[Projet AWS CLI]

Créer UN VPC

Dans cette étape, Nous allons créer un VPC qui vas nous permettre d'intégrer les sous réseaux, les passerelles internet et NAT, la table de routage, les Acl et les groupes de sécurité

Commande: aws ec2 create-vpc --cidr-block 10.0.0.0/16 --query Vpc.VpcId --output text\
"Cette commande permet de créer le VPC en lui même"

aws ec2 create-tags --resources vpc-0dfae477284d3029f --tags Key=Name,Value=VPCtonioEval\
"Cette commande permet de nommer le VPC"

Créer et nommer les subnets

Dans cette étape, Nous allons créer les sous réseaux avec comme base l'adresse 10.0.0.0/16.

Commande: aws ec2 create-subnet --vpc-id vpc-0dfae477284d3029f --cidr-block 10.0.1.0/24 --availability-zone us-east-1a --query Subnet.SubnetId --output text
"Pour cette commande elle va nous servir à créer le sous réseau en renseignant dans quelle VPC nous voulons le mettre l'adresse de sous réseau que nous voulons dans ce cas la ce sous réseau sera le Publique."

aws ec2 create-tags --resources subnet-016c7b6a405997f98 --tags Key=Name,Value=PublicSubnet
"Ceci nous permet de nommé le sous réseau en l'occurence dans ce cas c'est PublicSubnet"

aws ec2 create-subnet --vpc-id vpc-0dfae477284d3029f --cidr-block 10.0.2.0/24 --availability-zone us-east-1a --query Subnet.SubnetId --output text
"Pour cette commande elle va nous servir à créer le sous réseau en renseignant dans quelle VPC nous voulons le mettre l'adresse de sous réseau que nous voulons dans ce cas la ce sous réseau sera le Privé."

aws ec2 create-tags --resources subnet-07cf1c058ecd2d61a --tags Key=Name,Value=PrivateSubnet
"Ceci nous permet de nommé le sous réseau en l'occurence dans ce cas c'est PrivateSubnet"

Créer, nommer et attacher la Passerelle Internet

Dans cette étape, Nous avons créer, nommer et attacher la Passerelle Internet. Commande: aws ec2 create-internet-gateway --query InternetGateway.InternetGatewayId --output text\
"Cette commande permet de créer la passerelle internet"

aws ec2 create-tags --resources igw-023ac829911442225 --tags Key=Name,Value=tonio-gateway-cli\
"Ici nous allons donner un nom a notre passerelle Internet"

aws ec2 attach-internet-gateway --vpc-id vpc-0dfae477284d3029f --internet-gateway-id igw-023ac829911442225\
"Puis enfin nous allons attacher la passerelle Internet à notre VPC"

Créer la Passerelle NAT et l'IP Elastic

Dans cette étape, nous allons créer et configurer la Passerelle NAT et l'IP Elastic

Commande: aws ec2 allocate-address --domain vpc --query 'AllocationId' --output text --region us-east-1a --tag-specifications 'ResourceType=elastic-ip,Tags=[{Key=Name,Value=IPElasticTonioEval}]'
"Cette commande permet de créer les IP Elastic qui nous permettra de relier a la passerelle NAT"

aws ec2 create-nat-gateway --subnet-id subnet-07cf1c058ecd2d61a --allocation-id eipalloc-0b988b0fd1b38af09
"Cette commande permet de créer la passerelle NAT en reliant bien l'IP Elastic"

Créer et configurer la table de routage pour la Passerelle Internet et la Passerelle NAT

Dans cette étape, Nous allons nous attaquer a la configuration de la table de routage et lié la Passerelle Internet et NAT Commande: aws ec2 create-route-table --vpc-id vpc-0dfae477284d3029f --query RouteTable.RouteTableId --output text
"Cette commande, nous permettra de créer une table de routage et de l'intégrer dans notre VPC"

aws ec2 create-tags --resources rtb-087f5c9af12417985 --tags Key=Name,Value=tonio-routable-cli "Celle ci, concernera le nom que l'on donnera a notre table de routage"

aws ec2 create-route-table --vpc-id vpc-0dfae477284d3029f --query RouteTable.RouteTableId --output text
"Cette commande, nous permettra de créer une table de routage et de l'intégrer dans notre VPC"

aws ec2 create-tags --resources rtb-005f5882cae43f2c3 --tags Key=Name,Value=tonio-routable-nat-cli
"Celle ci, concernera le nom que l'on donnera a notre table de routage"

aws ec2 create-route --route-table-id rtb-087f5c9af12417985 --destination-cidr-block 0.0.0.0/0 --gateway-id igw-023ac829911442225
"Cette commande permet de lier la Passerelle Internet a la table de routage Publique" aws ec2 create-route --route-table-id rtb-005f5882cae43f2c3 --destination-cidr-block 0.0.0.0/0 --nat-gateway-id nat-08676ff584c3a39a6
"Cette commande permet de lier la Passerelle NAT a la table de routage Privé qui nous permettra d'avoir accès a internet sans avoir d'addresse IP publique"

Associer les tables de routage au subnets

Dans cette étape, nous allons associer les deux tables que nous avons au deux sous réseau existant

Commande: aws ec2 associate-route-table --route-table-id rtb-087f5c9af12417985 --subnet-id subnet-016c7b6a405997f98
"Dans cette commande, nous pouvons constater que cela va affecter la table de routage publique au sous réseau publique" aws ec2 associate-route-table --route-table-id rtb-005f5882cae43f2c3 --subnet-id subnet-07cf1c058ecd2d61a

"Dans cette commande, nous pouvons constater que cela va affecter la table de routage privée au sous réseau privée"

Créer et configurer les ACL

Dans cette étape, Nous allons créer un pool ACL et configurer les ACL

Commande: aws ec2 create-network-acl --vpc-id vpc-0dfae477284d3029f
"Cette commande, permet de créer le pool ACL ou nos règle entrante et sortante seront configurée dans notre cas nous avons besoin que des règles entrantes."

aws ec2 create-network-acl-entry --network-acl-id acl-05fecb616c8b1c844 --rule-number 115 --ingress --protocol tcp --rule-action allow --cidr-block 0.0.0.0/0 --port-range From=22,To=22
"Nous pouvons constater que cette Règle entrante d'ACL nous permettra de nous connecter au port 22 depuis n'importe quelle IPv4" aws ec2 create-network-acl-entry --network-acl-id acl-05fecb616c8b1c844 --rule-number 120 --ingress --protocol tcp --rule-action allow --cidr-block 0.0.0.0/0 --port-range From=5085,To=5085
"Nous pouvons constater que cette Règle entrante d'ACL nous permettra de nous connecter au port 5085 depuis n'importe quelle IPv4" aws ec2 create-network-acl-entry --network-acl-id acl-05fecb616c8b1c844 --rule-number 125 --ingress --protocol tcp --rule-action allow --cidr-block 0.0.0.0/0 --port-range From=1024,To=65535
"Nous pouvons constater que cette Règle entrante d'ACL nous permettra d'ouvrir tout les ports entre 1024 et 65535 depuis n'importe quelle IPv4"

Créer et configurer groupe de sécurité

Dans cette étape, nous allons créer et configurer le groupe de sécurité

Commande: aws ec2 create-security-group --group-name SG-Eval --description "Groupe de securite cli" --vpc-id vpc-0dfae477284d3029f
"Ceci, est la commande qui permet de créer un group de sécurité et de le lier avec notre VPC"

aws ec2 authorize-security-group-ingress --group-id sg-0792bbb65ef92d374 --protocol ssh --port 22 --cidr 0.0.0.0/0
"Nous avons, ci-dessus la commande qui permet de créer une règle qui autorise la connexion que sur le port 22 depuis n'importe quelle adresse IPv4 " aws ec2 authorize-security-group-ingress --group-id sg-0792bbb65ef92d374 --protocol tcp --port 5085 --cidr 0.0.0.0/0
"Nous avons, ci-dessus la commande qui permet de créer une règle qui autorise la connexion que sur le port 5085 depuis n'importe quelle adresse IPv4"

Créer les Instances

Dans cette étape,Nous allons créer les machines virtuelle qui vont nous permettre d'effectuer notre travail. Nous avons une machine publique qui dois acceuillir un application web donc qui dois être accessible depuis l'exterieur, et une deuxième machine privée cette fois ci qui ne sera accessible que depuis notre infrastructure et plus précisément depuis la machine virtuelle publique

Commande:

Pour Public: aws ec2 run-instances --image-id ami-058bd2d568351da34 --count 1 --instance-type t2.micro --key-name toniokey --security-group-ids sg-0bcb357697fc076ff --subnet-id subnet-016c7b6a405997f98 --associate-public-ip-address --tag-specifications --region us-east-1 'ResourceType=instance,Tags=[{Key=DebianServer ,Value=Beta}]'
"Pour cette commande, nous pouvons voir différente configuration qui vont nous permettre d'établir une machine virtuelle il nous faudra ajouter une images type debian ou ubuntu notre clé ssh qui nous permettra nous de nous connecter a cette machine d'intégrer le groupe de sécurité pour que les restriction entre en place le sous réseau dans lequel elle va se connecter et d'autoriser l'IP Publique"

Pour Private: aws ec2 run-instances --image-id ami-058bd2d568351da34 --count 1 --instance-type t2.micro --key-name toniokey --security-group-ids sg-0bcb357697fc076ff --subnet-id subnet-07cf1c058ecd2d61a --tag-specifications --region us-east-1 'ResourceType=instance,Tags=[{Key=DebianServer ,Value=Beta}]'
"Pour cette commande, nous pouvons voir une configuration similaire la seule chose qui change c'est le sous réseau qu'il faut modifier et mettre le private subnet et enlever le paramètre d'autosisation d'IP Publique"