Terraform

Déploiement de Laboratoires Virtuels avec Terraform et Debian 12

Introduction

Ce projet a pour but de vous familiariser avec Terraform, un outil d'Infrastructure as Code (IaC), en l'utilisant pour automatiser le déploiement et la gestion de machines virtuelles Debian 12 sous VMware Workstation Pro. Vous apprendrez à définir votre infrastructure sous forme de code, ce qui permet une reproductibilité, une rapidité et une gestion simplifiée de vos environnements de laboratoire.

Préparation de l'environnement

création d'une VM debian 12.

requis : serveur ssh , pas d'interface graphique , Open-VM-Tools installés , utilisateur standard avec accès `sudo` .

Notez le chemin complet vers le fichier `.vmx` de cette VM (ex: `C:\VMs\Debian12-Base\Debian12-Base.vmx` ou /home/user/vmware/Debian12-Base/Debian12-Base.vmx). C'est votre "template".

Installation de Terraform

Installation de Terraform CLI et Lancement du Provider VMware Workstation

Créez un répertoire pour votre projet Terraform:

“terraform_debian_lab”

Premier déploiement de VM Debian 12

Création du fichier de configuration (`main.tf`)

Exécution des commandes Terraform (init , plan , apply)

Amélioration et gestion du cycle de vie

Utilisation de variables.

Gestion des modifications

Destruction de l'infrastructure

Laboratoire de Cybersécurité simple

Déploiement de plusieurs VMs

Mise en place d'un scénario de base :

* Créez un fichier `main.tf` qui déploie deux VMs Debian 12:

* VM 1 (Attaquant) : Nommez-la `kali-like-vm`. Vous pouvez y installer quelques outils de base via un `remote-exec` provisioner (ex: `sudo apt install -y nmap masscan`).

Note : Pour un vrai Kali, il faudrait une image de base Kali. Ici, c'est une Debian avec des outils.

* VM 2 (Victime) : Nommez-la `victim-vm`. Installez un service vulnérable simple via un `remote-exec` provisioner (ex: un serveur web Python simple sur le port 8080 : `python3 -m http.server 8080 &`).

* Assurez-vous que les deux VMs sont sur le même réseau virtuel (ex: "VMnet8").

* Après `terraform apply`, vérifiez que la VM "attaquante" peut atteindre la VM "victime" et interagir avec le service vulnérable.

Pour aller plus loin

vous pouvez intégrer dans le lab des scripts Ansible pour améliorer le déploiement .

Compétences visées

- Concevoir une solution technique répondant à des besoins d'évolution de l'infrastructure
- Mettre en production des évolutions de l'infrastructure

Rendu

Le projet est à rendre sur votre github :

https://github.com/prenom-nom/terraform

* Fournissez les codes Terraform (fichiers `.tf`, `.tfvars`).

* Expliquez les commandes Terraform utilisées et leur effet.

* Décrivez les difficultés rencontrées et comment vous les avez résolues.

* Décrivez le scénario de cybersécurité mis en place et comment les VMs interagissent.

Base de connaissances

- terraform-provider-vmworkstation
- Terraform Registry
- Terraform Proxmox Provider
- Terraform Installation.