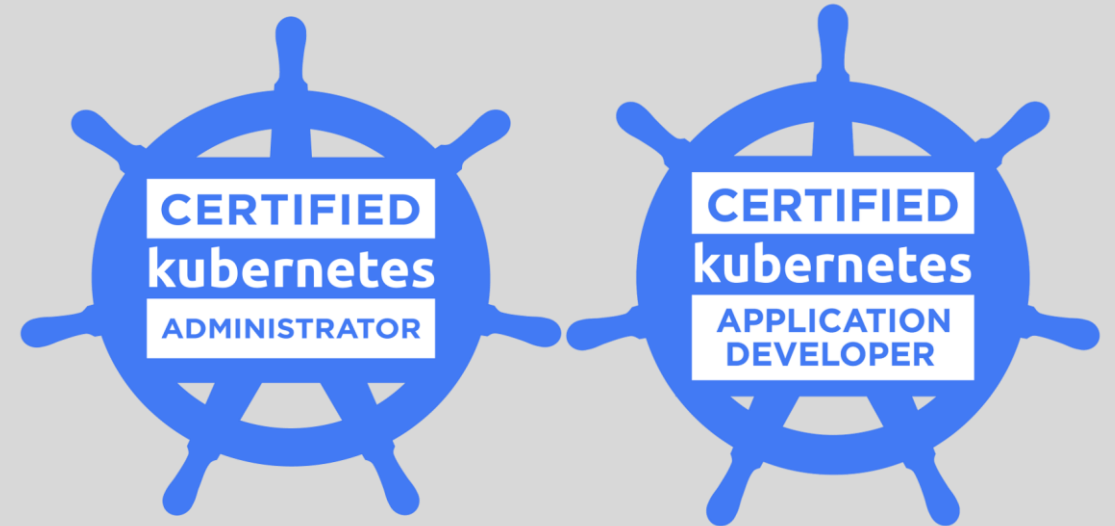
- Présentation du formateur
- Introduction au DevOps et IaC
- Terraform
- Déployez vos premières ressources
- Rendez vos déploiements dynamique
- Terraform Provisioners
- Remote management
- Module
- Mini-projet

Plan

- Présentation du formateur
- Introduction au DevOps et IaC
- Terraform
- Déployez vos premières ressources
- Rendez vos déploiements dynamique
- Terraform Provisioners
- Remote management
- Module
- Mini-projet

Présentation du formateur

- Dirane TAFEN (formateur et consultant DevOps)
- Capgemini
- Sogeti
- ATOS
- BULL
- AIRBUS
- ENEDIS



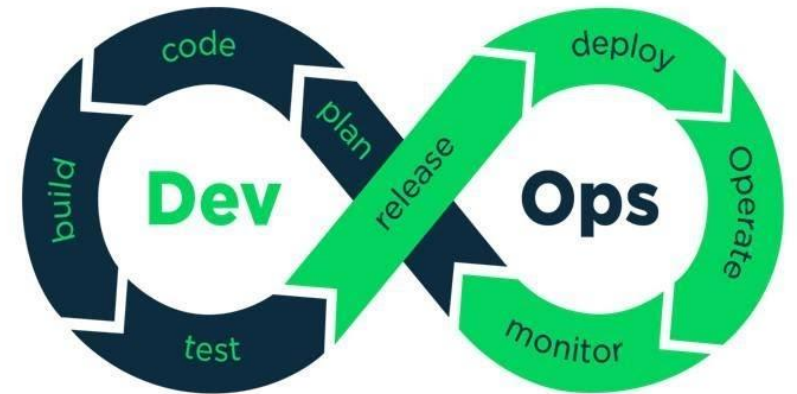
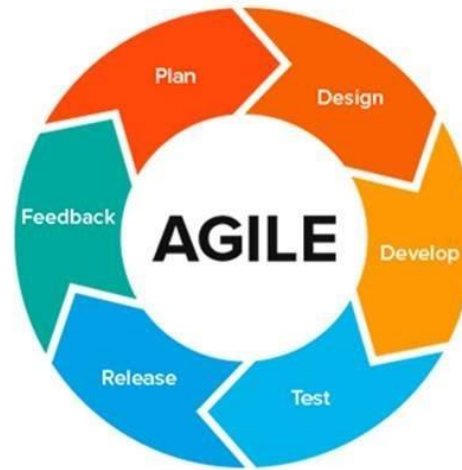
Plan

- Présentation du formateur
- Introduction au DevOps et IaC
- Terraform
- Déployez vos premières ressources
- Rendez vos déploiements dynamique
- Terraform Provisioners
- Remote management
- Module
- Mini-projet

Introduction au DevOps et IaC (1/3): Le DevOps

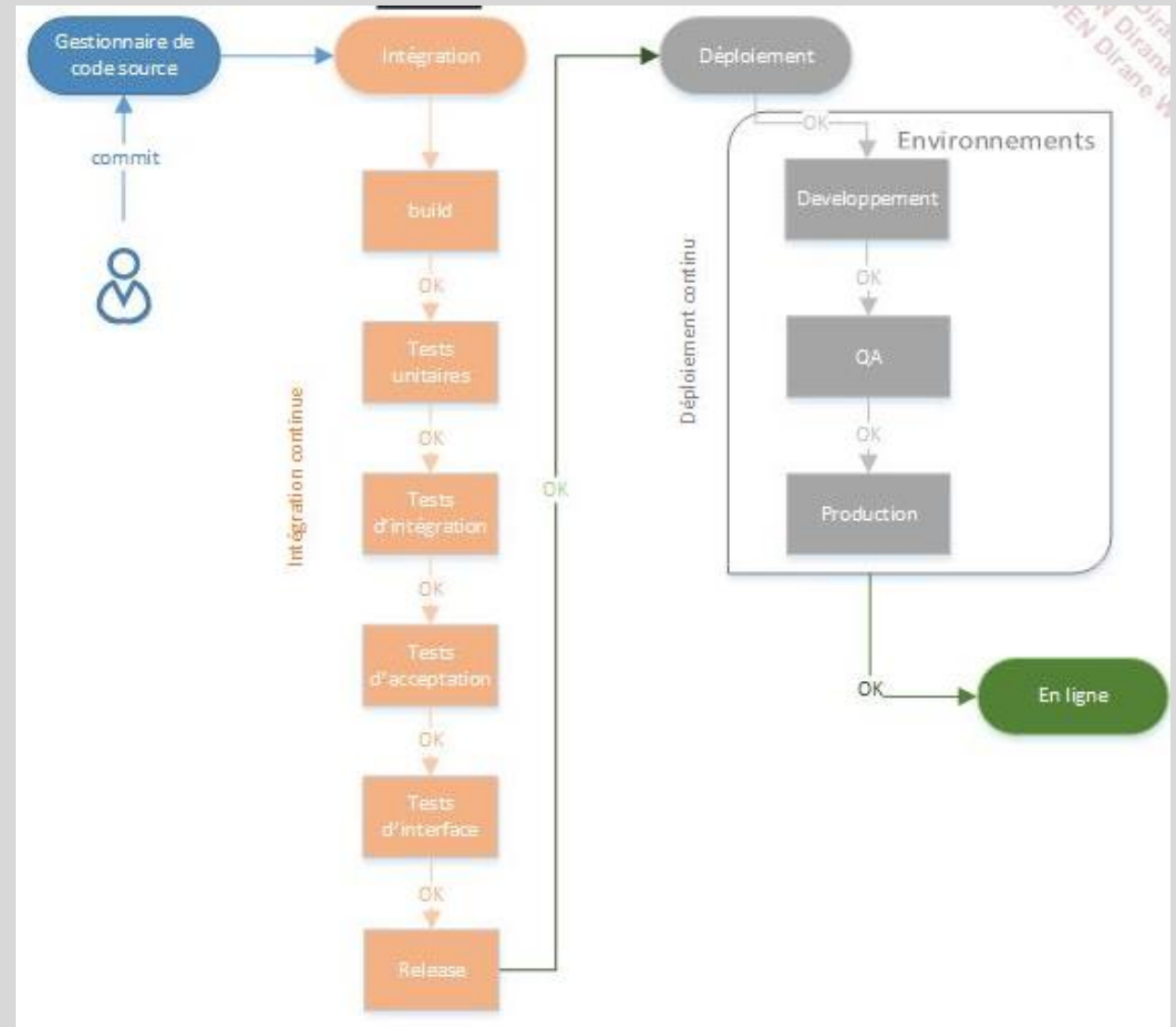
- Agile: méthode de développement
- DevOps: agilité dans le Dev et l'Ops = CI + CD

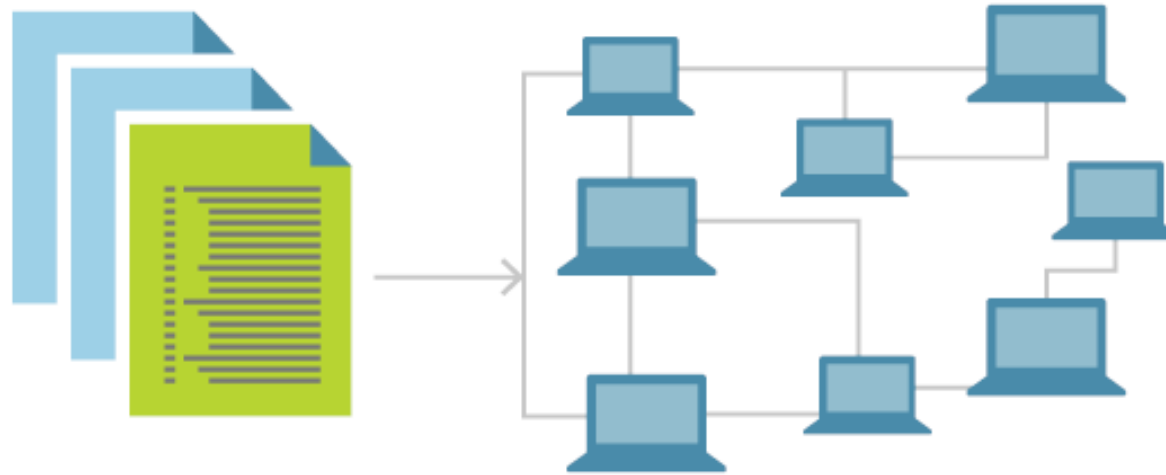
Agile vs. DevOps



Introduction au DevOps et IaC (2/3): CI/CD

- Intégration en continu
- Test en continu
- Déploiement en continu



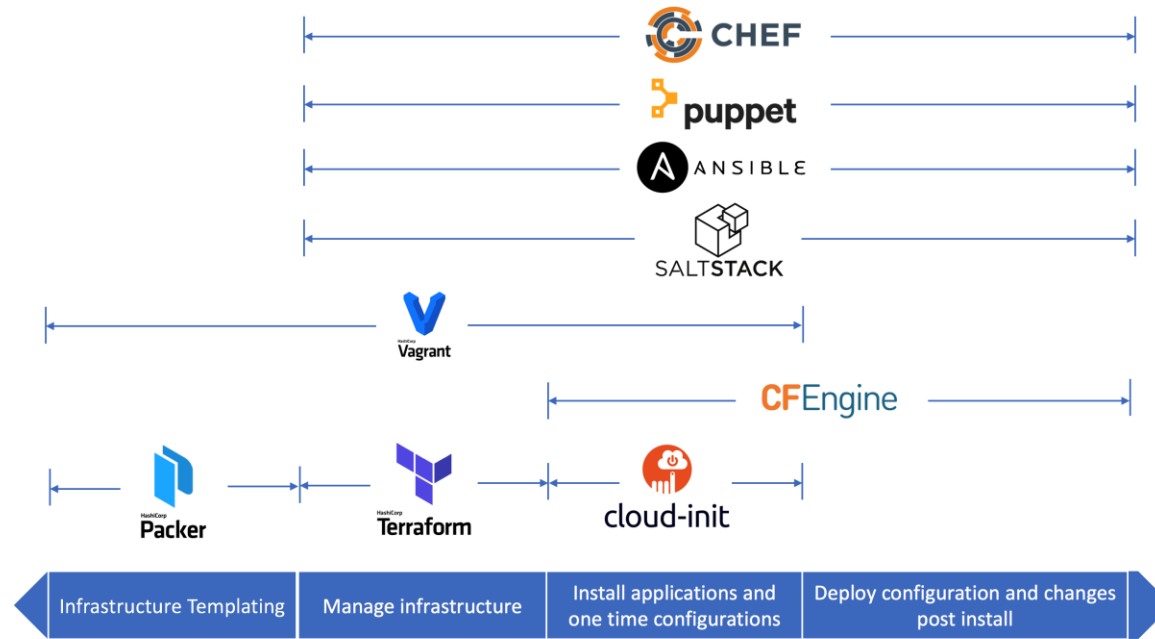


Introduction au DevOps et IaC (3/3): IaC

- Réutilisation
- Evolutivité
- Collaboration

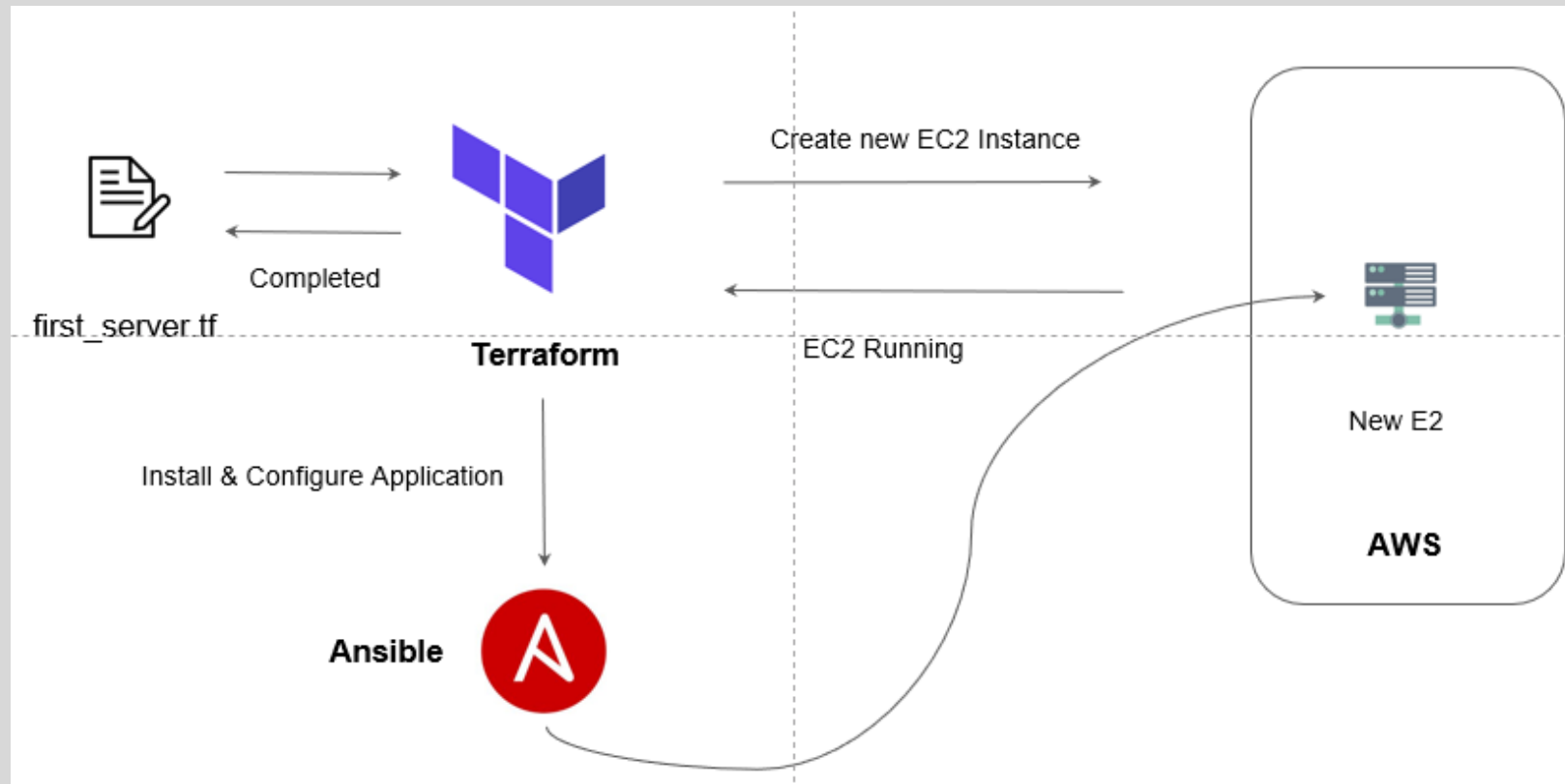
Plan

- Présentation du formateur
- Introduction au DevOps et IaC
- Terraform
- Déployez vos premières ressources
- Rendez vos déploiements dynamique
- Terraform Provisioners
- Remote management
- Module
- Mini-projet



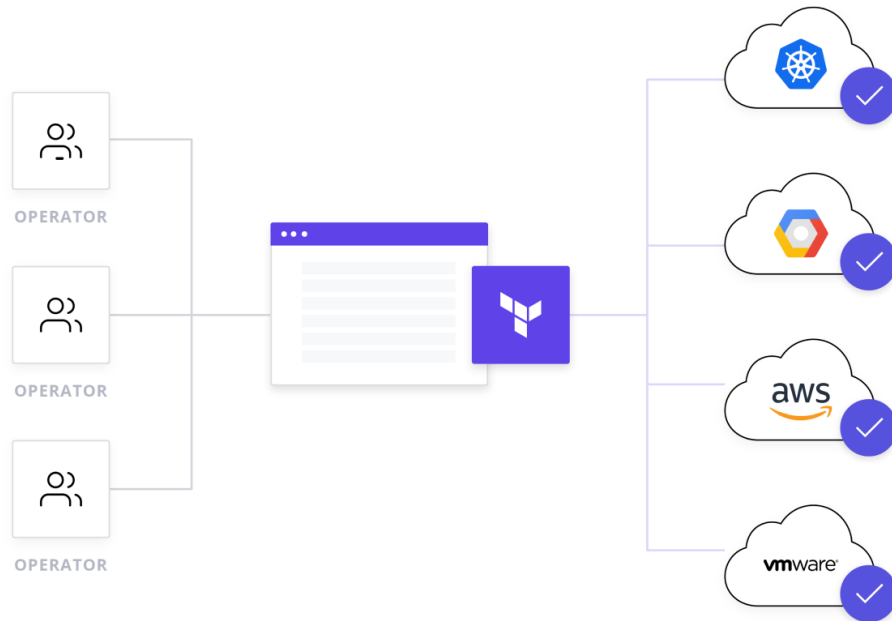
TERRAFORM (1/4): CHALLENGERS

Terraform (2/4): IaC et CM complémentaires



Terraform (3/4): Avantages

- Multi-Cloud (de multiple providers)
- Gratuit
- Langage facile à lire
- Extensible
- S'intègre avec les outils de Configuration Management



Terraform (4/4): Installation

Windows

macOS

Linux

FreeBSD

OpenBSD

Solaris

TP-0: Compte AWS et IDE


- Créez un compte AWS gratuit
- Protégez votre compte root avec un mot de passe fort
- Créez un compte nominatif avec les droits full admin et qui vous permettra de déployer les ressources
- Installez un IDE, par exemple ATOM et installez un plugin terraform pour vous faciliter la correction syntaxique
- Vous êtes prêt à installer terraform !

TP-I: Installez terraform

- Installez Terraform en récupérant le binaire via le lien suivant: <https://www.terraform.io/downloads.html>
- Si vous êtes sous windows, copiez le binaire dans un dossier de votre disque dur system par exemple C:\terraform\ , ensuite il vous faudra rajouter le repertoire precedent dans le PATH de votre système d'exploitation
- Si vous êtes sous linux, vous pouvez le déplacer dans /usr/bin/ après l'avoir rendu executable
- Pour verifiez l'installation, veuillez juste entrez la commande terraform et l'aide apparaitra !

Plan

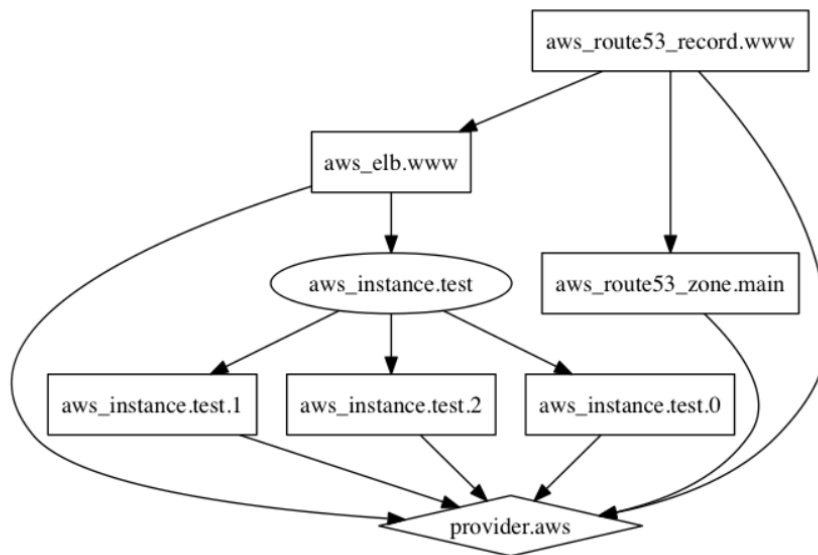
- Présentation du formateur
- Introduction au DevOps et IaC
- Terraform
- Déployez vos premières ressources
- Rendez vos déploiements dynamique
- Terraform Provisioners
- Remote management
- Module
- Mini-projet



DÉPLOYEZ VOS PREMIÈRES RESSOURCES (1/5): EC2

```
provider "aws" {  
    region      = "us-west-2"  
    access_key  = "PUT-YOUR-ACCESS-KEY-HERE"  
    secret_key  = "PUT-YOUR-SECRET-KEY-HERE"  
}  
  
resource "aws_instance" "myec2" {  
    ami = "ami-082b5a644766e0e6f"  
    instance_type = "t2.micro"  
}
```

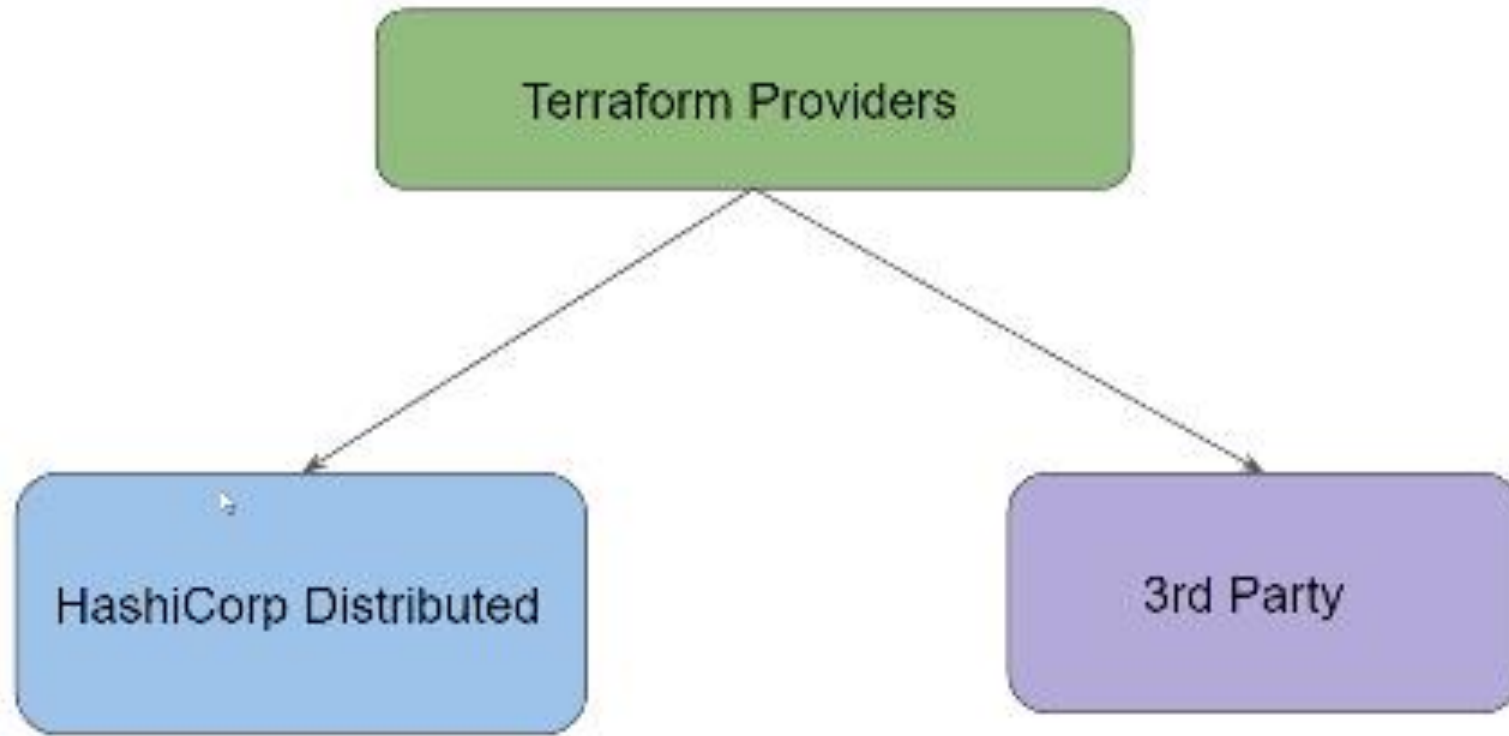
DÉPLOYEZ VOS PREMIÈRES RESSOURCES (2/5): RESOURCE & PROVIDERS

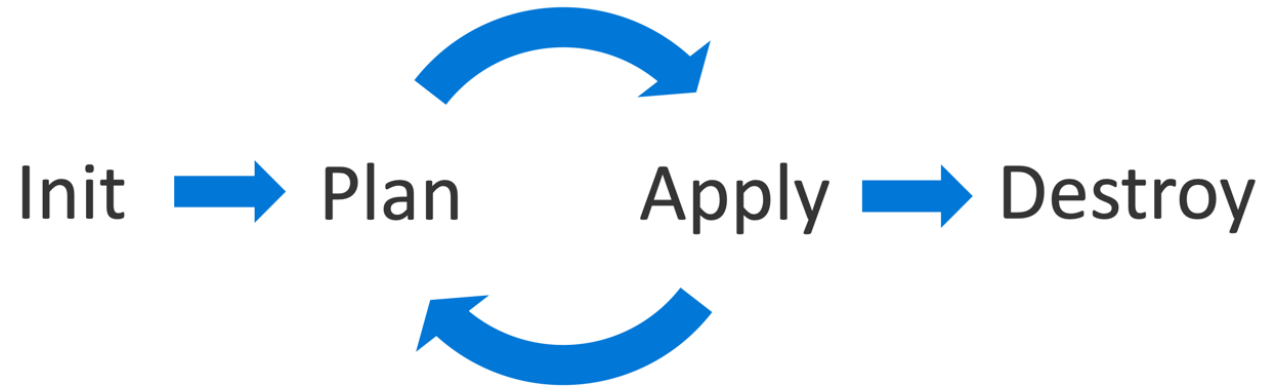


- Providers: fourni le plugin permettant de communiquer avec la plateforme à provisionner (AWS, GCP, AZURE, Digital Ocean, OVH ...), ils sont versionnés et mis à jour constamment (attention aux mises à jour)
- Ressource: Représente le type d'objet à créer sur la plateforme à provisionner

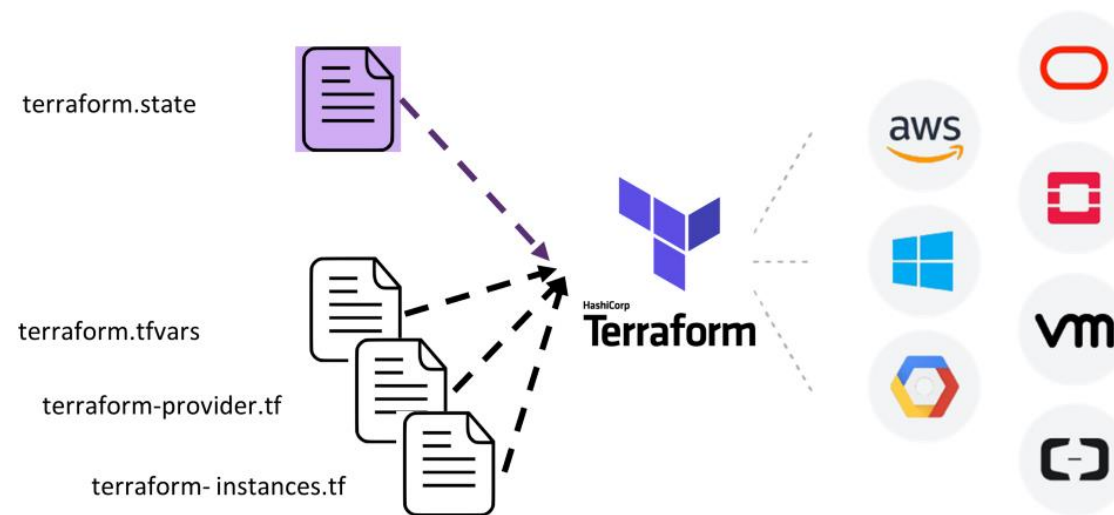
DÉPLOYEZ VOS PREMIÈRES RESSOURCES

(3/5):TYPE DE PROVIDER





**DÉPLOYEZ VOS PREMIÈRES RESSOURCES
(4/5): COMMANDES DE BASE**



DÉPLOYEZ VOS PREMIÈRES RESSOURCES (5/5): TFSTATE

- État de l'infra
- Sécurité (pas dans GIT)
- Remote state

TP-2: Déployez votre Ressource AWS avec terraform

- Récupérez le secret et access key de votre compte (dans les paramètres sécurité de votre compte dans IAM)
- Créez un paire de clé dans EC2 et nommez cette clé devops-<votre prenom>, un fichier devops-<votre prenom>.pem sera téléchargé (conservez la jalousement)
- Créez un fichier ec2.tf dans un répertoire nommé tp-2
- Renseignez les informations permettant de créer une VM avec l'image centos suivante: centos7-minimal-v20190919.0.0 (ami-0083662ba17882949)
- **ATTENTION** nous travaillerons uniquement dans la region US East (N.Virginia)us-east-1 dans **toute cette formation**
- Vérifiez que votre instance est bien créée et observez le contenu de fichier tfstate
- Modifiez le fichier ec2.tf afin d'y inclure le tag de votre instance : "Name: ec2-<votre prenom>"
- Appliquez la modification et constatez les changements apportés ainsi que dans le fichier tfstate
- Supprimez votre ec2


Plan

- Présentation du formateur
- Introduction au DevOps et IaC
- Terraform
- Déployez vos premières ressources
- Rendez vos déploiements dynamique
- Terraform Provisioners
- Remote management
- Module
- Mini-projet

```
provider "aws" {  
  region      = "us-west-2"  
  access_key  = "PUT-YOUR-ACCESS-KEY-HERE"  
  secret_key  = "PUT-YOUR-SECRET-KEY-HERE"  
}  
  
resource "aws_eip" "lb" {  
  vpc      = true  
}  
  
output "eip" {  
  value = aws_eip.lb  
}  
  
resource "aws_s3_bucket" "mys3" {  
  bucket = "kplabs-attribute-demo-001"  
}  
  
output "mys3bucket" {  
  value = aws_s3_bucket.mys3  
}
```

Rendez vos déploiements dynamique (1/7):Attribut et output

- Customization de votre déploiement -> attribut
- Réparation d'informations dynamiques -> output



RENDEZ VOS DÉPLOIEMENTS DYNAMIQUE (2/7): RÉFÉRENCEZ DES RESSOURCES

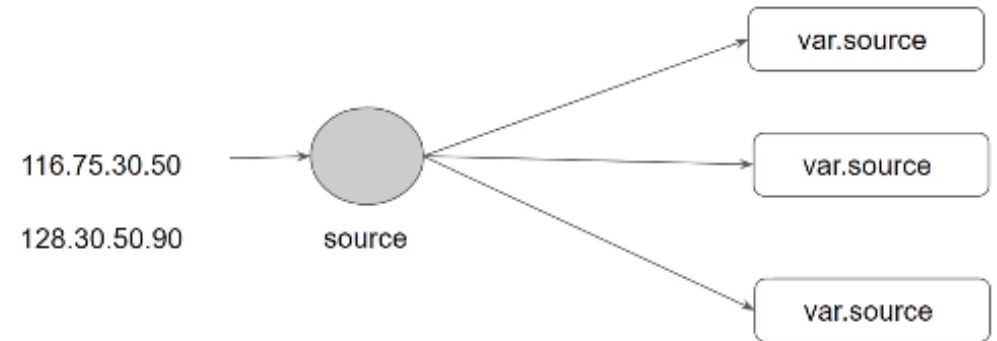
```
provider "aws" {  
    region      = "us-west-2"  
    access_key  = "PUT-YOUR-ACCESS-KEY-HERE"  
    secret_key  = "PUT-YOUR-SECRET-KEY-HERE"  
}  
  
resource "aws_instance" "myec2" {  
    ami = "ami-082b5a644766e0e6f"  
    instance_type = "t2.micro"  
}  
  
resource "aws_eip" "lb" {  
    vpc      = true  
}  
  
resource "aws_eip_association" "eip_assoc" {  
    instance_id = aws_instance.myec2.id  
    allocation_id = aws_eip.lb.id  
}
```

varsdemo.tf

```
resource "aws_security_group" "var_demo" {  
  name      = "kplabs-variables"  
  
  ingress {  
    from_port = 443  
    to_port   = 443  
    protocol  = "tcp"  
    cidr_blocks = [var.vpn_ip]  
  }  
  
  ingress {  
    from_port = 80  
    to_port   = 80  
    protocol  = "tcp"  
    cidr_blocks = [var.vpn_ip]  
  }  
  
  ingress {  
    from_port = 53  
    to_port   = 53  
    protocol  = "tcp"  
    cidr_blocks = [var.vpn_ip]  
  }  
}
```

variables.tf

```
variable "vpn_ip" {  
  default = "116.50.30.50/32"  
}
```



RENDEZ VOS DÉPLOIEMENTS DYNAMIQUE (3/7): VARIABLES (PARTIE I)



RENDEZ VOS DÉPLOIEMENTS DYNAMIQUE (4/7): VARIABLE (PARTIE 2)

```
provider "aws" {  
  region      = "us-west-2"  
  access_key  = "YOUR-ACCESS-KEY"  
  secret_key  = "YOUR-SECRET-KEY"  
}  
  
resource "aws_instance" "myec2" {  
  ami = "ami-082b5a644766e0e6f"  
  instance_type = var.instance_type  
}
```

variables.tf

```
variable "instancetype" {  
  default = "t2.micro"  
}
```

terraform.tfvars

```
instancetype="t2.large"
```

Loading Variable Values from CLI

```
terraform plan -var="instancetype=t2.small"
```

Loading from custom tfvars file

```
terraform plan -var-file="custom.tfvars"
```

Windows Specific Commands

```
setx TF_VAR_instancetype m5.large  
echo %TF_VAR_instancetype
```

Linux / MAC specific commands

```
export TF_VAR_instancetype t2.nano  
echo $TF_VAR_instancetype
```



RENDEZ VOS
DÉPLOIEMENTS
DYNAMIQUE
(5/7): VARIABLE
(PARTIE 3)

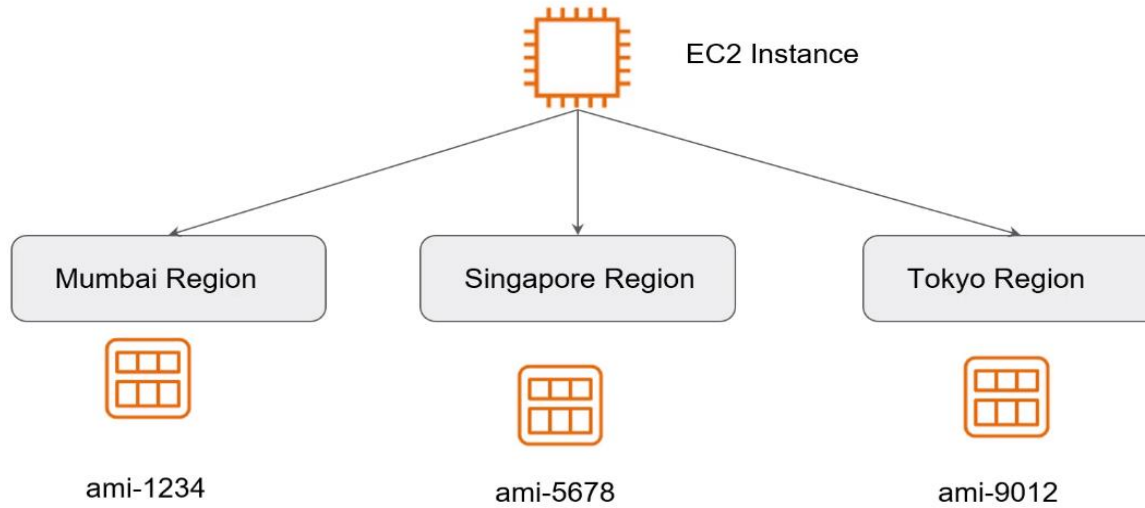
RENDEZ VOS DÉPLOIEMENTS DYNAMIQUE (6/7): VARIABLE (PARTIE 4)

variables.tf

```
variable "usernumber" {  
  type = number  
}  
  
variable "elb_name" {  
  type = string  
}  
  
variable "az" {  
  type = list  
}  
  
variable "timeout" {  
  type = number  
}
```

terraform.tfvars

```
elb_name="myelb"  
timeout="400"  
az=["us-west-1a","us-west-1b"]
```



data-source.tf

```
provider "aws" {
  region      = "ap-southeast-1"
  access_key  = "YOUR-ACCESS-KEY"
  secret_key  = "YOUR-SECRET-KEY"
}

data "aws_ami" "app_ami" {
  most_recent = true
  owners      = ["amazon"]

  filter {
    name   = "name"
    values = ["amzn2-ami-hvm*"]
  }
}

resource "aws_instance" "instance-1" {
  ami           = data.aws_ami.app_ami.id
  instance_type = "t2.micro"
}
```

RENDEZ VOS DÉPLOIEMENTS DYNAMIQUE (7/7): DATA

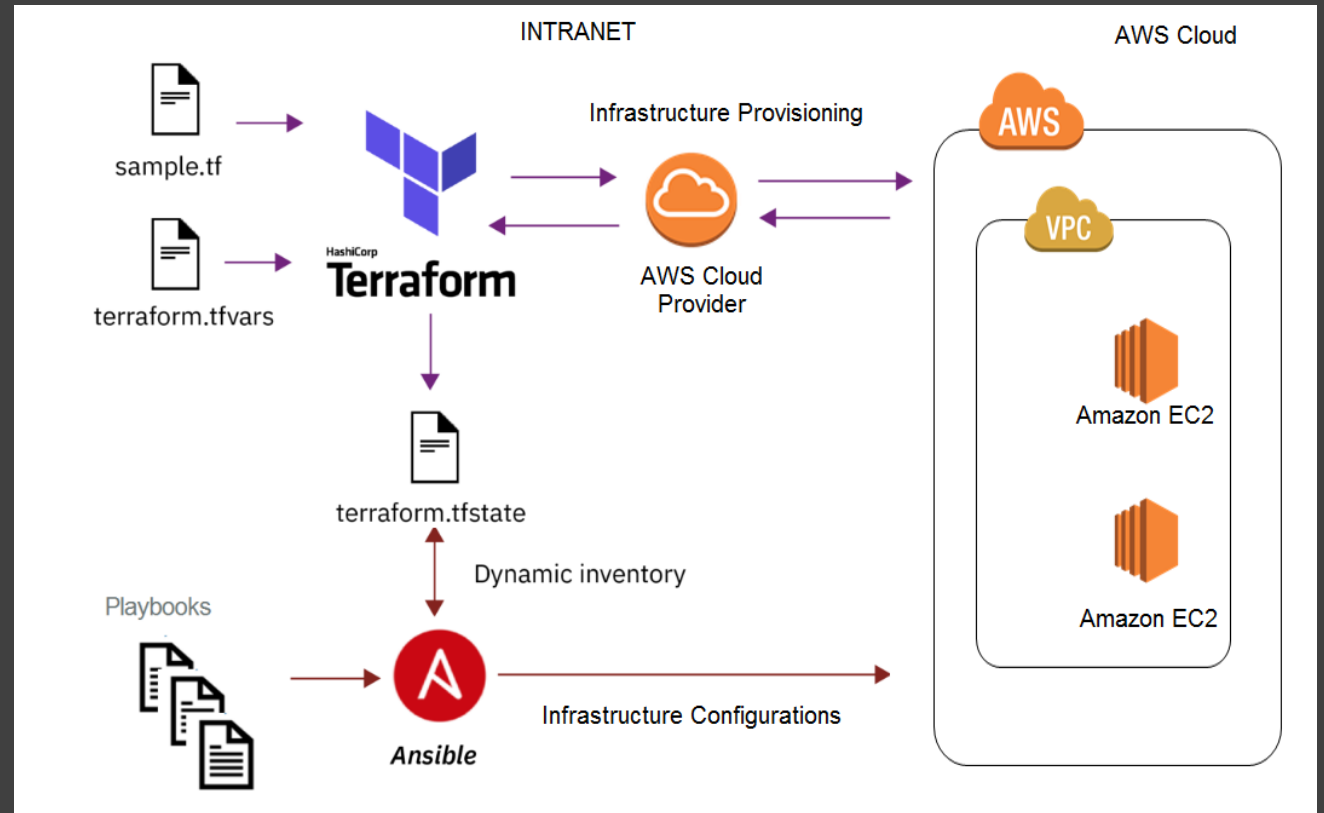
TP-3: Déployez une infrastructure dynamique

- L'objectif est de déployer une instance ec2 avec une ip publique et un security group
- IP publique: vous allez créer une ip publique pour votre EC2
- Security Group: créez une security group pour ouvrir le port 80 et 443, attachez cette security group à votre IP publique
- EC2
 - Votre ec2 doit avoir une taille variabilisée, la valeur par défaut devrait être t2.nano et la valeur à surcharger sera t2.micro
 - L'image AMI à utiliser sera l'image la plus à jour de AMAZON LINUX
 - Spécifiez la key pair à utiliser (devops-<votre prénom>)
 - Attachez l'ip publique à votre instance
 - Variabilisez le Tag afin qu'il contienne au moins le tag: « Name: ec2-<votre prenom> » le N est bien en majuscule
- Supprimez vos ressources avec terraform destroy
- Créez un dossier tp-3 comme vous l'avez fait au tp-2 pour conserver votre code

Plan

- Présentation du formateur
- Introduction au DevOps et IaC
- Terraform
- Déployez vos premières ressources
- Rendez vos déploiements dynamique
- Terraform Provisioners
- Remote management
- Module
- Mini-projet

TERRAFORM PROVISIONERS (1/3): PROBLÉMATIQUE



```
resource "aws_instance" "myec2" {  
    ami = "ami-082b5a644766e0e6f"  
    instance_type = "t2.micro"  
  
    provisioner "local-exec" {  
        command = "echo ${aws_instance.myec2.private_ip} >> private_ips.txt"  
    }  
}
```

TERRAFORM PROVISIONERS (2/3): LOCAL

TERRAFORM PROVISIONERS (3/3): REMOTE

```
resource "aws_instance" "myec2" {
  ami = "ami-082b5a644766e0e6f"
  instance_type = "t2.micro"
  key_name = "kplabs-terraform"

  provisioner "remote-exec" {
    inline = [
      "sudo amazon-linux-extras install -y nginx1.12",
      "sudo systemctl start nginx"
    ]

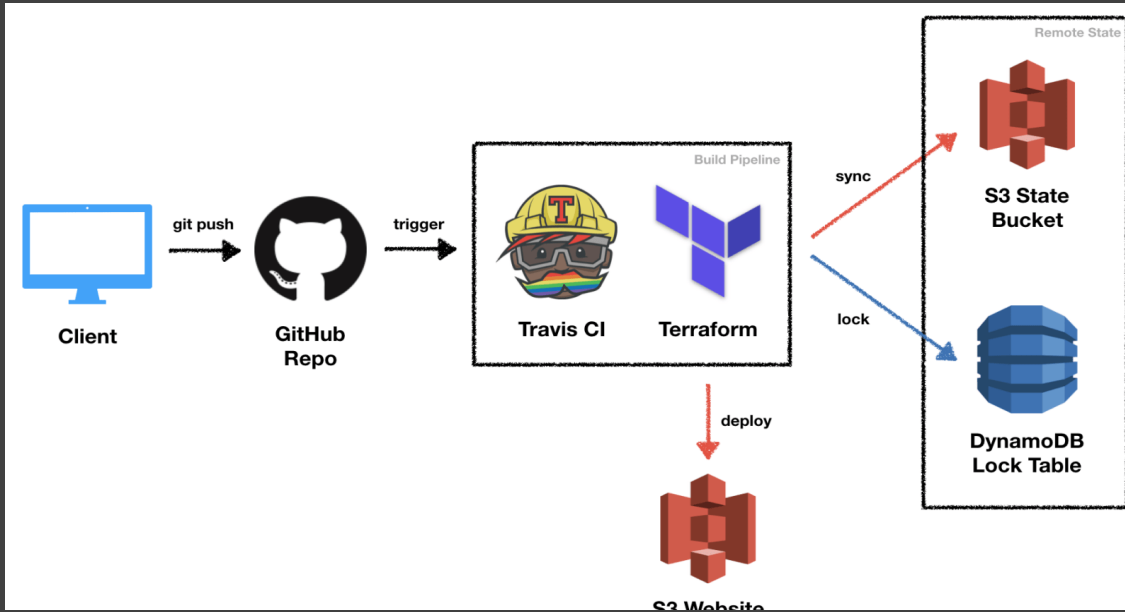
    connection {
      type = "ssh"
      user = "ec2-user"
      private_key = file("../kplabs-terraform.pem")
      host = self.public_ip
    }
  }
}
```

TP-4: Déployez nginx et enregistrez l'ip

- A partir du code du tp-3, vous allez le modifier pour installer nginx sur votre VM
- Vous allez récupérer l'ip, id et la zone de disponibilité de la vm et vous les mettrez dans un fichier nommé `infos_ec2.txt`
- Supprimez vos ressources avec terraform destroy
- Créez un dossier tp-4 comme vous l'avez fait au tp-3 pour conserver votre code

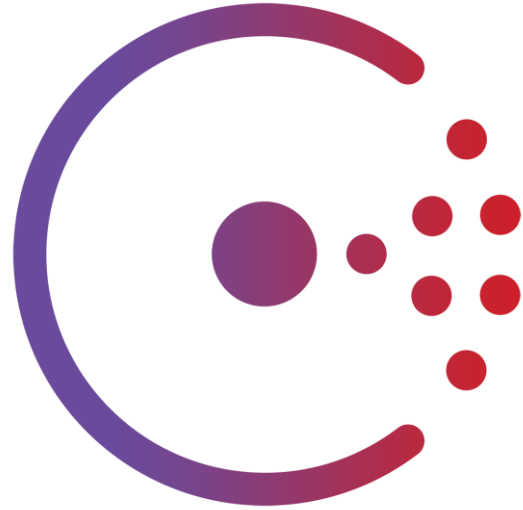
Plan

- Présentation du formateur
- Introduction au DevOps et IaC
- Terraform
- Déployez vos premières ressources
- Rendez vos déploiements dynamique
- Terraform Provisioners
- Remote management
- Module
- Mini-projet



```
provider "aws" {  
  region      = "us-west-2"  
  access_key  = "YOUR-ACCESS-KEY"  
  secret_key  = "YOUR-SECRET-KEY"  
}  
  
resource "aws_instance" "myec2" {  
  ami          = "ami-082b5a644766e0e6f"  
  instance_type = "t2.micro"  
}  
  
resource "aws_iam_user" "lb" {  
  name = "loadbalancer"  
  path = "/system/"  
}  
  
terraform {  
  backend "s3" {  
    bucket = "kplabs-remote-backends"  
    key    = "demo.tfstate"  
    region = "us-east-1"  
    access_key = "YOUR-ACCESS-KEY"  
    secret_key = "YOUR-SECRET-KEY"  
  }  
}
```

REMOTE MANAGEMENT (1/3): REMOTE BACKEND



OSS

REMOTE MANAGEMENT (2/3): SUPPORTED BACKEND

REMOTE MANAGEMENT (3/3): SECURITE

Repositories

1

Code

7,448

Issues

35

Users

2

Pages

368

Script

163

153

145

132

125

101


100

71

59

We've found 7,448 code results

Sort: Best match

 [REDACTED]


Showing the top match. Last indexed 4 days ago.

1

slack:

2


api_token: "xosp-??"

 [REDACTED]

Showing the top match. Last indexed on Mar 28.

1


xosp-[REDACTED]

 [REDACTED]

Showing the top match. Last indexed on Mar 28.

1

SLACK_API_TOKEN="xosp-hogehoghoge"

 [REDACTED]

Showing the top match. Last indexed on Mar 29.

1

{

2

"SLACK_TOKEN": "xosp-[REDACTED]"

3

}

providers.tf

```
provider "aws" {
  region = "us-west-1"
  access_key = "YOUR-ACCESS-KEY"
  secret_key = "YOUR-SECRET-KEY"
}
```

rds.tf

```
resource "aws_db_instance" "default" {
  allocated_storage = 5
  storage_type      = "gp2"
  engine            = "mysql"
  engine_version    = "5.7"
  instance_class    = "db.t2.micro"
  name              = "mydb"
  username          = "foo"
  password          = file("../rds_pass.txt")
  parameter_group_name = "default.mysql5.7"
  skip_final_snapshot = "true"
}
```

rds_pass.txt

```
mysecretpassword505
```

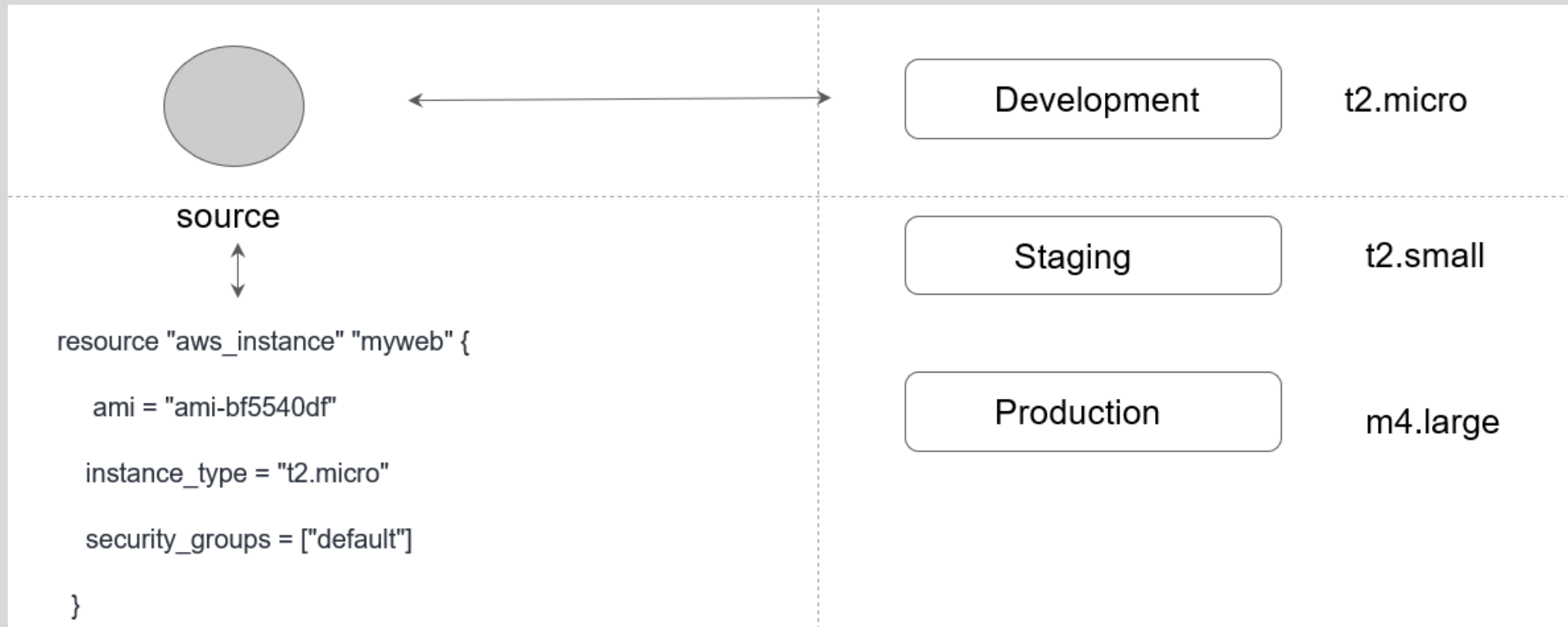

TP-5: Remote Backend

- Créez un s3 nommé terraform-backend-<votre prénom>
- Modifiez votre rendu du tp-4 afin d'y intégrer le stockage du tfstate sur votre s3
- Vérifiez après avoir lancer un déploiement que le fichier sur le drive est bien créé et contient bien les infos à jour
- Supprimez vos ressources avec terraform destroy
- Créez un dossier tp-5 comme vous l'avez fait au tp-4 pour conserver votre code

Plan

- Présentation du formateur
- Introduction au DevOps et IaC
- Terraform
- Déployez vos premières ressources
- Rendez vos déploiements dynamique
- Terraform Provisioners
- Remote management
- Module
- Mini-projet

Module (1/3): Problématique



MODULE (2/3): STRUCTURE

Dev

- jenkins-using-packer.tfvars
- main.tf
- variable.tf

Module

Ec2andLoadBalancer

- main.tf
- variable.tf

ECS

- main.tf
- output.tf
- variable.tf

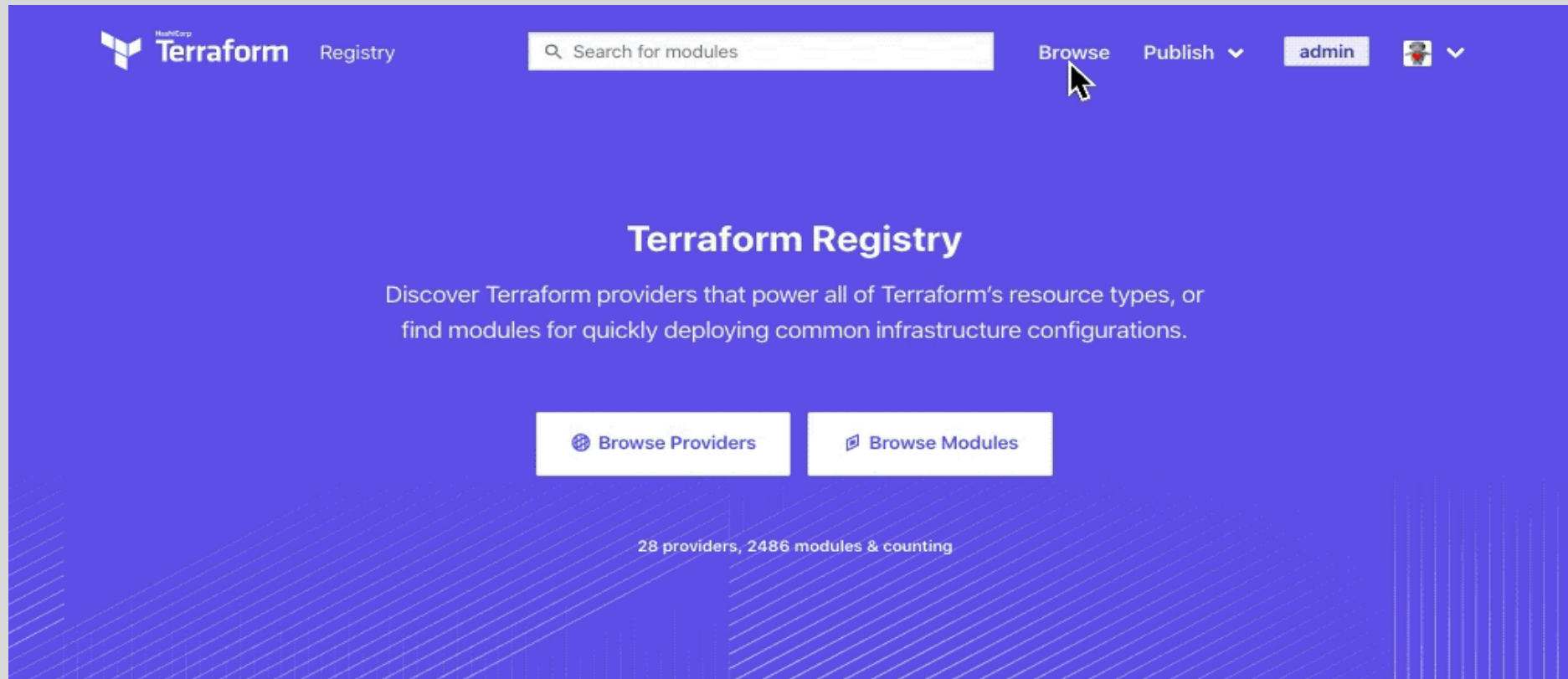
Networking

- main.tf
- output.tf
- variable.tf

SecurityGroup

- main.tf
- output.tf
- variable.tf

Module (3/3): Registry



TP-6: Module ec2

- Créez un module ec2module afin de déployer l'instance de la façon que vous l'avez fait aux tps précédents (ec2 + security group + ip publique)
- Créez ensuite deux dossiers, prod et dev, chacun avec un terraform (main.tf) utilisant le module ec2module créé pour déployer une instance avec respectivement pour taille t2.micro pour la prod et t2.nano pour la dev
- Veuillez également à surcharger le tag pour que ai cette forme : « Name: ec2-prod-<votre prenom> » pour la prod et « Name: ec2-dev-<votre prenom> » pour la Dev
- Lancez ensuite la création de votre ec2 de prod et de dev
- Vérifiez que les ec2 portent bien le bon nom (Tag) et ont la bonne taille correspondant à l'environnement
- Supprimez vos ressources avec terraform destroy
- Créez un dossier tp-6 comme vous l'avez fait au tp-5 pour conserver votre code

Plan

- Présentation du formateur
- Introduction au DevOps et IaC
- Terraform
- Déployez vos premières ressources
- Rendez vos déploiements dynamique
- Terraform Provisioners
- Remote management
- Module
- Mini-projet

Mini-projet: Déployez une infra complète

- Ecrivez un module pour créer une instance ec2 utilisant la dernière version de ubuntu bionic (qui s'attachera l'ebs et l'ip publique) dont la taille et le tag seront variabilisés
- Ecrivez un module pour créer un volume ebs dont la taille sera variabilisée
- Ecrivez un module pour une ip publique (qui s'attachera la security group)
- Ecrivez un module pour créer une security qui ouvrira le 80 et 443
- Créez un dossier app qui va utiliser les 4 modules pour déployer une ec2, bien-sûr vous allez surcharger les variables afin de rendre votre application plus dynamique
- A la fin du déploiement, installez nginx et enregistrez l'ip publique dans un fichier nommé ip_ec2.txt (ces éléments sont à intégrer dans le rôle ec2)
- A la fin de votre travail, poussez votre rôle sur github et envoyez nous le lien de votre repo à easytrainingfr@gmail.com et nous vous dirons si votre solution respecte les bonnes pratiques



**MERCI POUR VOTRE ATTENTION ET À
TRÈS BIENTÔT SUR EAZYTRAINING**