



EVALUATION EN COURS DE FORMATION

PARCOURS ADMINISTRATEUR SYSTEME DEVOPS



Sécuriser l'Infrastructure

Evaluation : ECF3

Version : du 10/11/2025

Auteur : FT

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

Sommaire

Table des matières

1. PRESENTATION	2
1.1. DESCRIPTION DE LA COMPETENCE – PROCESSUS DE MISE EN ŒUVRE	2
1.2. CONTEXTE(S) PROFESSIONNEL(S) DE MISE EN ŒUVRE.....	2
1.3. CRITERES DE PERFORMANCE.....	2
2. ECF2 - QUESTIONS.....	3
2.1. QUELS SONT LES DANGERS D'EXPOSER SES VARIABLES / SECRETS ?.....	3
2.2. POURQUOI GARDE-T-ON NOTRE CODE SUR UN VCS ? (EX: GITHUB / GITLAB / BITBUCKET).....	3
2.3. POURQUOI CREER UNE BRANCHE DE DEVELOPPEMENT ?.....	4
2.4. À QUOI SERT UNE PULL REQUEST ?	4
2.5. POURQUOI UTILISER UN GESTIONNAIRE DE SECRETS TEL QUE VAULT?	5
2.6. COMMENT DEMARRER UN SERVEUR VAULT ?	5
2.7. QU'EST-CE QUE L'ENCRYPTION SSL?.....	6
3. PARTIE 1 : GESTION DES VARIABLES.....	7
3.1. CREATION DU REPO GIT.....	7
3.2. CREATION DE LA BRANCHE "DEV"	9
3.3. CREATION DE L'ORGANISATION ET DU WORKSPACE TERRAFORM CLOUD	10
3.4. CHOIX DU PROVIDER.....	13
3.5. CREATION D'UN COMpte SUR LE PROVIDER CLOUD AWS.....	13
3.6. CREATION D'UN MODULE CONTENANT UNE VM LINUX SIMPLE	15
3.7. PASSAGE DES CLES API DANS LES VARIABLES.....	18
3.8. PASSAGE DES CLES API VIA TERRAFORM CLOUD	19
4. PARTIE 2 : VAULT - PASSAGE DE SECRETS A LA VOLEE.....	21
4.1. MISE EN PLACE DE VAULT (OPTION UI CONFIGUREEE)	21
4.2. CONNECTER LE MODULE TERRAFORM AU SERVEUR VAULT	22
4.3. COPIE DES CLES API (UNSAFE ET SEULEMENT POUR LA CORRECTION)	22
4.4. CLES API RENSEIGNEES PAR VAULT.....	22
4.5. PULL REQUEST SUR MAIN(MASTER)	26
5. ANNEXE	29
5.1. CREATION D'UN MODULE CONTENANT UNE VM LINUX SIMPLE	29
5.2. PASSAGE DES CLES API VIA TERRAFORM CLOUD	31
5.3. DECLARATION DE VAULT DANS TERRAFORM (MAIN.TF)	32

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

1. PRESENTATION

1.1. DESCRIPTION DE LA COMPETENCE – PROCESSUS DE MISE EN ŒUVRE

- À la demande d'un client ou de son responsable, mettre en place des pratiques de sécurité pour le provisioning de l'infrastructure.
- En tenant compte d'un cahier des charges, garder les secrets séparément du code provisioning, automatisé leurs usages et mettre en place un système de registre privé pour nos prochains modules.
- Mettre tout cela en service, tester le bon fonctionnement de toutes les parties et documenter son travail.

1.2. CONTEXTE(S) PROFESSIONNEL(S) DE MISE EN ŒUVRE

Ce travail s'effectue seul, en relation avec les équipes en charge du réseau et de la sécurité.

1.3. CRITERES DE PERFORMANCE

- Avoir un **repo git** sur lequel le code se trouve
- Avoir une **branche dev** sur laquelle toutes les étapes ont été commit
- Créer une **pull request**
- Avoir setup **Terraform cloud** en version control pour ce même repo git
- Terraform cloud doit être configuré sur la branche dev (pour la correction)
- **Aucun fichier** finissant par **tfvars** n'apparaît dans le repo git
- **Aucun secret ne fait partie des variables** du repo et/ou du Terraform cloud
- Le code terraform est configuré pour interagir avec un **serveur vault** (par exemple sur votre vm local)
- Vérifier que lors de **chaque merge** un **Plan** soit automatiquement lancer par **Terraform** cloud
- La configuration est conforme au cahier des charges
- Les documentations sont mises à jour
- Savoir-faire techniques, savoir-faire organisationnels, savoir-faire relationnels
- Documenter le code fourni
- Diagnostiquer un dysfonctionnement et le corriger
- Consulter de la documentation technique rédigée en anglais

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

2. ECF2 - QUESTIONS

2.1. QUELS SONT LES DANGERS D'EXPOSER SES VARIABLES / SECRETS ?

L'exposition des variables ou secrets représente un risque majeur de sécurité.

Les dangers principaux sont les suivants :

- **Compromission de services**

Si une clé API ou des secrets sont exposés, un attaquant peut l'utiliser pour accéder au système (base de données, service cloud, etc) et de ce fait peut modifier la configuration ou le rendre inaccessible.

- **Accès non autorisés**

Des informations comme des mots de passe, des clés API, des tokens d'authentification qui sont exposées peuvent permettre à des personnes mal intentionnées d'y accéder et de les utiliser à des fins frauduleuses ce qui rend vulnérable les systèmes cibles.

- **Vulnérabilités / Failles de sécurité**

Un secret compromis peut être une porte d'entrée vers d'autres vulnérabilités comme injecter du code malveillant, ou déployer du code compromis.

- **Utilisation des ressources ou données**

Un secret exposé peut donner accès à une base de données, des fichiers, contenant des informations sensibles (utilisateurs, mots de passe, ...) et d'utiliser les ressources à des fins malhonnêtes.

- **Conséquences financières**

Beaucoup de services cloud (AWS, GCP, Azure) facturent à l'usage et si des secrets sont volés, des personnes mal intentionnées peuvent exploiter ses ressources et entraîner des conséquences financières en lançant des services non désirés.

2.2. POURQUOI GARDE-T-ON NOTRE CODE SUR UN VCS ? (EX: GITHUB / GITLAB / BITBUCKET)

Le code est gardé sur un VCS (Version Control System) qui est donc un système de gestion de versions, pour suivre le développement de logiciels.

Les raisons d'utiliser ce type d'outil sont les suivantes :

- **Historique des versions**

Cela permet de garder une trace de toutes les modifications apportées au code, comme les ajouts ou les suppressions, et de les enregistrer avec un horodatage et un commentaire.

Cela permet donc de revenir en arrière, de savoir qui a fait quoi, quand et pourquoi.

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS	Auteur	FT
ECF3	Titre	Sécuriser l'Infrastructure	Du	10/11/2025

- **Effectuer un travail collaboratif**

Plusieurs personnes peuvent travailler sur un même projet en parallèle par le biais de branches séparées, de suivre les tâches et de pouvoir effectuer des fusions quand les changements sont validés.

- **Gestions des Branches**

Comme indiqué précédemment, il est possible de créer des branches pour ajouter des fonctionnalités au code principal, ou de corriger des bugs, pour ensuite fusionner ces changements ou corrections.

- **Sauvegarde centralisée**

Le code n'est pas stocké uniquement sur la machine locale mais est aussi hébergé dans un environnement distribué, ce qui permet de stocker une copie complète du projet dans un espace sécurisé.

2.3. POURQUOI CREER UNE BRANCHE DE DEVELOPPEMENT ?

Il est possible de créer des branches pour ajouter des fonctionnalités au code principal, ou de corriger des bugs, pour ensuite fusionner ces changements ou corrections.

Cela permet donc plus précisément d'effectuer les actions suivantes :

- **Isoler le travail**

Une branche permet de séparer le travail du code principal (branche master ou main) pour développer, tester sans impacter le projet principal.

- **Effectuer des développements en parallèle**

Cela permet de travailler sur des fonctionnalités différentes et que chacun ait sa propre branche sans impacter le code principal ou le travail des autres collaborateurs.

- **Déploiement progressif**

Une fois qu'une fonctionnalité est terminée sur une branche, elle peut être soumise à une "pull request", pour être lue et validée avant de l'intégrer à la branche principale.

Cela permet des revues de code simplifiées dans une démarche Agile, en organisant le flux de travail de manière structurée et progressive.

2.4. À QUOI SERT UNE PULL REQUEST ?

Une Pull Request (sur GitHub) ou Merge (sur Gitlab) permet l'intégration d'une branche dans une autre, d'assurer la qualité et de travailler en équipe efficacement.

Cette demande est donc une demande de relecture à l'équipe ou au responsable du projet, pour valider les changements apportés dans une branche dans une autre branche qui est généralement la principale.

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

2.5. POURQUOI UTILISER UN GESTIONNAIRE DE SECRETS TEL QUE VAULT ?

Les raisons principales d'utiliser un gestionnaire de secrets comme Vault sont les suivantes :

- **Centralisation des secrets**

Les secrets sont stockés et chiffrés dans un "coffre-fort" sécurisé ce qui permet de protéger les données sensibles (mots de passe, tokens d'accès, clés API, certificats SSL, etc).

- **Contrôle d'accès granulaire**

Vault permet de définir qui a accès à quoi de manière "fine", ce qui permet de limiter d'éventuelles compromissions.

- **Sécurité renforcée**

Vault utilise des mécanismes de chiffrement, ou d'intégration avec des solutions d'authentification comme LDAP ou IAM cloud pour récupérer les secrets de manière sécurisé et automatisée sans les stocker localement.

- **Traçabilité des accès**

Des logs ou journaux d'audit sont enregistrés à chaque demande d'accès pour savoir qui a accédé à quoi, quand et d'où.

2.6. COMMENT DEMARRER UN SERVEUR VAULT ?

Deux méthodes sont possibles pour démarrer un serveur Vault, la méthode avec un démarrage rapide, pour du développement ou des tests, en utilisant le mode "dev" qui utilise un stockage en mémoire (les données sont perdues lors d'un arrêt du serveur).

Mais pour un environnement de production, la configuration devra être plus robuste avec une configuration de sécurité appropriée, en mettant en place un stockage persistant.

- Démarrage en mode "dev"

- Démarrage rapide : **vault server -dev** # un root-token est fourni

- Configuration de la CLI
 - export VAULT_ADDR='http://127.0.0.1:8200'
 - export VAULT_TOKEN='<root-token>'

- Vérifier le status : **vault status**

- Gestion des secrets:
 - création : **vault kv put -mount=secret test key1=password** (dans le chemin secret/data/test)
 - pour le lire : **vault kv get -mount=secret test**
 - lister les secrets : **vault secrets list**

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

- **Démarrage en mode "prod"**

Il faut créer un fichier de configuration (vault.hcl) pour définir le backend de stockage persistant (ex : S3, File), le listener HTTPS, les paramètres de sécurité, initialiser le serveur Vault (pour générer un ensemble de clés de déverrouillage et un token root).

- Exemple de fichier **vault.hcl** (simplifié)

```
storage "raft" {          # raft : backend de stockage intégré pour conserver les données de Vault
    path  = "/opt/vault/data"
    node_id = "vault-server-1"
}

cluster_addr = "http://127.0.0.1:8201"

listener "tcp" {
    address  = "0.0.0.0:8200"
    tls_disable = "false"
    tls_cert_file = "/opt/vault/tls/vault.crt"
    tls_key_file = "/opt/vault/tls/vault.key"
}
```

- Démarrage du serveur : **vault server -config=vault.hcl**

2.7. QU'EST-CE QUE L'ENCRYPTION SSL?

SSL signifie **Secure Sockets Layer**, c'est un protocole de chiffrement qui sert à sécuriser la communication entre un navigateur et un serveur web, il a été remplacé par **TLS (Transport Layer Security)**.

On identifie un site web utilisant SSL/TLS grâce à la présence de "HTTPS" au début de son adresse web (au lieu de "HTTP") et un petit cadenas affiché dans la barre d'adresse, en utilisant un certificat SSL nécessaire pour garantir l'établissement d'une connexion sécurisée.

Cela permet d'authentifier le serveur, de protéger les données sensibles et d'empêcher les attaques de type "man-in-the-middle" (interception de la connexion), et de garantir la confidentialité et la sécurité des communications sur internet.

SSL combine 2 types de chiffrement :

Phase	Type de chiffrement	Rôle
Échange de clé	Asymétrique 	Protéger la clé de session
Communication	Symétrique 	Vitesse + sécurité

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

3. PARTIE 1 : GESTION DES VARIABLES

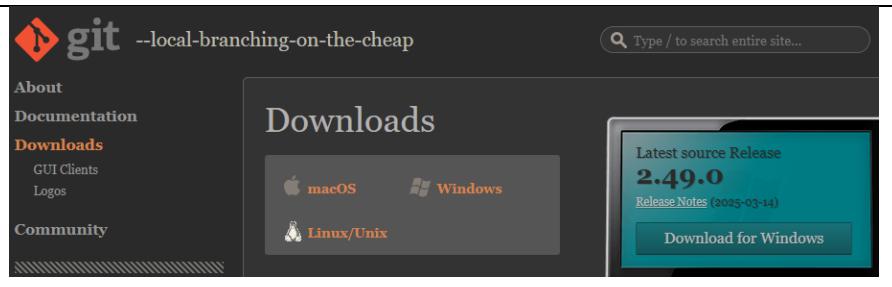
3.1. CREATION DU REPO GIT

a. Repository GIT (Windows)

GIT peut être téléchargé via le lien suivant :

<https://git-scm.com/downloads>

L'installation se fera dans un environnement Windows.



The screenshot shows the official Git website at https://git-scm.com/. The main navigation bar includes 'About', 'Documentation', 'Downloads' (selected), 'Community', and 'Logos'. Under 'Downloads', there are links for 'GUI Clients' (macOS, Windows, Linux/Unix) and 'Logos'. A prominent call-to-action button on the right says 'Download for Windows'.

- Création du répertoire de travail dans Git Bash: `git init terraformcloud-aws-vm`

```
$ git init terraformcloud-aws-vm
Initialized empty Git repository in C:/terraformcloud-aws-vm/.git/
```

- Création du fichier README.md

```
cd terraform-aws-vm
```

```
touch README.md
```

```
$ cd terraformcloud-aws-vm/
ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (master)
$ touch README.md
```

- Ajouter des actions dans README.md qui seront effectuées

```
GNU nano 8.3                                         README.md
Créer une repo git
Créer une branche dev
Pushez chaque prochaines étapes sur dev (pour la correction)
Créer une organisation et un workspace Terraform Cloud
Connecter ce workspace au repo précédemment mentionnée Terraform Cloud
Choisissez un provider parmi cette liste https://registry.terraform.io/browse/providers
Créer un compte sur le provider cloud AWS : https://aws.amazon.com/
Créer un module contenant une VM linux simple configuré à ce provider
SE référer à la documentation terraform pour celui-ci: https://registry.terraform.io/providers/hashicorp/aws/latest/docs
(Pas besoin de provisionner quoi que ce soit, faire juste en sorte que l'initialisation du provider fonctionne
et de pouvoir planifier une machine virtuelle sur le provider)
À la place de passer les clés api directement, passez les via des default variables dans un fichier nommé variables.tf
Enlevez ces variables du code et passer les via Terraform Cloud
```

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

- Ajout du fichier dans le dépôt et commit :

```

git add .
$ git status
On branch master
No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file: README.md

```

```

git commit -m "Initial commit terraformcloud-aws-vm project"
$ git commit -m "Initial commit terraformcloud-aws-vm project"
[master (root-commit) 945bdd4] Initial commit terraformcloud-aws-vm project
 1 file changed, 13 insertions(+)
 create mode 100644 README.md

```

```

ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (master)
$ git status
On branch master
nothing to commit, working tree clean

```

- Renommer la Branche principale : `git branch -M main`

```

ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (master)
$ git branch -M main

ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (main)

```

b. Repository GitHub

Le compte d'accès Github sera le suivant : <https://github.com/fabrice-git-hub>

- Ajout Remote VCS et push sur GitHub

```
git remote add origin https://github.com/fabrice-git-hub/ec3.git
```

```
git push -u origin main
```

```

$ git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 739 bytes | 123.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/fabrice-git-hub/terraformcloud-aws-vm.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

```

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

- Vérification sur Github

The screenshot shows a GitHub repository page for 'terraformcloud-aws-vm'. The repository is private and has 1 branch ('main') and 0 tags. The README file contains the following content:

```


- Créez une repo git
- Créez une branche dev
- Pushez chaque prochaines étapes sur dev (pour la correction)
- Créez une organisation et un workspace Terraform Cloud
- Connectez ce workspace au repo précédemment mentionnée Terraform Cloud
- Choisissez un provider parmi cette liste https://registry.terraform.io/browse/providers
- Créez un compte sur le provider cloud AWS : https://aws.amazon.com/
- Créez un module contenant une VM linux simple configuré à ce provider
- Se référer à la documentation terraform pour celui-ci: https://registry.terraform.io/providers/hashicorp/aws/latest/docs (Pas besoin de provisionner quoi que ce soit, faire juste en sorte que l'initialisation du provider fonctionne et de pouvoir planifier une machine virtuelle sur le provider)
- À la place de passer les clés api directement, passez les via des default variables dans un fichier nommé variables.tf
- Enlevez ces variables du code et passer les via Terraform Cloud

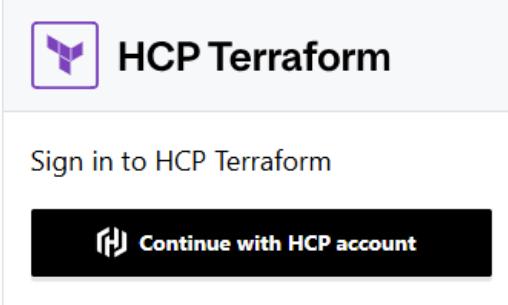
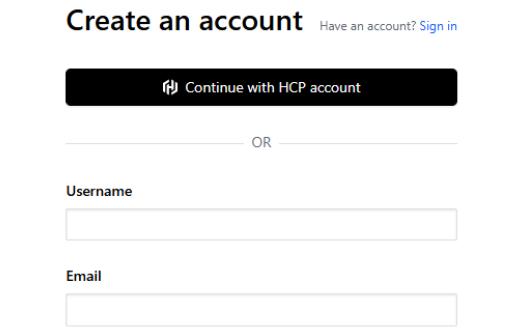
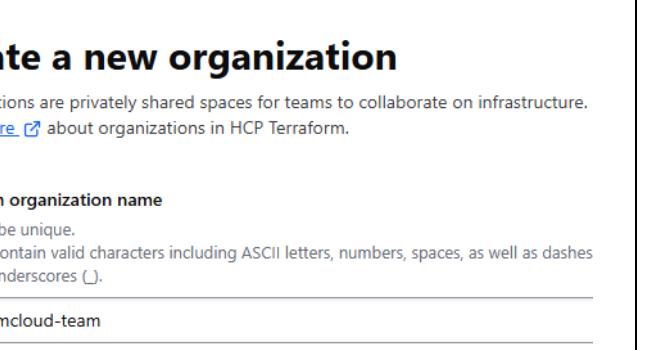
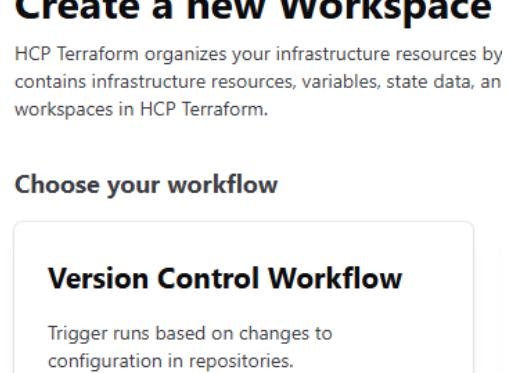
```

3.2. CREATION DE LA BRANCHE "DEV"

git checkout -b dev	ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (main) \$ git checkout -b dev Switched to a new branch 'dev' ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (dev)
git push -u origin dev Chaque prochaine étape sera "pushée"sur dev	\$ git push -u origin dev Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0) remote: remote: Create a pull request for 'dev' on GitHub by visiting: remote: https://github.com/fabrice-git-hub/terraformcloud-aws-vm/pull/new/dev remote: To https://github.com/fabrice-git-hub/terraformcloud-aws-vm.git * [new branch] dev -> dev branch 'dev' set up to track 'origin/dev'.
• Vérification sur Github	

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

3.3. CREATION DE L'ORGANISATION ET DU WORKSPACE TERRAFORM CLOUD

<ul style="list-style-type: none"> Se connecter sur Terraform Cloud 	
<ul style="list-style-type: none"> Créer un nouveau Compte 	
<ul style="list-style-type: none"> Création de l'organisation "terraformcloud-team" 	
<ul style="list-style-type: none"> Création du Workspace "terraformcloud-aws-vm" Sélectionner Version Control Workflow 	

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

- Sélectionner le VCS **Github**

1 Connect to VCS

Connect to a version control provider

Choose the version control provider

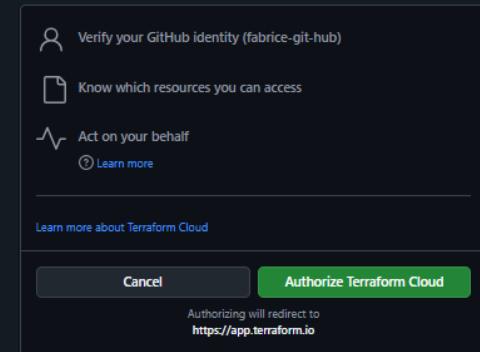
Project: Default Project ⓘ

 GitHub ⚙

 GitLab

- Autoriser **Terraform Cloud**

Terraform Cloud by HashiCorp would like permission to:



- Sélectionner le repository :

terraformcloud-aws-vm



Install Terraform Cloud

Install on your personal account fabrice-git-hub



for these repositories:

All repositories

This applies to all current and future repositories owned by the resource owner.
Also includes public repositories (read-only).

Only select repositories

Select at least one repository. Also includes public repositories (read-only).



Selected 1 repository.

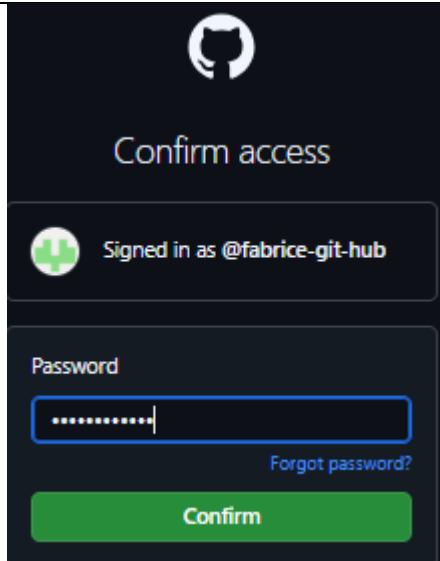
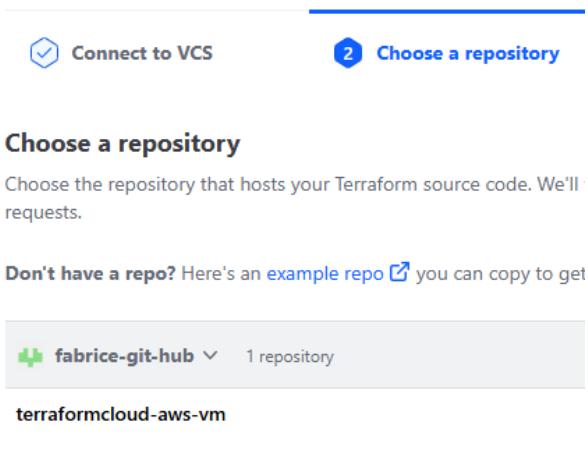
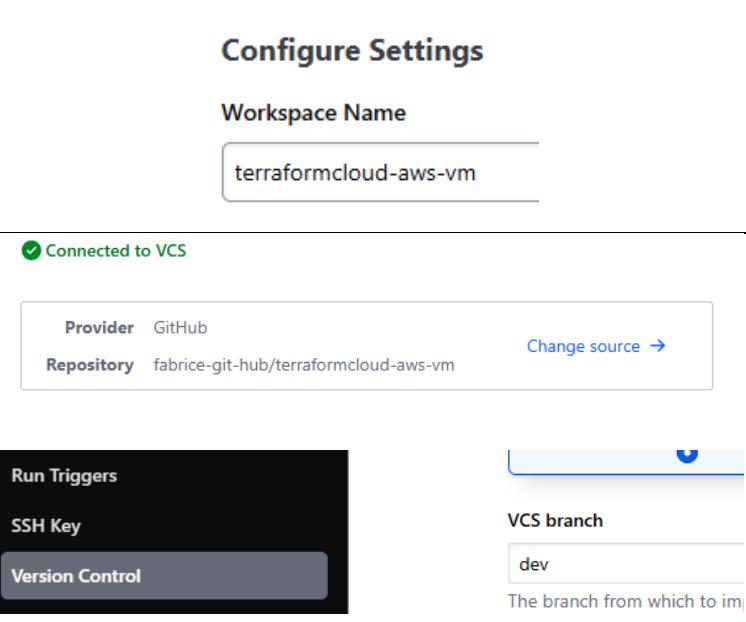
 fabrice-git-hub/terraformcloud-aws-vm 

with these permissions:

✓ Read access to code, metadata, and pull requests

✓ Read and write access to commit statuses

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

<ul style="list-style-type: none"> • Se connecter à Github 	
<ul style="list-style-type: none"> • Finalisation de la création du Workspace 	 <p>Choose a repository</p> <p>Choose the repository that hosts your Terraform source code. We'll request.</p> <p>Don't have a repo? Here's an example repo you can copy to get started.</p> <p>fabrice-git-hub 1 repository</p> <p>terraformcloud-aws-vm</p>
<ul style="list-style-type: none"> • Configuration de Terraform Cloud sur la branche DEV 	 <p>Connected to VCS</p> <p>Provider GitHub Change source →</p> <p>Repository fabrice-git-hub/terraformcloud-aws-vm</p> <p>Run Triggers</p> <p>SSH Key</p> <p>Version Control</p> <p>VCS branch</p> <p>dev</p> <p>The branch from which to import</p>

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

3.4. CHOIX DU PROVIDER

Sélectionner le provider sur [Terraform Providers](#)

<ul style="list-style-type: none"> Choix du Provider 	<p>Providers Providers are a logical abstraction of an upstream API. They are responsible for understanding API interactions and exposing resources.</p>
<ul style="list-style-type: none"> Dernière Version du provider AWS Terraform 5.95.0 La version du binaire Terraform qui sera utilisée en local sera la 1.11.4 	<p>aws aws Official by: HashiCorp Public Cloud</p> <p>Lifecycle management of AWS resources, including EC2, Lambda, is maintained internally by the HashiCorp AWS Provider team.</p> <p>VERSION 5.95.0 PUBLISHED 4 days ago SOURCE CODE hashicorp/terraform-provider-aws</p>

3.5. CREATION D'UN COMPTE SUR LE PROVIDER CLOUD AWS

<ul style="list-style-type: none"> Se connecter à https://aws.amazon.com/ Région choisie : us-east-1 	<p>The screenshot shows the AWS IAM user sign-in interface on the left, which includes fields for Account ID or alias, IAM username (set to 'fabrice'), Password, and a 'Sign in' button. To the right is the Lightsail welcome screen with the Amazon logo, a 'Learn more' button, and a message about getting started.</p>
--	--

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

<ul style="list-style-type: none"> • Un compte root aura été créé ainsi qu'un compte IAM qui sera le compte de connexion utilisé pour les échanges avec terraform. 	<p>The screenshot shows the AWS Identity and Access Management (IAM) service. In the top left, there's a navigation bar with 'IAM' and 'Personnes'. Below it, the title 'Identity and Access Management (IAM)' is displayed. A search bar labeled 'Rechercher sur IAM' is present. On the right, a sidebar titled 'Personnes (1) Infos' shows a single user entry for 'Personnes'. A search bar 'Rechercher' and two filter fields ('Nom d'utilisateur' and 'Fabrice') are also visible.</p>
<ul style="list-style-type: none"> • Création de clés d'accès (Access Key ID et Secret Access Key) pour permettre à Terraform de provisionner les ressources. 	<p>The screenshot shows the 'Access Keys' section under 'Gestion des accès'. It displays a single access key entry for 'AKIAJXT...'. The key has a description 'fab-key' and was last used 'Hier'. It was last used in the 'us-east-1' region.</p>

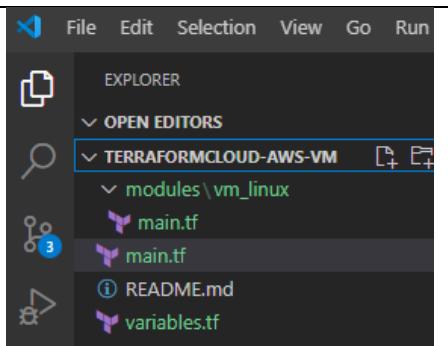
<ul style="list-style-type: none"> • Choix de la VM Linux (dans le catalogue des images Amazon) qui sera utilisée 	<p>The screenshot shows the 'AMI de démarrage rapide' section of the AWS Marketplace. It lists various AMIs, with one specific item highlighted: 'Amazon Linux 2 AMI (HVM) - Kernel 5.10, ami-07a6f770277670015 (64 bits (x86)) / Amazon Linux'. This item is marked as 'Éligible à l'offre gratuite' and 'Fournisseur vérifié'. Other filters like 'Tous les Linux/Unix' and 'Architecture' are visible.</p>
--	---

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

3.6. CREATION D'UN MODULE CONTENANT UNE VM LINUX SIMPLE

- La documentation terraform servira de référence :
<https://registry.terraform.io/providers/hashicorp/aws/latest/docs>

- L'outil **Visual Studio Code** sera utilisé pour l'écriture des fichiers de configuration .tf



The screenshot shows the Visual Studio Code interface with the Explorer sidebar open. The tree view displays a folder named 'TERRAFORMCLOUD-AWS-VM' containing a 'modules\vm_linux' folder. Inside this folder are several files: 'main.tf' (listed twice), 'README.md', and 'variables.tf'. The 'main.tf' file is currently selected.

- Actions qui seront exécutées :
 - ✓ Pas de provisionnement
 - ✓ Faire juste une initialisation du provider
 - ✓ Lancer un "terraform plan" de la machine virtuelle
(Pas besoin de provisionner quoi que ce soit, faire juste en sorte que l'initialisation du provider fonctionne et de pouvoir planifier une machine virtuelle sur le provider AWS).

a. Crédation du répertoire "modules" contenant le dossier "vm_linux"

<ul style="list-style-type: none"> Création des répertoires dans terraformcloud-aws-vm : <code>mkdir -p modules/vm_linux</code> Accès au sous-dossier : <code>cd modules/vm_linux</code> 	<ul style="list-style-type: none"> Création du fichier terraform : <code>touch main.tf</code>
<pre>\$ mkdir -p modules/vm_linux \$ cd modules/vm_linux \$ touch main.tf</pre>	

- Le fichier **main.tf** (dans "modules/vm_linux") contiendra la configuration de base pour la création d'une instance EC2 Linux (**AMI Amazon Linux 2**)

```
main.tf .\ U   main.tf ...\\vm_linux U X
```

modules > vm_linux > main.tf > ...

```
1 resource "aws_instance" "vm_linux" {
2     ami           = "ami-07a6f770277670015" # Image Amazon Linux 2
3     instance_type = "t2.micro"                 # taille de l'instance EC2 (1 vCPU/ 1 GB)
4     tags = [
5         Name = "vm_linux"
6     ]
7 }
```

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

b. Cr éation du fichier principal main.tf (./terraformcloud-aws-vm)

Ce fichier contiendra la configuration du provider AWS ainsi que le renseignement des cl es API.

<ul style="list-style-type: none"> Acc s au r pertoire racine : <pre>cd /c/terraformcloud-aws-vm</pre>	<ul style="list-style-type: none"> Cr ation du fichier terraform : touch main.tf <pre>ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm/modules/vm_linux (dev) \$ cd /c/terraformcloud-aws-vm ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (dev) \$ touch main.tf</pre>
---	--



```

main.tf > U • main.tf ...vm_linux U

main.tf > ...
1  terraform {
2    required_providers {
3      aws = {
4        source  = "hashicorp/aws"
5        version = "5.95.0" # version du provider AWS terraform
6      }
7    }
8    required_version = "1.11.4" # version du binaire terraform en local
9  }
10 provider "aws" {
11   region     = "us-east-1" # d cl ration de la r gion
12   access_key =
13   secret_key = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
14 }
15
16 module "vm_linux" {
17   source = "./modules/vm_linux" # r pertoire contenant le main.tf pour cr ation de l'instance EC2
18 }
```

c. Initialisation et plan Terraform

- Initialisation et t l chargement du plugin du provider AWS

```

ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (dev)
$ terraform init -upgrade
Initializing the backend...
Upgrading modules...
- vm_linux in modules/vm_linux
Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.95.0"...
- Installing hashicorp/aws v5.95.0...
- Installed hashicorp/aws v5.95.0 (signed by HashiCorp)
Terraform has made some changes to the provider dependency selections recorded
in the .terraform.lock.hcl file. Review those changes and commit them to your
version control system if they represent changes you intended to make.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

- Lancement du **plan terraform**

Résultat du terraform plan en [ANNEXE - 5.1](#)

Dans le cas d'un problème de connexion et d'authentification avec la clé, le message suivant apparaîtra:

```
Planning failed. Terraform encountered an error while generating this plan.

| Error: Retrieving AWS account details: validating provider credentials: retrieving call
er identity from STS: operation error STS: GetCallerIdentity, https response error Status
Code: 403, RequestID: eda17990-911c-4550-a229-ad3039670a4b, api error SignatureDoesNotMat
ch: The request signature we calculated does not match the signature you provided. Check
your AWS Secret Access Key and signing method. Consult the service documentation for deta
ils.

with provider["registry.terraform.io/hashicorp/aws"],
on main.tf line 10, in provider "aws":
10: provider "aws" {
```

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

3.7. PASSAGE DES CLES API DANS LES VARIABLES

<ul style="list-style-type: none"> Dans le répertoire racine : <pre>terraformcloud-aws-vm</pre>	<ul style="list-style-type: none"> Création des fichiers des variables : touch variables.tf et terraform.tfvars
	<pre>ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (dev) \$ ls README.md main.tf modules/ terraform.tfvars variables.tf</pre>

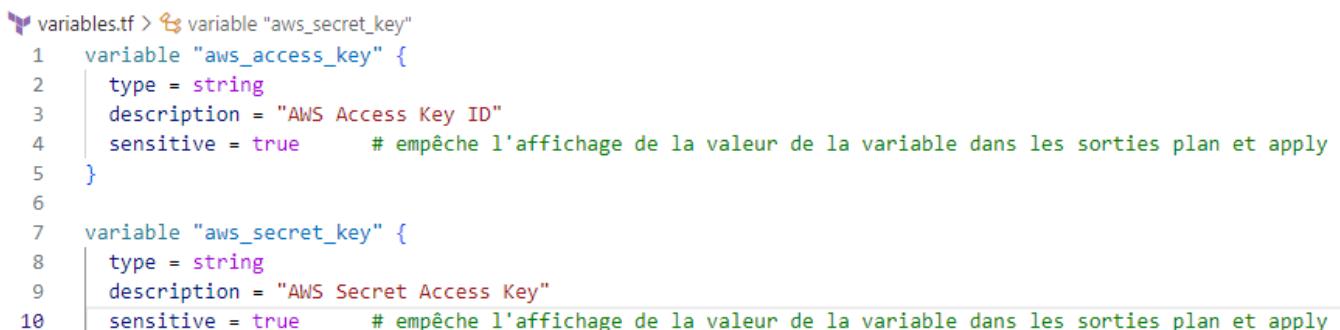
- Modification **main.tf**



```

main.tf > ...
1 < terraform {
2   required_providers {
3     aws = {
4       source  = "hashicorp/aws"
5       version = "5.95.0" # version du provider AWS terraform
6     }
7   }
8   required_version = "1.11.4" # version du binaire terraform en local
9 }
10 < provider "aws" {
11   region      = "us-east-1" # déclaration de la région
12   access_key  = var.aws_access_key
13   secret_key  = var.aws_secret_key
14 }
15
16 < module "vm_linux" {
17   source = "./modules/vm_linux" # répertoire contenant le main.tf pour création de l'instance EC2
18 }
```

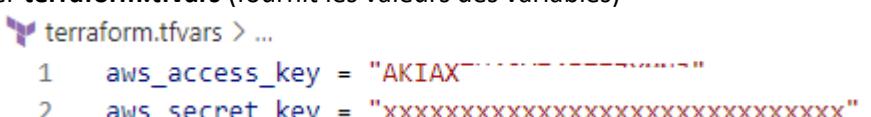
- Fichier **variables.tf** (déclaration des variables)



```

variables.tf > variable "aws_secret_key"
1   variable "aws_access_key" {
2     type = string
3     description = "AWS Access Key ID"
4     sensitive = true      # empêche l'affichage de la valeur de la variable dans les sorties plan et apply
5   }
6
7   variable "aws_secret_key" {
8     type = string
9     description = "AWS Secret Access Key"
10    sensitive = true      # empêche l'affichage de la valeur de la variable dans les sorties plan et apply
```

- Fichier **terraform.tfvars** (fournit les valeurs des variables)



```

terraform.tfvars > ...
1   aws_access_key = "AKIAXXXXXXXXXXXXXX"
2   aws_secret_key = "xxxxxxxxxxxxxxxxxxxxxxxxxxxx"
```

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

- Création du fichier **.gitignore** et ignorer les fichiers et dossier suivants :

```

1 *.exe
2 *.hcl
3 *.tfvars
4 .terraform/
5

```

3.8. PASSAGE DES CLES API VIA TERRAFORM CLOUD

a. GIT

- Suppression des valeurs **aws_access_key** et **aws_secret_key** dans le fichier **terraform.tfvars**

You, 14 minutes ago | 1 author (You)

```

1 aws_access_key = ""
2 aws_secret_key = "" You

```

- Effectuer un Commit et Push sur dev :

git status	\$ git status On branch dev Changes not staged for commit: (use "git add <file>..." to update what will be committed) (use "git restore <file>..." to discard changes in working directory) modified: .gitignore Untracked files: (use "git add <file>..." to include in what will be committed) main.tf modules/ variables.tf
git add .	ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (dev) \$ git status On branch dev Changes to be committed: (use "git restore --staged <file>..." to unstage) modified: .gitignore new file: main.tf new file: modules/vm_linux/main.tf new file: variables.tf
git commit -m "Ajout du module VM Linux"	ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (dev) \$ git commit -m "Add VM Linux Module" [dev 1f4e21d] Add VM Linux Module 4 files changed, 35 insertions(+) create mode 100644 main.tf create mode 100644 modules/vm_linux/main.tf create mode 100644 variables.tf

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

git push origin dev	<pre>ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (dev) \$ git push -u origin dev Enumerating objects: 10, done. Counting objects: 100% (10/10), done. Delta compression using up to 4 threads Compressing objects: 100% (5/5), done. Writing objects: 100% (8/8), 1.09 KiB 112.00 KiB/s, done. Total 8 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0) remote: remote: Create a pull request for 'dev' on GitHub by visiting: remote: https://github.com/fabrice-git-hub/terraformcloud-aws-vm/pull/new/dev remote: To https://github.com/fabrice-git-hub/terraformcloud-aws-vm.git * [new branch] dev -> dev branch 'dev' set up to track 'origin/dev'.</pre>
----------------------------	---

b. Terraform Cloud

<ul style="list-style-type: none"> Dans le Worspace Terraform Cloud/Variables -> Add Variable 	<p>Workspace variables (0)</p> <p>Variables defined within a workspace a</p> <div style="border: 1px solid #ccc; padding: 10px; margin-bottom: 10px;"> <p>Key</p> <p>There are no variables added.</p> </div> <p>+ Add variable</p>									
Sélectionner Terraform Variable	<p>Select variable category</p> <p><input checked="" type="radio"/> Terraform variable</p> <p>These variables should match interpolation or set a n</p>									
Création des variables aws_access_key et aws_secret_key Renseigner le champ "Value" et marquer " Sensitive "	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left; padding: 5px;">Key</th> <th style="text-align: left; padding: 5px;">Value</th> <th style="text-align: left; padding: 5px;">Category</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">aws_access_key <small>Sensitive</small></td><td style="padding: 5px;">Sensitive - write only</td><td style="padding: 5px;">terraform</td></tr> <tr> <td style="padding: 5px;">aws_secret_key <small>Sensitive</small></td><td style="padding: 5px;">Sensitive - write only</td><td style="padding: 5px;">terraform</td></tr> </tbody> </table>	Key	Value	Category	aws_access_key <small>Sensitive</small>	Sensitive - write only	terraform	aws_secret_key <small>Sensitive</small>	Sensitive - write only	terraform
Key	Value	Category								
aws_access_key <small>Sensitive</small>	Sensitive - write only	terraform								
aws_secret_key <small>Sensitive</small>	Sensitive - write only	terraform								
Exécution automatique d'un PLAN après git push origin dev Résultat du PLAN pour provisionnement en ANNEXE - 5.2	<p> fabrice-git-hub triggered a run from GitHub a few seconds ago</p> <p>Run ID: run-9kzXVe7uczFuqamP </p> <p>Configuration: From GitHub by fabrice-git-hub Branch: dev Repo: fabrice-git-hub/terraformcloud-aws-vm</p> <p>Commit: e2b17e4: terraform.tfvars modified</p> <p>Trigger: File change detected in VCS</p> <p>Execution Mode: Remote</p>									

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

4. PARTIE 2 : VAULT - PASSAGE DE SECRETS A LA VOLEE

4.1. MISE EN PLACE DE VAULT (OPTION UI CONFIGUREE)

Pour les besoins de cette évaluation, VAULT sera lancé dans un environnement Windows en mode DEV.

- Téléchargement de Vault à l'adresse : [Binaire Vault](#)
- Lancement de Vault : `vault server -dev -dev-root-token-id root`
 - ✓ Le serveur Vault démarre et fournit une adresse pour accéder à l'UI <http://127.0.0.1:8200> ainsi qu'un token de root initial et les clés de déchiffrement.
 - ✓ Le serveur de développement écoute sur l'interface de bouclage 127.0.0.1 sur le port TCP 8200, sans TLS activé par défaut.

`-dev` : Active le mode développement, Vault s'exécute en mémoire et démarre en mode "unsealed".

`-dev -dev-root-token-id root` : Initialise le jeton root, s'applique uniquement en mode « dev » et peut également être spécifié via la variable d'environnement VAULT_DEV_ROOT_TOKEN_ID.

```
c:\vault_1.19.2_windows>vault server -dev -dev-root-token-id root
=> Vault server configuration:

Administrative Namespace:
  Api Address: http://127.0.0.1:8200
    Cgo: disabled
  Cluster Address: https://127.0.0.1:8201
  Environment Variables: _ALLUSERSPROFILE, APPDATA, COMPUTERNAME, F
VER, MOZ_PLUGIN_PATH, NUMBER_OF_PROCESSORS, OS, OneDrive, PATHEXT, P
lePath, PUBLIC, Path, ProgramData, ProgramFiles, ProgramFiles(x86),
PATH, windir
  Go Version: go1.23.8
  Listener 1: tcp (addr: "127.0.0.1:8200", cluster add
ed)
  Log Level:
    Mlock: supported: false, enabled: false
  Recovery Mode: false
  Storage: inmem
  Version: Vault v1.19.2, built 2025-04-17T16:41:30Z
  Version Sha: 2ee4ea013b31a770a2fc421bb1e4bc74a9669185

=> Vault server started! Log data will stream in below:

2025-04-22T11:45:44.879+0200 [INFO] proxy environment: http_proxy=
2025-04-22T11:45:44.879+0200 [INFO] incrementing seal generation: 0
2025-04-22T11:45:44.879+0200 [WARN] no 'api_addr' value specified; i
2025-04-22T11:45:44.892+0200 [INFO] core: Initializing version hist
2025-04-22T11:45:44.892+0200 [INFO] events: Starting event system
2025-04-22T11:45:44.902+0200 [INFO] core: security barrier not initia
2025-04-22T11:45:44.902+0200 [INFO] core: security barrier initialized
2025-04-22T11:45:44.902+0200 [INFO] core: post-unseal setup starting
2025-04-22T11:45:44.924+0200 [INFO] core: loaded wrapping token key
You may need to set the following environment variables:
PowerShell:
  $env:VAULT_ADDR="http://127.0.0.1:8200"
cmd.exe:
  set VAULT_ADDR=http://127.0.0.1:8200
The unseal key and root token are displayed below in case you want to
seal/unseal the Vault or re-authenticate.
Unseal Key: ScIxwckxUwEB0h7Tz7i9qBS/dUrkIe/meYr5DL+W6DM=
Root Token: root
Development mode should NOT be used in production installations!
```

- ✓ Ajout de la variable d'environnement qui demande au client de communiquer avec le serveur en mode dev : `set VAULT_ADDR=http://127.0.0.1:8200`
- Connexion à l'UI

URL : http://127.0.0.1:8200 Token d'accès : root	 Sign in to Vault Method <input type="text" value="Token"/> Token <input type="text" value="...."/> <input type="button" value="Sign in"/>
---	--

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

4.2. CONNECTER LE MODULE TERRAFORM AU SERVEUR VAULT

Pour que Terraform interagisse avec le serveur Vault, il faut configurer le provider Vault dans le fichier de configuration **main.tf**.

Version du provider Vault : <https://registry.terraform.io/providers/hashicorp/vault/latest/docs>

- **Modification du fichier main.tf**

✓ Ajout du provider	<pre>provider "vault" { address = "http://127.0.0.1:8200" token = "root" }</pre>
✓ Ajout dans required_providers	<pre>vault = { source = "hashicorp/vault" version = "4.7.0" # version }</pre>

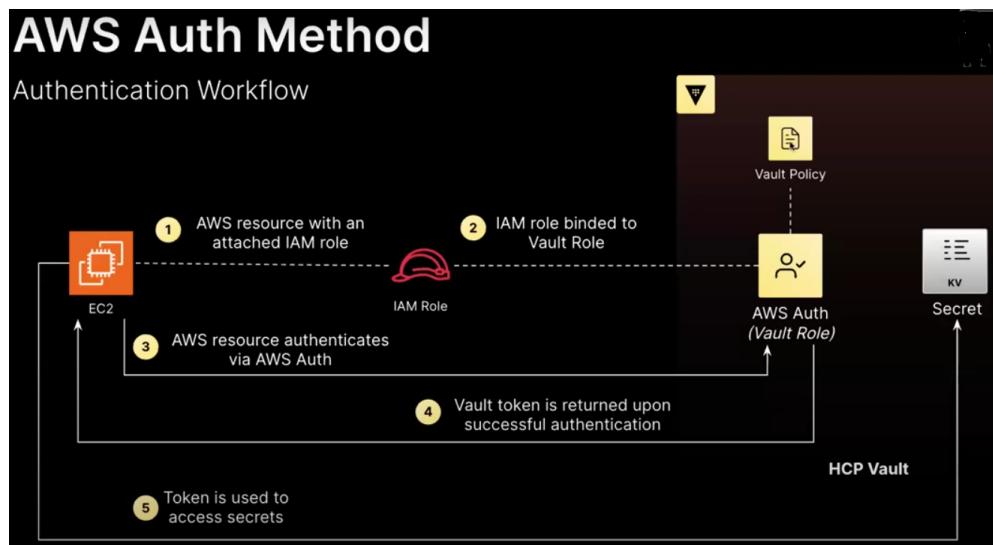
- Fichier **main.tf** complet en [ANNEXE - 5.3a](#)

4.3. COPIE DES CLES API (UNSAFE ET SEULEMENT POUR LA CORRECTION)

Copie des clés en local

```
set AWS_ACCESS_KEY_ID="AKIAXXXXXX"
set AWS_SECRET_ACCESS_KEY="XXXXXXXXXXXX"
```

4.4. CLES API RENSEIGNEES PAR VAULT



Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

a. Activer le moteur secret AWS dans Vault

Lancer la commande suivante pour générer des credentials temporaires AWS à la volée.

Ce moteur de secrets permet à Vault de générer des informations d'identification AWS dynamiques et à durée limitée.

vault secrets enable -path=aws aws

```
C:\vault_1.19.2_windows>vault secrets enable -path=aws aws
Success! Enabled the aws secrets engine at: aws/
```

b. Ajout des credentials dans Vault

- Configuration du moteur de secrets avec les clés d'accès AWS d'un utilisateur IAM disposant des permissions nécessaires :

<pre>vault write aws/config/root \ access_key=\$AWS_ACCESS_KEY_ID \ secret_key=\$AWS_SECRET_ACCESS_KEY \ region=us-east-1</pre>	<pre>C:\vault_1.19.2_windows>vault write aws/config/root access_key=\$AWS_ACCESS_KEY_ID Success! Data written to: aws/config/root C:\vault_1.19.2_windows>vault write aws/config/root secret_key=\$AWS_SECRET_ACCESS_KEY Success! Data written to: aws/config/root C:\vault_1.19.2_windows>vault write aws/config/root region=us-east-1 Success! Data written to: aws/config/root</pre>
---	--

- Détails de la configuration root du backend AWS monté dans Vault : **vault read aws/config/root**

```
C:\vault_1.19.2_windows>vault read aws/config/root
Key                                Value
---                                ---
access_key                          AKIAXTU40Y[REDACTED]
disable_automated_rotation          false
iam_endpoint                        n/a
identity_token_audience           n/a
identity_token_ttl                 0s
max_retries                         -1
region                             us-east-1
role_arn                            n/a
rotation_period                     0s
rotation_schedule                   n/a
rotation_window                     0
sts_endpoint                        n/a
stsFallback_endpoints              []
stsFallback_regions                []
sts_region                          n/a
username_template                  {{ if (eq .Type "STS") }}{{
{{ else }}{{ printf "vault-%s-%s-%s" (printf "%s-%s" (.Dis
| truncate 64 )}}{{ end }}
```

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

c. Création d'un rôle AWS dans Vault

- Configurez un rôle Vault associé à un ensemble d'autorisations et à un type d'identifiant AWS.

```
vault write aws/roles/my-role credential_type=iam_user
```

```
C:\vault_1.19.2_windows>vault write aws/roles/my-role credential_type=iam_user
Success! Data written to: aws/roles/my-role
```

Explication :

Vault saura que lorsqu'on demandera des credentials avec ce rôle (my-role), il devra créer un utilisateur IAM dans AWS.

```
vault write aws/roles/my-role policy_document=<C:\vault_1.19.2_windows\aws_role.json
```



```
C:\vault_1.19.2_windows>vault write aws/roles/my-role policy_document=<C:\vault_1.19.2_windows\aws_role.json
Success! Data written to: aws/roles/my-role
```

Explication :

Cela met à jour le rôle AWS dans Vault avec une policy IAM définie dans un fichier JSON local.

policy_document : C'est une policy IAM au format JSON, qui définit ce que les utilisateurs/roles générés par Vault pourront faire sur AWS.

Vérification de la création du rôle vault list aws/roles	C:\vault_1.19.2_windows>vault list aws/roles Keys ----- my-role
Vérification obtention des informations d'identification AWS dynamiques et temporaires sécurisées gérées par Vault, basées sur la configuration du rôle IAM my-role vault read aws/creds/my-role	C:\vault_1.19.2_windows>vault read aws/creds/my-role Key Value --- ----- lease_id aws/creds/my-role/IJuTGCsB7c9Y50MwXNz45OYV lease_duration 768h lease_renewable true access_key AKIAXTU_ secret_key JdRL5VcE session_token <nil>

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

- Vérification dans la console AWS (à chaque nouvelle connexion de Vault à AWS un utilisateur est créé)

The screenshot shows the AWS Identity and Access Management (IAM) service. On the left, there's a sidebar with links like 'Identity and Access Management (IAM)', 'Tableau de bord', 'Gestion des accès', 'Personnes', 'Rôles', 'Politiques', 'Fournisseurs d'identité', 'Paramètres du compte', and 'Gestion de l'accès racine'. The main area is titled 'Personnes (7)'. It displays a table with columns: 'Nom d'utilisateur', 'Dernière activité', 'Dernière connexion à', and 'Identifiant de la clé d''. There are seven entries listed, each with a checkbox, a user name, a timestamp, a date/time, and a status.

Nom d'utilisateur	Dernière activité	Dernière connexion à	Identifiant de la clé d'
Fabrice	Il y a 19 minutes	April 21, 2025, 18:07 (...)	Active - AKIAXTU4OY7...
vault-token-my-role-1745331196-UsMydwGOoal2zw6Q4ssq	-	-	Active - AKIAXTU4OY7...
vault-token-my-role-1745331216-xxSxkVOzER7RqlU6uuhv	-	-	Active - AKIAXTU4OY7...
vault-token-my-role-1745331436-HhvHBDquRChaPHIPY9Kn	-	-	-
vault-token-my-role-1745331873-tz07HyVe7BRHqowMpHvD	-	-	-
vault-token-terraform-my-role-1745331266-leffbg24GsF54P2FwQAg	Il y a 22 minutes	-	Active - AKIAXTU4OY7...
vault-token-terraform-my-role-1745331415-bbBurDtJCTVEtt4Zui2v	-	-	-

d. Modification du fichier main.tf

- Utilisation de la **data source Vault**

Ce bloc demande à Vault de générer et récupérer des identifiants AWS dynamiques (clé d'accès et clé secrète) pour un rôle nommé my-role.

```

15 data "vault_aws_access_credentials" "creds" {
16   backend = "aws"
17   role    = "my-role"
18 }
19 You, 5 seconds ago | 1 author (You)
20 provider "aws" {
21   region      = "us-east-1" # déclaration de la région
22   access_key = data.vault_aws_access_credentialscreds.access_key
23   secret_key = data.vault_aws_access_credentialscreds.secret_key

```

- Fichier **main.tf** complet en [ANNEXE - 5.3b](#)

e. Terraform init

```
$ terraform init
Initializing the backend...
Initializing modules...
Initializing provider plugins...
- Finding hashicorp/vault versions matching "4.7.0"...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Installing hashicorp/vault v4.7.0...
- Installed hashicorp/vault v4.7.0 (signed by HashiCorp)
- Using previously-installed hashicorp/aws v5.95.0
```

f. Terraform plan ([ANNEXE - 5.3c](#))

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

4.5. PULL REQUEST SUR MAIN(MASTER)

a. Push des modifications sur la branche DEV

git status	<pre>ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (dev) ● \$ git status On branch dev Your branch is up to date with 'origin/dev'. Changes not staged for commit: (use "git add <file>..." to update what will be committed) (use "git restore <file>..." to discard changes in working directory) modified: main.tf</pre>
git add .	<pre>ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (dev) ● \$ git add . ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (dev) ● \$ git status On branch dev Your branch is up to date with 'origin/dev'. Changes to be committed: (use "git restore --staged <file>..." to unstage) modified: main.tf</pre>
git commit -m "Added managing API Keys with Vault"	<pre>ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (dev) ● \$ git commit -m "Added managing API Keys with Vault" [dev 4604871] Added managing API Keys with Vault 1 file changed, 16 insertions(+), 2 deletions(-)</pre>
git push origin dev	<pre>\$ git push origin dev Enumerating objects: 5, done. Counting objects: 100% (5/5), done. Delta compression using up to 4 threads Compressing objects: 100% (3/3), done. Writing objects: 100% (3/3), 515 bytes 128.00 KiB/s, done. Total 3 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0) remote: Resolving deltas: 100% (2/2), completed with 2 local objects. To https://github.com/fabrice-git-hub/terraformcloud-aws-vm.git e2b17e4..4604871 dev -> dev</pre>
Vérification dans Github	

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

b. Pull Request sur Main (Master)

- Cliquer sur **Compare & pull request**

A screenshot of a GitHub repository page for 'terraformcloud-aws-vm'. The repository is private. At the top right, there is a 'Unwatch' button. Below it, a message says 'dev had recent pushes 9 minutes ago'. On the right, there is a green 'Compare & pull request' button. Below the message, there are buttons for 'dev' (selected), '2 Branches', '0 Tags', and navigation links 'Go to file', 'Add file', and 'Code'.

- Création Pull request

A screenshot of the GitHub 'Open a pull request' interface. It shows a comparison between 'base: main' and 'compare: dev'. A green checkmark indicates 'Able to merge'. The title field contains 'AWS VM deployment with API Keys managed by Vault'. The description field contains a bulleted list: '- Mise en place serveur vault avec l'option ui configurée', '- Connexion module terraform au serveur vault', '- Cés API renseignées par Vault et non les variables terraform', and '- Pull request sur master'. The bottom right has a 'Create pull request' button.

A screenshot showing the details of a pull request titled 'AWS VM deployment with API Keys managed by Vault #1'. The pull request is from 'fabrice-git-hub' to 'main' from 'dev'. The commit history shows 'fabrice-git-hub added 4 commits 18 hours ago' with changes to 'Add VM Linux Module' and 'main.tf modified'. The status summary on the right shows 'All checks have passed' (1 successful check), 'No conflicts with base branch' (Merging can be performed automatically), and a 'Merge pull request' button.

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

c. Merge Pull Request (Git CLI)

1. Mise à jour du dépôt local avec les dernières modifications : git pull origin main	<pre>ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (dev) \$ git pull origin main From https://github.com/fabrice-git-hub/terraformcloud-aws-vm * branch main -> FETCH_HEAD Already up to date.</pre>
2. Bascule vers la branche de base de la pull request : git checkout main	<pre>ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (dev) \$ git checkout main Switched to branch 'main' Your branch is up to date with 'origin/main'. ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (main)</pre>
3. Fusionnez la branche principale avec la branche de base : git merge dev	<pre>Updating ab2bd64..4604871 Fast-forward .gitignore 2 +-+ main.tf 32 ++++++===== modules/vm_linux/main.tf 7 ++++++ variables.tf 11 ++++++++ 5 files changed, 53 insertions(+), 1 deletion(-) create mode 100644 main.tf create mode 100644 modules/vm_linux/main.tf create mode 100644 variables.tf</pre>
4. Envoyez les modifications : git push -u origin main	<pre>Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0) To https://github.com/fabrice-git-hub/terraformcloud-aws-vm.git ab2bd64..4604871 main -> main branch 'main' set up to track 'origin/main'.</pre>
Etat de la pull request sur Github	 Pull request successfully merged and closed You're all set — the <code>dev</code> branch can be safely deleted.

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

5. ANNEXE

5.1. CREATION D'UN MODULE CONTENANT UNE VM LINUX SIMPLE

```
$ terraform plan
```

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:

+ create

Terraform will perform the following actions:

```
# module.vm_linux.aws_instance.vm_linux will be created
+ resource "aws_instance" "vm_linux" {
    + ami                                = "ami-07a6f770277670015"
    + arn                                = (known after apply)
    + associate_public_ip_address        = (known after apply)
    + availability_zone                  = (known after apply)
    + cpu_core_count                     = (known after apply)
    + cpu_threads_per_core              = (known after apply)
    + disable_api_stop                  = (known after apply)
    + disable_api_termination           = (known after apply)
    + ebs_optimized                      = (known after apply)
    + enable_primary_ipv6                = (known after apply)
    + get_password_data                 = false
    + host_id                            = (known after apply)
    + host_resource_group_arn            = (known after apply)
    + iam_instance_profile               = (known after apply)
    + id                                 = (known after apply)
    + instance_initiated_shutdown_behavior = (known after apply)
    + instance.lifecycle                = (known after apply)
    + instance.state                     = (known after apply)
    + instance.type                      = "t2.micro"
    + ipv6_address_count                = (known after apply)
    + ipv6_addresses                     = (known after apply)
    + key_name                           = (known after apply)
    + monitoring                         = (known after apply)
    + outpost_arn                        = (known after apply)
    + password_data                      = (known after apply)
    + placement_group                   = (known after apply)
    + placement_partition_number         = (known after apply)
    + primary_network_interface_id      = (known after apply)
    + private_dns                         = (known after apply)
    + private_ip                          = (known after apply)
    + public_dns                          = (known after apply)
    + public_ip                           = (known after apply)
```

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

```

+ secondary_private_ips          = (known after apply)
+ security_groups                = (known after apply)
+ source_dest_check              = true
+ spot_instance_request_id      = (known after apply)
+ subnet_id                      = (known after apply)
+ tags                           = {
    + "Name" = "vm_linux"
}
+ tags_all                       = {
    + "Name" = "vm_linux"
}
+ tenancy                         = (known after apply)
+ user_data                       = (known after apply)
+ user_data_base64                = (known after apply)
+ user_data_replace_on_change     = false
+ vpc_security_group_ids          = (known after apply)

+ capacity_reservation_specification (known after apply)

+ cpu_options (known after apply)

+ ebs_block_device (known after apply)

+ enclave_options (known after apply)

+ ephemeral_block_device (known after apply)

+ instance_market_options (known after apply)

+ maintenance_options (known after apply)

+ metadata_options (known after apply)

+ network_interface (known after apply)

+ private_dns_name_options (known after apply)

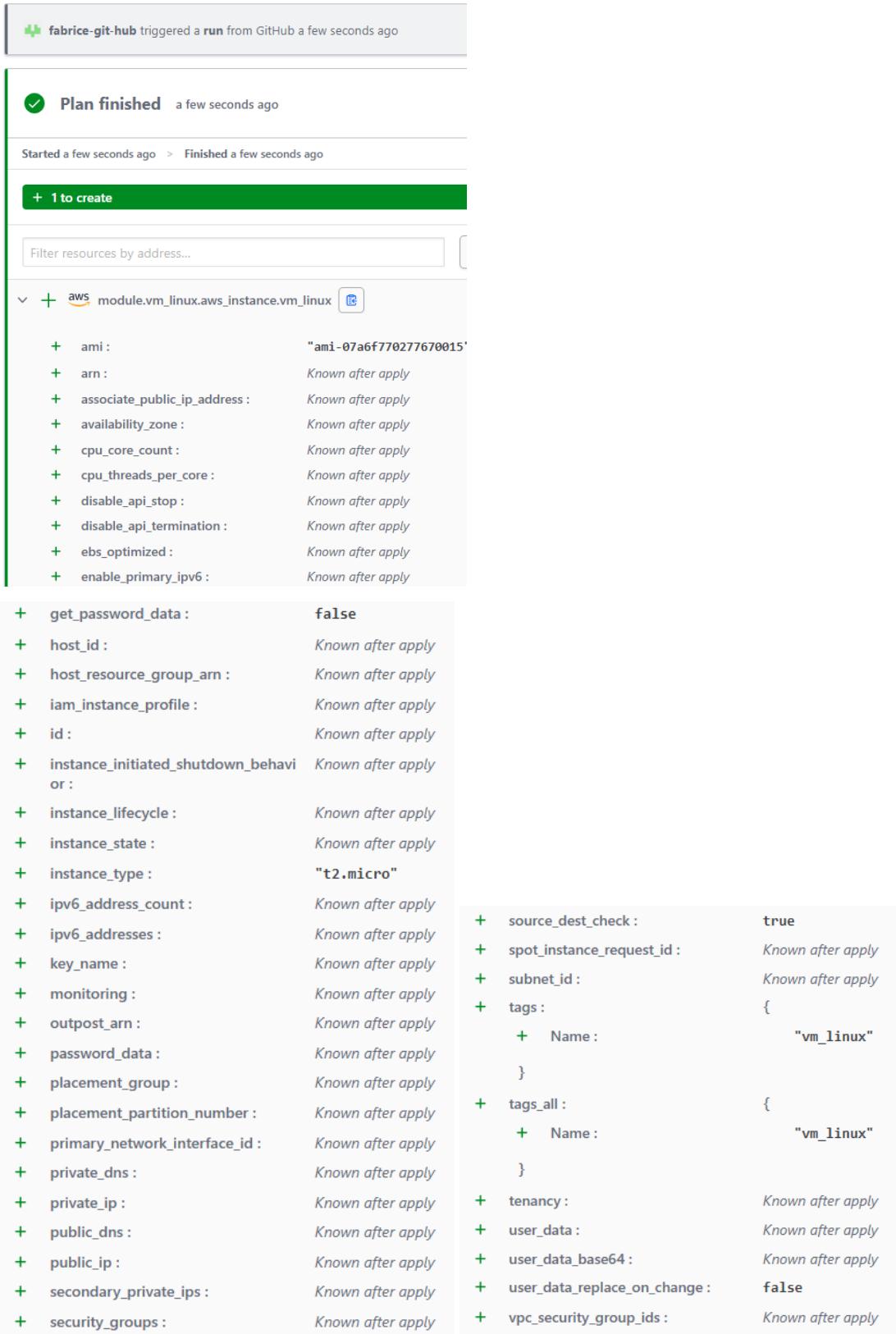
+ root_block_device (known after apply)
}


```

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

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

5.2. PASSAGE DES CLES API VIA TERRAFORM CLOUD



The screenshot shows a Terraform Cloud interface for a successful plan run. The top bar indicates the run was triggered from GitHub. The main area displays the results of the 'Plan finished' run, which started and finished a few seconds ago. A green button at the top says '+ 1 to create'. Below it is a search bar labeled 'Filter resources by address...'. The resource list shows an AWS instance module with various attributes listed:

```

+ ami : "ami-07a6f770277670015"
+ arn : Known after apply
+ associate_public_ip_address : Known after apply
+ availability_zone : Known after apply
+ cpu_core_count : Known after apply
+ cpu_threads_per_core : Known after apply
+ disable_api_stop : Known after apply
+ disable_api_termination : Known after apply
+ ebs_optimized : Known after apply
+ enable_primary_ipv6 : Known after apply

+ get_password_data : false
+ host_id : Known after apply
+ host_resource_group_arn : Known after apply
+ iam_instance_profile : Known after apply
+ id : Known after apply
+ instance_initiated_shutdown_behavior : Known after apply
+ instance_lifecycle : Known after apply
+ instance_state : Known after apply
+ instance_type : "t2.micro"
+ ipv6_address_count : Known after apply
+ ipv6_addresses : Known after apply
+ key_name : Known after apply
+ monitoring : Known after apply
+ outpost_arn : Known after apply
+ password_data : Known after apply
+ placement_group : Known after apply
+ placement_partition_number : Known after apply
+ primary_network_interface_id : Known after apply
+ private_dns : Known after apply
+ private_ip : Known after apply
+ public_dns : Known after apply
+ public_ip : Known after apply
+ secondary_private_ips : Known after apply
+ security_groups : Known after apply

```

On the right side, there is a detailed view of the 'aws_instance.vm_linux' resource, showing its configuration and state:

```

+ source_dest_check : true
+ spot_instance_request_id : Known after apply
+ subnet_id : Known after apply
+ tags :
  + Name : "vm_linux"
+
+ tags_all :
  + Name : "vm_linux"
+
+ tenancy : Known after apply
+ user_data : Known after apply
+ user_data_base64 : Known after apply
+ user_data_replace_on_change : false
+ vpc_security_group_ids : Known after apply

```

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

5.3. DECLARATION DE VAULT DANS TERRAFORM (MAIN.TF)

a. Connecter le module Terraform au serveur Vault

```

main.tf > terraform > required_providers > vault
You, 4 seconds ago | 1 author (You)
1  terraform {
    You, 4 seconds ago | 1 author (You)
2    required_providers {
        You, 13 hours ago | 1 author (You)
3      aws = {
4        source  = "hashicorp/aws"
5        version = "5.95.0" # version du provider AWS terraform
6      }
    You, 4 seconds ago | 1 author (You)
7      vault = {
8        source  = "hashicorp/vault"
9        version = "4.7.0" # version du provider VAULT terraform
10       You
11    }
12    required_version = "1.11.4" # version du binaire terraform en local
13  }
You, 12 hours ago | 1 author (You)
14  provider "aws" {
15    region      = "us-east-1" # déclaration de la région
16    access_key = var.aws_access_key
17    secret_key = var.aws_secret_key
18  }
19
You, 4 seconds ago | 1 author (You)
20  provider "vault" {
21    address = "http://127.0.0.1:8200"
22    token   = "root"
23  }
24
You, 13 hours ago | 1 author (You)
25  module "vm_linux" {
26    source = "./modules/vm_linux" # répertoire contenant le main.tf pour
27  }

```

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

b. Clés API renseignées par Vault

```

└ main.tf > module "vm_linux"
  You, 14 minutes ago | 1 author (You)
1   terraform {
  You, 14 minutes ago | 1 author (You)
2     required_providers {
      You, 15 hours ago | 1 author (You)
3       aws = {
4         source  = "hashicorp/aws"
5         version = "5.95.0" # version du provider AWS terraform
6       }
      You, 14 minutes ago | 1 author (You)
7       vault = {
8         source  = "hashicorp/vault"
9         version = "4.7.0" # version du provider VAULT terraform
10        }
11      }
12     required_version = "1.11.4" # version du binaire terraform en local
13   }
14

  You, 14 minutes ago | 1 author (You)
15   data "vault_aws_access_credentials" "creds" {
16     backend = "aws"
17     role    = "my-role"
18   }
  You, 14 minutes ago | 1 author (You)
19   provider "aws" {
20     region      = "us-east-1" # déclaration de la région
21     access_key = data.vault_aws_access_credentialscreds.access_key
22     secret_key = data.vault_aws_access_credentialscreds.secret_key
23   }
  You, 14 minutes ago | 1 author (You)
25   provider "vault" {
26     address = "http://127.0.0.1:8200"
27     token   = "root"
28   }

  You, 15 hours ago | 1 author (You)
30   module "vm_linux" [
31     source = "./modules/vm_linux" # répertoire contenant le main.tf pour création de l'instance EC2
32   ] You, 15 hours ago • Add VM Linux Module ...
33

```

Evaluation	Parcours	ADMINISTRATEUR SYSTEME DEVOPS Sécuriser l'Infrastructure	Auteur	FT
ECF3	Titre		Du	10/11/2025

c. Terraform plan

```
ulukai@DESKTOP-8VK8LT7 MINGW64 /c/terraformcloud-aws-vm (dev)
● $ terraform plan
data.vault_aws_access_credentials.creds: Reading...
data.vault_aws_access_credentials.creds: Still reading... [10s elapsed]
data.vault_aws_access_credentials.creds: Read complete after 17s [id=aws/creds]

Terraform used the selected providers to generate the following execution plan.
+ create

Terraform will perform the following actions:

# module.vm_linux.aws_instance.vm_linux will be created
+ resource "aws_instance" "vm_linux" {
    + ami                                = "ami-07a6f770277670015"
    + arn                                = "(known after apply)"
    + associate_public_ip_address        = "(known after apply)"
    + availability_zone                  = "(known after apply)"
    + cpu_core_count                     = "(known after apply)"
    + cpu_threads_per_core              = "(known after apply)"
    + disable_api_stop                 = "(known after apply)"
    + disable_api_termination          = "(known after apply)"
    + ebs_optimized                     = "(known after apply)"
    + enable_primary_ipv6               = "(known after apply)"
    + get_password_data                = false
    + host_id                            = "(known after apply)"
    + host_resource_group_arn           = "(known after apply)"
    + iam_instance_profile              = "(known after apply)"
    + id                                 = "(known after apply)"
    + instance_initiated_shutdown_behavior = "(known after apply)"
    + instance_lifecycle                = "(known after apply)"
    + instance_state                   = "(known after apply)"
    + instance_type                     = "t2.micro"
    + ipv6_address_count                = "(known after apply)"
    + ipv6_addresses                    = "(known after apply)"
    + key_name                           = "(known after apply)"
    + monitoring                        = "(known after apply)"
    + outpost_arn                       = "(known after apply)"
    + password_data                     = "(known after apply)"
    + placement_group                  = "(known after apply)"

+ primary_network_interface_id      = "(known after apply)"
+ private_dns                       = "(known after apply)"
+ private_ip                         = "(known after apply)"
+ public_dns                         = "(known after apply)"
+ public_ip                          = "(known after apply)"
+ secondary_private_ips            = "(known after apply)"
+ security_groups                   = "(known after apply)"
+ source_dest_check                 = true
+ spot_instance_request_id          = "(known after apply)"
+ subnet_id                          = "(known after apply)"
+ tags                               = {
    + "Name" = "vm_linux"
  }
+ tags_all                           = {
    + "Name" = "vm_linux"
  }
+ tenancy                            = "(known after apply)"
+ user_data                          = "(known after apply)"
+ user_data_base64                  = "(known after apply)"
+ user_data_replace_on_change       = false
+ vpc_security_group_ids             = "(known after apply)"

+ capacity_reservation_specification (known after apply)

+ cpu_options (known after apply)

+ ebs_block_device (known after apply)

+ enclave_options (known after apply)

+ ephemeral_block_device (known after apply)

+ instance_market_options (known after apply)

+ maintenance_options (known after apply)

+ metadata_options (known after apply)

+ network_interface (known after apply)

+ private_dns_name_options (known after apply)

+ root_block_device (known after apply)
}

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