

Contents

1 Présentation :	1
2 Préalables :	1
3 Vérifications.....	2
4 vérifications de base.....	2
5 vérifications.....	2
6 Création du dossier du projet :	2
7 Initialiser Terraform :	4
8 Voir le plan :	4
9 Création de la ressource :	4
10 Vérifications :	4
10.1 vérifications en mode CLI.....	4
10.2 Vérifier les polices attachées.....	5
11 Suppression de la ressource.....	5
11.1 Avec Terraform :	5
12 On vérifie dans AWS , ou CLI la suppression effective du bucket.....	6

1 Présentation :

Buts :

- Présenter la méthode d'utilisation et de lancement de Terraform avec un exemple très simple.
- utiliser Terraform pour créer un rôle IAM dans AWS
- rôle attribué à une fonction lambda pour lui permettre de créer des log de type groups, stream et event

2 Préalables :

- OS : Windows
- Posséder un compte dans AWS
- Installer AWS CLI
- Installer Terraform
- Télécharger terraform.exe à partir du site officiel de HashiCorp
 - <https://developer.hashicorp.com/terraform>
- Installer dans :c:\terraform

3 Vérifications

#terraform -version

```
C:\Windows\System32>terraform -version
Terraform v1.14.3
on windows_amd64
```

4 vérifications de base

#aws --version

```
C:\Windows\System32>aws --version
aws-cli/1.42.17 Python/3.12.6 Windows/11 botocore/1.40.17
```

5 vérifications

#aws sts get-caller-identity

6 Création du dossier du projet :

Répertoire : terraform-lab-iam

Créer un fichier main :

Main.trf dans le répertoire terraform-lab-iam

Copier le code suivant dans le fichier :

```
terraform {
  required_version = ">= 1.5"

  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
}

provider "aws" {
  region = "ca-central-1"
}
```

Rajouter la section suivante :

Création d'un rôle pour la fonction lambda

```
resource "aws_iam_role" "lambda_exec_role" {
  name = "lab-lambda-exec-role"

  assume_role_policy = jsonencode({
    Version = "2012-10-17"
```

```
Statement = [{  
  Effect = "Allow"  
  Principal = {  
    Service = "lambda.amazonaws.com"  
  }  
  Action = "sts:AssumeRole"  
}]  
})  
}
```

Rajouter la section suivante :
Ce que le rôle doit pouvoir faire

```
resource "aws_iam_policy" "lambda_logs_policy" {  
  name = "lab-lambda-logs-policy"  
  
  policy = jsonencode({  
    Version = "2012-10-17"  
    Statement = [{  
      Effect = "Allow"  
      Action = [  
        "logs:CreateLogGroup",  
        "logs:CreateLogStream",  
        "logs:PutLogEvents"  
      ]  
      Resource = "*"   
    }]  
  })  
}
```

Rajouter la section suivante :
Attacher la police au rôle

```
resource "aws_iam_role_policy_attachment" "attach_logs" {  
  role      = aws_iam_role.lambda_exec_role.name  
  policy_arn = aws_iam_policy.lambda_logs_policy.arn  
}
```

Position répertoire et fichier :

```
Directory of C:\terraform\terraform-lab-iam

12/17/2025  08:00 PM    <DIR>          .
12/17/2025  07:39 PM    <DIR>          ..
12/17/2025  07:47 PM    <DIR>          .terraform
12/17/2025  07:47 PM                1,407 .terraform.lock.hcl
12/17/2025  07:46 PM                1,036 main.tf
12/17/2025  08:00 PM                181 terraform.tfstate
12/17/2025  08:00 PM                3,146 terraform.tfstate.backup
                4 File(s)                5,770 bytes
                3 Dir(s)  698,063,101,952 bytes free

C:\terraform\terraform-lab-iam>
```

7 Initialiser Terraform :

```
#terraform init
```

L'initialisation crée automatiquement les deux fichiers suivants dans le répertoire de travail, du projet.

```
.terraform\
.terraform.lock.hcl
```

8 Voir le plan :

```
#terraform plan
```

On doit avoir comme résultat :

Plan: 1 to add, 0 to change, 0 to destroy

9 Création de la ressource :

```
# Terraform apply
```

10 Vérifications :

10.1 vérifications en mode CLI

```
# aws iam get-role --role-name lab-lambda-exec-role
```



```
Administrator: Command Prompt

}

# aws_iam_role_policy_attachment.attach_logs will be destroyed
- resource "aws_iam_role_policy_attachment" "attach_logs" {
  - id          = "lab-lambda-exec-role-20251218004830910200000001" -> null
  - policy_arn  = "arn:aws:iam::[redacted]:policy/lab-lambda-logs-policy" -> null
  - role       = "lab-lambda-exec-role" -> null
}

Plan: 0 to add, 0 to change, 3 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

aws_iam_role_policy_attachment.attach_logs: Destroying... [id=lab-lambda-exec-role-20251218004830910200000001]
aws_iam_role_policy_attachment.attach_logs: Destruction complete after 0s
aws_iam_policy.lambda_logs_policy: Destroying... [id=arn:aws:iam::[redacted]:policy/lab-lambda-logs-policy]
aws_iam_role.lambda_exec_role: Destroying... [id=lab-lambda-exec-role]
aws_iam_policy.lambda_logs_policy: Destruction complete after 0s
aws_iam_role.lambda_exec_role: Destruction complete after 0s

Destroy complete! Resources: 3 destroyed.

C:\terraform\terraform-lab-iam>
```

.....

12 On vérifie dans AWS , ou CLI la suppression effective

Soit via la console, soit via le CLI

```
# aws iam get-role --role-name lab-lambda-exec-role
```

Résultat attendu :

```
C:\terraform\terraform-lab-iam> aws iam get-role --role-name lab-lambda-exec-role

An error occurred (NoSuchEntity) when calling the GetRole operation: The role with name lab-lambda-exec-role cannot
ound.

C:\terraform\terraform-lab-iam>
```

13 schemas

