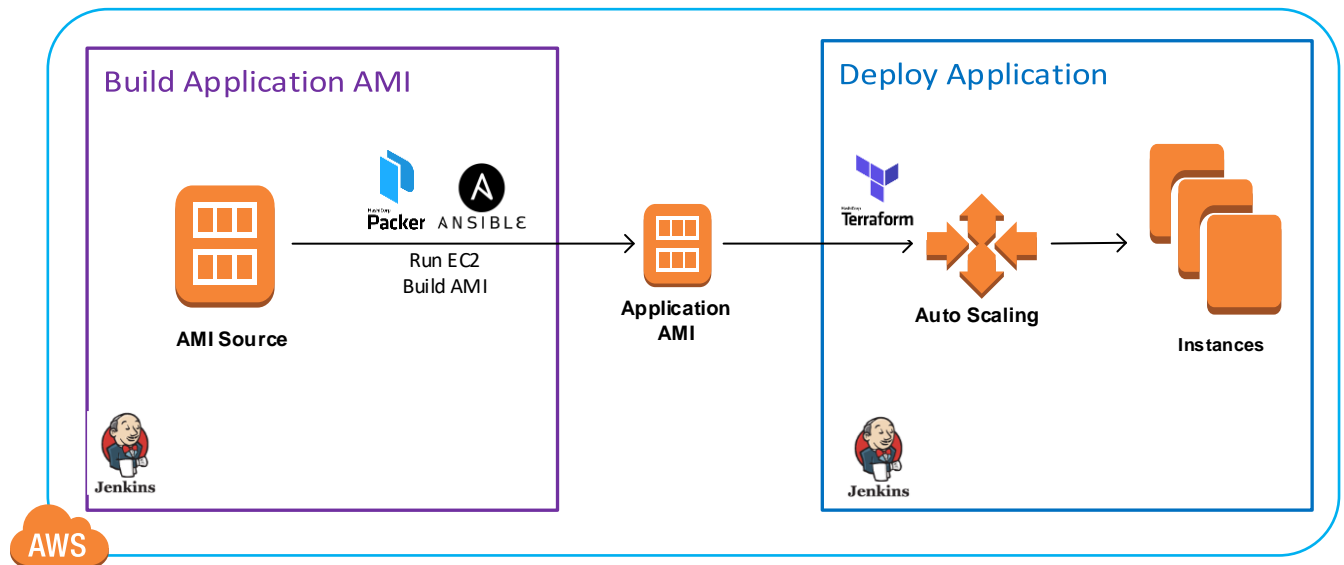


## TP-3 Déployer des ressources sur AWS avec Terraform

### Introduction au TP



Le but du TP est de déployer l'application web à l'aide de l'AMI Applicative préconstruite dans le TP précédent, de manière automatisée avec *Terraform*.

Nous allons automatiser dans un premier temps le déploiement l'infrastructure (VPC, subnets, route tables, NAT Instance..) et ensuite le déploiement de l'application dans un « Autoscaling AWS ».

## Terraform

<https://www.terraform.io/intro/index.html>

### Terraform init

<https://www.terraform.io/docs/commands/init.html>

Cette commande permet d'initialiser un projet Terraform. Il faudra l'exécuter à chaque fois que vous créez un projet Terraform et lorsque vous ajoutez des Providers à votre « Stack ».

### Terraform plan

<https://www.terraform.io/docs/commands/plan.html>

Cette commande permet d'afficher les ressources qui vont être modifiées, sans vraiment appliquer les changements.

### Terraform apply

<https://www.terraform.io/docs/commands/apply.html>

Cette commande permet de déployer la « Stack » définie dans le projet.

### Terraform destroy

<https://www.terraform.io/docs/commands/destroy.html>

Cette commande permet de détruire la « Stack » définie dans le projet.

### tfstate

<https://www.terraform.io/docs/state/index.html>

Ne jamais supprimer le fichier « tfstate » avant d'avoir effectué un « destroy » car ce fichier stocke l'état de la « Stack », sinon il vous sera impossible de déprovisionner programmatiquement les ressources déployées par Terraform.

D'autre part, évitez de supprimer ou modifier manuellement les ressources déployées par Terraform, car si les ressources ne sont plus existantes et que vous réexécutez Terraform il apparaîtra que Terraform ne retrouve plus ses ressources.

Avant de commencer, il faut:

- 1- Nettoyer tous les VPC créés et ne laisser que le VPC par défaut.
- 2- Déployer une instance ToolsCICD dans le « Default VPC »
  - a. Security group autorisant uniquement le SSH depuis l'EFREI (ou depuis votre ip publique)
  - b. Instance Profile (EC2 Rôle) avec les droits Administrateurs
  - c. Installer *Ansible*, *Terraform* et *Packer* (voir le fichier de commandes joint).
- 3- Recréer l'AMI avec *Packer* et *Ansible* avec les fichiers fournis (voir sur Moodle).

Premiers pas avec Terraform : Déployer l'application web dans le défaut VPC

#### **Provider AWS**

<https://www.terraform.io/docs/providers/aws/index.html>

Nous utiliserons juste la variable « région » car les informations d'authentification seront fournies par l'instance « profile » configurée dans les étapes initiales.

#### **Instance**

<https://www.terraform.io/docs/providers/aws/r/instance.html>

Vous utiliserez dans un premier temps l'ID de l'AMI que nous avons précédemment créé via *Packer*.

#### **Security groups**

[https://www.terraform.io/docs/providers/aws/r/security\\_group.html](https://www.terraform.io/docs/providers/aws/r/security_group.html)

[https://www.terraform.io/docs/providers/aws/r/security\\_group\\_rule.html](https://www.terraform.io/docs/providers/aws/r/security_group_rule.html)

Ouvrir le port d'écoute de votre serveur Apache en *ingress* depuis le *Range IP* de l'EFREI.

#### Tests

Terraform init

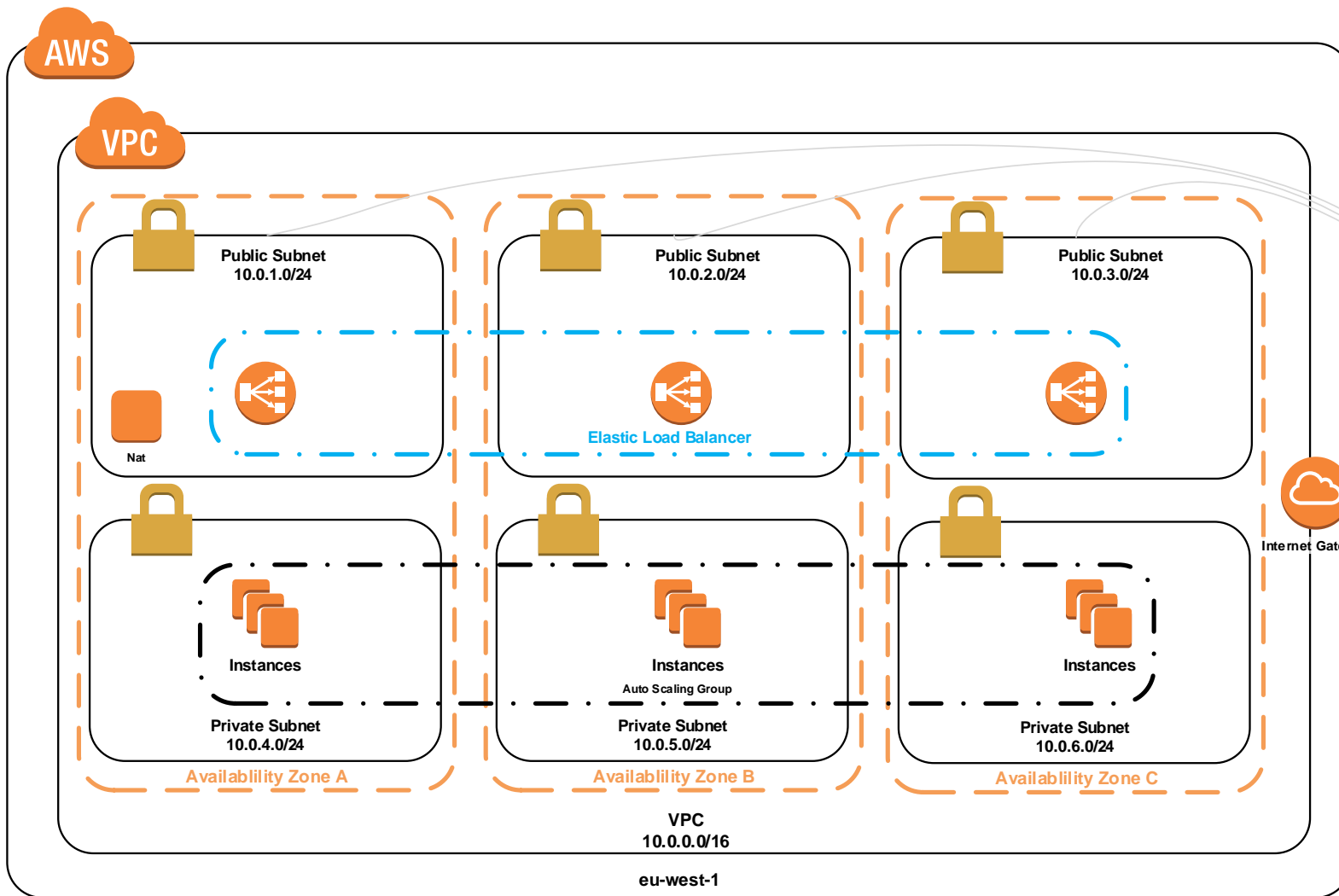
Terraform plan

Terraform apply

Le site web devrait être joignable sur le port d'écoute de l'IP publique de l'instance.

Détruire la « stack » de ressources

Terraform destroy



Custom Public Route Table	
Destination	Target
10.0.0.0/16	local
0.0.0.0/0	IGW

Main Route Table	
Destination	Target
10.0.0.0/16	local
0.0.0.0/0	NAT

## Déployer Infra avec Terraform

➔ Dans le projet /home/ubuntu/TP\_CICD/DeployInfra

<https://www.terraform.io/docs/providers/aws/r/vpc.html>

CIDR
10.0.0.0/16

### Internet Gateway (IGW)

[https://www.terraform.io/docs/providers/aws/r/internet\\_gateway.html](https://www.terraform.io/docs/providers/aws/r/internet_gateway.html)

### Subnets

<https://www.terraform.io/docs/providers/aws/r/subnet.html>

Subnet	AZ	CIDR
Public-1	eu-west-1a	10.0.1.0/24
Public-2	eu-west-1b	10.0.2.0/24
Public-3	eu-west-1c	10.0.3.0/24
Private-1	eu-west-1a	10.0.4.0/24
Private-2	eu-west-1b	10.0.5.0/24
Private-3	eu-west-1c	10.0.6.0/24

### Nat Instance (User data + SG + rules)

<https://www.terraform.io/docs/providers/aws/r/instance.html>

[https://www.terraform.io/docs/providers/aws/r/security\\_group.html](https://www.terraform.io/docs/providers/aws/r/security_group.html)

[https://www.terraform.io/docs/providers/aws/r/security\\_group\\_rule.html](https://www.terraform.io/docs/providers/aws/r/security_group_rule.html)

### Route Table (Main et Public + route)

[https://www.terraform.io/docs/providers/aws/r/default\\_route\\_table.html](https://www.terraform.io/docs/providers/aws/r/default_route_table.html)

[https://www.terraform.io/docs/providers/aws/r/route\\_table.html](https://www.terraform.io/docs/providers/aws/r/route_table.html)

### Route table Association (Public)

[https://www.terraform.io/docs/providers/aws/r/route\\_table\\_association.html](https://www.terraform.io/docs/providers/aws/r/route_table_association.html)

## Déployer l'Application avec Terraform

➔ Dans le projet `/home/ubuntu/TP_CICD/DeployWeb`

Quelques connaissances requises (AWS ELB et AWS EC2 Autoscaling Group)

What Is a Classic Load Balancer?

<https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/introduction.html>

What Is Amazon EC2 Auto Scaling?

<https://docs.aws.amazon.com/autoscaling/ec2/userguide/what-is-amazon-ec2-auto-scaling.html>

### AMI Data Source

<https://www.terraform.io/docs/providers/aws/d/ami.html>

Il faut récupérer l'AMI qui a été créée par *Packer*.

### VPC Data source

<https://www.terraform.io/docs/providers/aws/d/vpc.html>

Il faut récupérer le VPC qui a été créé dans la « Stack » Infra, récupérer le à partir de son nom.

### Subnets Data source

<https://www.terraform.io/docs/providers/aws/d/subnet.html>

Il faut récupérer les Subnets qui ont été créés dans la « Stack » Infra, récupérer les à partir du nom.

### AZ Data source

[https://www.terraform.io/docs/providers/aws/d/availability\\_zones.html](https://www.terraform.io/docs/providers/aws/d/availability_zones.html)

On récupère les AZ disponibles dans la région.

### SG for ASG & SG for ELB

[https://www.terraform.io/docs/providers/aws/r/security\\_group.html](https://www.terraform.io/docs/providers/aws/r/security_group.html)

[https://www.terraform.io/docs/providers/aws/r/security\\_group\\_rule.html](https://www.terraform.io/docs/providers/aws/r/security_group_rule.html)

On autorise le flux venant d'internet (ou uniquement le CIDR de l'école pour restreindre la visibilité) à accéder au Security group de l'ELB.

On autorise depuis le Security group de l'Autoscaling en *ingress* uniquement le Security Group de l'ELB à communiquer sur le port d'écoute de l'application. Cela permettra à l'ELB de renvoyer le trafic vers les instances de l'Autoscaling.

### Launch Configuration for ASG

[https://www.terraform.io/docs/providers/aws/r/launch\\_configuration.html](https://www.terraform.io/docs/providers/aws/r/launch_configuration.html)

La Launch Configuration permet de définir comment les instances sont déployées dans un Autoscaling Group.

### Auto Scaling Group(ASG)

[https://www.terraform.io/docs/providers/aws/r/autoscaling\\_group.html](https://www.terraform.io/docs/providers/aws/r/autoscaling_group.html)

Elastic Load Balancer(ELB)

<https://www.terraform.io/docs/providers/aws/r/elb.html>

Scale Up & Scale Down Policies

[https://www.terraform.io/docs/providers/aws/r/cloudwatch\\_metric\\_alarm.html](https://www.terraform.io/docs/providers/aws/r/cloudwatch_metric_alarm.html)

[https://www.terraform.io/docs/providers/aws/r/autoscaling\\_policy.html](https://www.terraform.io/docs/providers/aws/r/autoscaling_policy.html)

Pour permettre à l'ASG d'effectuer les actions de « Scale In » et « Scale Down », il faut spécifier des alarmes Cloud Watch et des Autoscaling Policies.

### Aller plus loin avec Terraform

- Variables

<https://www.terraform.io/docs/configuration/variables.html>

- Modules

<https://www.terraform.io/docs/configuration/modules.html>

### Articles en lien :

<https://medium.com/@endofcake/using-terraform-for-zero-downtime-updates-of-an-auto-scaling-group-in-aws-60faca582664>

### Immutable Infrastructure

<https://www.digitalocean.com/community/tutorials/what-is-immutable-infrastructure>

<https://www.hashicorp.com/resources/what-is-mutable-vs-immutable-infrastructure>

### Stratégie de déploiement

<https://opensource.com/article/17/5/colorful-deployments>

### Certificates

<https://letsencrypt.org/fr/>