0. A chaque demarrage

encryption passphrase

username

password

Vérifier la Connexion SSH: sudo systemctl status ssh

sudo netstat -tulnp | grep 4242

verif Port 4242 ouvert : sudo ufw status  
connecter via SSH en utilisant un autre terminal : ssh -p 4243 aumartin@<127.0.0.1>

1. Vérification des Partitions

lsblk (= lister les périphériques de bloc)

NAME : Le nom du périphérique ou de la partition.

MAJ : Les numéros majeurs et mineurs du périphérique.

RM : Indicateur de périphérique amovible (1 pour oui, 0 pour non).

SIZE : La taille du périphérique ou de la partition.

RO : Indicateur de lecture seule (1 pour oui, 0 pour non).

TYPE : Le type de périphérique (disk pour un disque dur, part pour une partition, lvm pour un volume logique, etc.).

MOUNTPOINT : Le point de montage où la partition est montée dans le système de fichiers.

Options Utiles

lsblk -f : Affiche les systèmes de fichiers des partitions.

lsblk -l : Affiche la sortie sous forme de liste plutôt que de tableau.

lsblk -o NAME,SIZE,FSTYPE,MOUNTPOINT : Affiche des colonnes spécifiques (nom, taille, type de système de fichiers, et point de montage).

Commandes Complémentaires

df -h : Affiche l'espace disque utilisé et disponible sur les systèmes de fichiers.

fdisk -l : Liste les tables de partition.

blkid : Affiche les identifiants des systèmes de fichiers et les types de systèmes de fichiers.

2. Configurating Your Virtual Machine

# 2.1. Installing Sudo

su -

su = (swich user) changer d'utilisateur dans un terminal.

Sans argument, elle change pour l'utilisateur root par défaut.

- : L'option - (ou --login) fait que l'environnement de l'utilisateur cible (ici root) est initialisé comme s'il s'agissait d'une nouvelle connexion. Cela inclut la mise à jour des variables d'environnement, le répertoire de travail et les chemins d'accès.

=> Effectuer des configurations nécessitant des privilèges administratifs (= effectuer des modifications critiques sur le système)

Cela peut inclure des tâches comme :

Création et gestion de partitions.

Configuration des services systèmes comme SSH, UFW (pare-feu), etc.

Installation et configuration de logiciels.

apt-get update -y

apt-get update : Mettre à jour la liste des paquets disponibles (apt = Advanced Package Tool) dans les dépôts logiciels configurés sur votre système Debian.

=> votre système contacte les dépôts de logiciels configurés pour obtenir la liste actuelle des paquets disponibles.

Cette liste est ensuite stockée localement sur votre système pour que le gestionnaire de paquets puisse l'utiliser pour les futures opérations d'installation, de mise à niveau ou de suppression de logiciels.

Garantir la Fraîcheur des Informations : M'assure que votre système utilise les informations les plus récentes sur les paquets disponibles.

-y : L'option -y permet d'automatiser le processus de MaJ en répondant automatiquement "oui" à toutes les questions posées par la commande.

Cela est utile lorsque vous exécutez la commande dans un script ou lorsque vous voulez éviter les interruptions lors de la mise à jour régulière de votre système.

apt-get upgrade -y

Mettre à niveau tous les paquets installés sur votre système Debian vers leurs versions les plus récentes disponibles dans les dépôts logiciels configurés.

=> Mise à Jour des Paquets : Lorsque de nouvelles versions de logiciels sont disponibles dans les dépôts, apt-get upgrade met à jour les paquets installés sur votre système avec ces nouvelles versions.

Cela permet de garantir que votre système utilise les dernières fonctionnalités, correctifs de bugs et améliorations de sécurité.

Sécurité : Les mises à jour de sécurité sont souvent incluses dans les nouvelles versions des logiciels. En exécutant régulièrement apt-get upgrade, vous vous assurez que votre système est protégé contre les vulnérabilités connues.

Stabilité : Les mises à jour de logiciels peuvent également inclure des correctifs de bugs qui améliorent la stabilité et les performances de votre système.

apt install sudo

Installer le programme sudo sur votre système.

sudo (abréviation de "SuperUser Do")

=> permet à un utilisateur d'exécuter des commandes avec les privilèges d'un autre utilisateur, généralement l'utilisateur root.

nstaller sudo est généralement une étape importante lors de la configuration d'un nouveau système Linux, car cela permet de mettre en place un système d'administration sécurisé et de limiter l'utilisation directe du compte root, ce qui peut réduire les risques de sécurité.

usermod -aG sudo your\_username OU adduser <username> sudo

usermod : commande utilisée pour modifier les attributs d'un utilisateur.

-aG sudo : option de la commande usermod qui ajoute l'utilisateur spécifié au groupe sudo.

-a signifie "append" (ajouter), et -G spécifie le groupe.

your\_username : C'est le nom de l'utilisateur auquel vous souhaitez accorder des privilèges d'administration avec sudo.

=> l'utilisateur spécifié pourra utiliser sudo pour exécuter des commandes avec les privilèges d'administration. Cela permet de limiter l'utilisation directe du compte root, ce qui est généralement considéré comme une pratique plus sécurisée.

Commandes Complémentaires (pour verif)

dpkg -l | grep sudo : (Debian Package) Lister tous les paquets installés sur votre système | filtrer les résultats de la commande afin de ne montrer que les lignes qui contiennent le mot "sudo".

getent group sudo : Affiche les informations du groupe sudo.

groups your\_username : Affiche les groupes auxquels appartient un utilisateur spécifique, y compris le groupe sudo s'il y est ajouté.

OPTION faite maison non obligatoire : j'ai Installe net-tools pour vérifier les ports sur ma VM :

sudo apt install net-tools

# 2.2. Installing and Configuring SSH (Secure Shell Host)

## 2.2.1 Installing

sudo apt install openssh-server

sudo : préfixe exécute la commande avec des privilèges administratifs (superutilisateur ou root). Cela est nécessaire car l'installation de logiciels affecte le système d'exploitation et nécessite des autorisations élevées.

apt : Il s'agit de l'outil de gestion de paquets pour les distributions Debian et basées sur Debian, comme Ubuntu. Il est utilisé pour gérer les paquets logiciels, c'est-à-dire les installer, les mettre à jour et les supprimer.

install : Ce sous-commande indique à apt que vous souhaitez installer un paquet logiciel.

openssh-server : nom du paquet que vous voulez installer.

"OpenSSH" est une suite de connectivité pour chiffrer les communications réseau en utilisant le protocole SSH, et "server" indique qu'il s'agit du composant serveur.

=> Télécharge le paquet openssh-server et ses dépendances à partir des dépôts de logiciels configurés.

Installe le paquet sur votre système.

Configure le serveur SSH pour qu'il puisse démarrer automatiquement avec le système et être prêt à accepter les connexions SSH.

sudo systemctl status ssh => check l'état actuel du service SSH

systemctl : commande qui contrôle le système init de Linux (systemd) qui est utilisé pour gérer les services et d'autres aspects du système, y compris le démarrage, l'arrêt et la vérification de l'état des services.

status : option de la commande systemctl qui indique que nous voulons vérifier l'état actuel du service spécifié.

ssh : C'est le nom du service pour lequel nous voulons obtenir l'état.

Commandes Complémentaires (pour verif):

dpkg -l | grep ssh : (Debian Package) vérifier si le package openssh-server (ou tout autre package lié à SSH) est bien installé sur votre système

installation de vim car chiant d'utiliser vi:

apt-get install vim -y

## 2.2.2 Config

sudo vim /etc/ssh/sshd\_config

sudo : C'est une commande utilisée sous Unix et Linux pour exécuter des commandes avec les privilèges de superutilisateur (root). Elle permet d'exécuter vi avec des droits élevés nécessaires pour modifier des fichiers système comme /etc/ssh/sshd\_config.

vim : C'est un éditeur de texte en mode terminal (= vim version -ameliore) très puissant et largement utilisé sous Unix et Linux. Il est connu pour sa flexibilité et ses fonctionnalités avancées, bien qu'il puisse être intimidant pour les utilisateurs novices.

/etc/ssh/sshd\_config : C'est le chemin du fichier de configuration du démon SSH (sshd). Ce fichier contient les paramètres de configuration pour le service SSH sur votre système. Modifier ce fichier permet de personnaliser le comportement et les options de configuration du service SSH.

Changer #port22 en port 4242

Le # est utilisé pour commenter une ligne, ce qui signifie que la ligne est ignorée par le programme.

En retirant le #, vous activez la configuration du port 4242 pour SSH.

Changer #PermitRootLogin prohibit-password en PermitRootLogin no

pour modifier la conf SSH pour désactiver la connexion en tant que root

sudo systemctl restart ssh

sudo systemctl status ssh

## 2.2.3 - Installing and Configuring UFW (Uncomplicated Firewall)

apt-get install ufw => installe UFW OU sudo apt install ufw

ATTENTION LE PORT 22 est active

revenir dessus

sudo ufw enable pour desactiver UFW

sudo ufw status

sudo ufw allow ssh

sudo ufw allow 4242

## 2.2.4 - Connecting to Server via SSH

J'ai choisi methode : Connexion directe via adresse IP et port

Si ça fonctionne c’est mieux, cest plus portable

Là c'est une IP interne à ma machine. Le réseau ça fonctionne via IP que tu sois en interne à ta machine ou sur internet, comme ça tout fonctionne pareil.

Par contre oui ça changera à chaque démarrage de ta VM donc il faut savoir où la trouver

Une VM a une IP propre. Mais qui n’est connue que de ta machine hôte (l’ordi sur lequel tu as démarré la VM)

ip a | grep inet OU ip a show enp0s3 | grep inet

lo (loopback) : Utilisée pour la communication interne du système, IP 127.0.0.1.

enp0s3 (Ethernet) : Interface réseau utilisée pour les connexions externes, avec IP 10.0.2.15 dans votre cas.

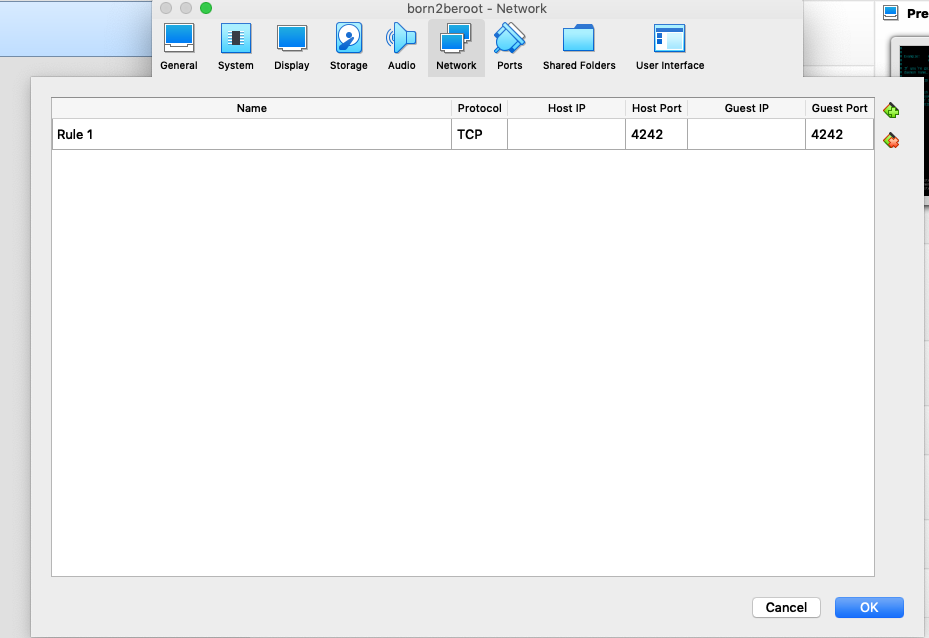
ssh aumartin@127.0.0.1 -p 4242

ssh : Lance le client SSH pour se connecter à un serveur distant via le protocole SSH.

<username> : Remplacez <username> par votre nom d'utilisateur sur la VM.

-p 4242 : Spécifie que le port 4242 doit être utilisé pour la connexion SSH.

logout ou exit pour déconnecter et terminer la connexion SSH établie avec le serveur distant.

J'ai du faire un Port forwarding  


OU  
  
commande : ssh -L 4243:localhost:4242 aumartin@10.0.2.15

Justification pour l'utilisation du Port Forwarding

L'utilisation du port forwarding est justifiée dans votre cas car le port 4242 semble déjà utilisé ou bloqué sur votre réseau local. En redirigeant le port 4242 de votre machine virtuelle vers un autre port (comme 4243) sur votre machine hôte, vous contournez les problèmes de conflits de port et vous pouvez toujours accéder à votre machine virtuelle via SSH de manière sécurisée.

Tester la connexion a distance

Ouvrir un terminal (hote)

ssh aumartin@127.0.0.1 -p 4243

Si ca marche pas : rm ~/.ssh/known\_hosts dans iTerm puis retape ssh aumartin@127.0.0.1 -p 4243  
exit pour quitter la connexion

3. User Management

# 3.1 Mettre en place une politique de mot de passe fort

## 3.1.1 Password Age

sudo vi /etc/login.defs

=> ouvre le fichier /etc/login.defs en utilisant l'éditeur de texte vi avec des privilèges administratifs (sudo). Ce fichier contient les paramètres de configuration liés aux comptes d'utilisateurs sur le système Linux, notamment les règles de gestion des mots de passe comme la durée de validité, les avertissements de changement de mot de passe, etc.

tu changes les lignes dans la section Password aging controls

160 PASS\_MAX\_DAYS 30

161 PASS\_MIN\_DAYS 2

162 PASS\_WARN\_AGE 7

## 3.1.2 Password Strength

sudo apt install libpam-pwquality

libpam-pwquality: Ce paquet contient des modules PAM (Pluggable Authentication Modules) pour améliorer la sécurité des mots de passe en permettant de définir des politiques de force de mot de passe, comme la longueur minimale, les caractères requis, etc.

=> installer la bibliothèque de vérification de la qualité des mots de passe, libpam-pwquality, sur un système Linux

dpkg -l | grep libpam-pwquality

man pam\_pwquality

sudo vi /etc/pam.d/common-password

=> ouvre le fichier common-password situé dans le répertoire /etc/pam.d/ en mode édition avec l'éditeur de texte vi.

Ce fichier est utilisé par le module PAM (Pluggable Authentication Modules) pour configurer les politiques de mot de passe communes pour différents services système.

ajouter des options a la ligne

password requisite pam\_pwquality.so retry=3

minlen=10 => len de 10 caracteres

ucredit=-1 dcredit=-1 => contenir au moins une lettre majuscule ("uppercase credit") && ontenir au moins un chiffre (digit credit)

La valeur -1 signifie que cette exigence doit être respectée.

maxrepeat=3 => max 3 caractères consecutifs identiques

reject\_username => rejeter contient le username

difok=7 => comporter au moins 7 caractères qui ne sont pas présents dans l’ancien mot de passe.

enforce\_for\_root => votre mot de passe root devra suivre cette politique.

sudo reboot => redémarrer votre système pour que ces modifications prennent effet de manière cohérente

# 3.2 Configuration du groupe sudoers

sudo visudo

\* Defaults env\_reset (deja presente default)

=> Cette option réinitialise l'environnement pour une commande exécutée via sudo. Cela signifie que l'environnement de l'utilisateur ne sera pas transféré à la commande exécutée avec sudo, assurant ainsi une isolation et une sécurité accrues.

\* Defaults mail\_badpass (deja presente default)

=> Lorsqu'un utilisateur entre un mot de passe incorrect lors de l'utilisation de sudo, une notification par courrier électronique est envoyée à l'administrateur système. Cela aide à surveiller et à détecter les tentatives d'utilisation non autorisée de sudo.

\* Defaults secure\_path="/usr/local/sbin:/usr/local/bin:/usr/bin:/sbin:/bin"

=> Cette option définit les chemins d'exécution sécurisés pour les commandes exécutées via sudo. Seules les commandes situées dans ces répertoires seront disponibles pour l'exécution via sudo, limitant ainsi les risques associés à l'exécution de commandes potentiellement malveillantes.

\* Defaults badpass\_message="Le mot de passe est incorrect, veuillez réessayer !"

=> message qui sera affiché à l'utilisateur si un mot de passe incorrect est entré lors de l'utilisation de sudo. Cela permet de fournir une rétroaction claire à l'utilisateur sur la raison de l'échec de l'authentification.

\* Defaults passwd\_tries=3

=> Cette option limite le nombre de tentatives autorisées pour entrer un mot de passe correct lors de l'utilisation de sudo. Après trois tentatives infructueuses, sudo refusera l'accès jusqu'à ce qu'une nouvelle session soit démarrée.

\* Defaults logfile="/var/log/sudo/sudo.log" (=> journal unique)

=> pécifie le chemin du fichier journal dans lequel toutes les actions exécutées via sudo seront enregistrées. Cela permet une traçabilité complète des commandes exécutées avec sudo.

Pour un script comme monitoring.sh qui collecte diverses informations système à intervalles réguliers (toutes les 10 minutes), l'utilisation d'un fichier journal unique (sudo.log) pourrait être plus pratique et suffisante. Cela centralise les données et simplifie la maintenance et l'analyse.

\* Defaults log\_input, log\_output

=> activent la journalisation des entrées (log\_input) et des sorties (log\_output) des commandes exécutées via sudo. Cela inclut les commandes entrées par l'utilisateur ainsi que leurs résultats.

\* Defaults requiretty

Activation le mode TTY (Teletype) pour sudo

=> Cette option requiert qu'une console soit associée à la commande sudo pour qu'elle puisse être exécutée. Cela empêche l'exécution de commandes sudo à partir de scripts ou de processus sans terminal associé, ce qui ajoute une couche supplémentaire de sécurité.

options :

Defaults insults

Defaults passprompt="N'oubliez pas de sourire :)" : Cela modifie le message de prompt de mot de passe de sudo pour inclure une remarque amicale. C'est une façon subtile d'ajouter une touche personnelle.

Defaults lecture : Cette option affiche un message humoristique ou informatif à chaque utilisation de sudo. Cela peut ajouter un peu de légèretévirt à l'utilisation quotidienne de sudo.

Utilisez cd /var/log pour accéder au répertoire des journaux.

sudo mkdir /var/log/sudo

sudo touch /var/log/sudo/sudo.log

# 3.3 Changement des mots de passe existants

sudo passwd root

sudo passwd aumartin

# 3.4 Création d'un groupe

sudo addgroup user42

sudo groupadd evaluating

getent group => verifier group

OU groups

Ajouter un user au groupe

sudo adduser <username> user42

OU sudo usermod -aG user42 <username>

getent group user42 => verif qui fait parti d'un groupe

# 3.5 Création d'un new user

cut -d: -f1 /etc/passwd => check all local users

sudo adduser <username>

getent passwd <username> => verifi si user bien créé

chage -l <username> => check si règle mdp fonctionnent pour le user

Supprimez les utilisateurs non nécessaires : sudo deluser <nouvel\_utilisateur>

# 3.6 Modifier le hostname

Vérifiez le hostname : hostname  
Modifier le fichier /etc/hostname : sudo nano /etc/hostname

Modifier le fichier /etc/hosts : sudo nano /etc/hosts

Recherchez la ligne avec votre ancien hostname et modifiez-la pour le nouveau. Par exemple :

127.0.1.1 ancien\_hostname

en :  
127.0.1.1 new\_hostname

OU  
appliquer immédiatement le nouveau hostname : sudo hostnamectl set-hostname new\_hostname

4. Verification avant la partie script nommé monitoring.sh

Vérifier la Connexion SSH: sudo systemctl status ssh

sudo netstat -tulnp | grep 4242

verif Port 4242 ouvert : sudo ufw status  
connecter via SSH en utilisant un autre terminal : ssh -p 4243 aumartin@<127.0.0.1>

contrôle connexion ssh en tant que root est interdite : ssh -p 4243 root@127.0.0.1

Vérifiez que aumartin appartient aux groupes sudo et user42 : groups aumartin

Assurez-vous que le groupe user42 existe : getent group user42

Vérifier les Règles de Mot de Passe : sudo chage -l aumartin

Vérifier le Pare-feu UFW : sudo ufw status

5. Snapshot

**Faire un Snapshot :**

* + Faites un snapshot de votre machine virtuelle après avoir vérifié que tout fonctionne correctement. Cela vous permet de revenir à un état stable en cas de problème futur.
  + Pour créer un snapshot :
    - Dans VirtualBox, sélectionnez votre machine virtuelle, cliquez sur "Snapshots" et choisissez "Take Snapshot".
  + Faites cela après avoir confirmé que toutes les configurations sont correctes.

6. Script monitoring.sh

# 6.1 Objectif du script monitoring.sh

Le script monitoring.sh doit collecter périodiquement des informations système et les afficher sur tous les terminaux connectés (via la commande wall). Voici les informations que le script doit collecter et afficher :

1. **Informations système basiques :**
   * Architecture du système d'exploitation et sa version de kernel.
   * Nombre de processeurs physiques.
   * Nombre de processeurs virtuels.
2. **Utilisation de la mémoire :**
   * Mémoire vive disponible et son taux d'utilisation en pourcentage.
   * Mémoire disponible (y compris le swap) et son taux d'utilisation en pourcentage.
3. **Utilisation du processeur :**
   * Taux d'utilisation actuel des processeurs en pourcentage.
4. **Informations sur le système :**
   * Date et heure du dernier redémarrage.
   * État de LVM (actif ou pas).
5. **Statistiques sur les utilisateurs et les connexions :**
   * Nombre de connexions actives.
   * Nombre d'utilisateurs connectés.
6. **Réseau :**
   * Adresse IPv4 du serveur.
   * Adresse MAC (Media Access Control) du serveur.
7. **Sécurité et administration :**
   * Nombre de commandes exécutées avec sudo.

# 6.2 Étapes pour développer monitoring.sh

1. **Collecte des informations :**
   * Utilisez des commandes comme uname -mrs pour l'architecture et la version du kernel.
   * grep 'physical id' /proc/cpuinfo | sort -u | wc -l pour le nombre de processeurs physiques.
   * grep 'processor' /proc/cpuinfo | wc -l pour le nombre de processeurs virtuels.
   * free -m pour la mémoire vive disponible.
   * df -h pour la mémoire disponible (y compris le swap).
   * top -bn1 | grep 'Cpu(s)' pour le taux d'utilisation du processeur.
   * who -b pour la date et l'heure du dernier redémarrage.
   * sudo lvs --noheadings | wc -l pour vérifier si LVM est actif.
   * sudo netstat -tn | grep ESTABLISHED | wc -l pour le nombre de connexions actives.
   * who | wc -l pour le nombre d'utilisateurs connectés.
   * hostname -I pour l'adresse IPv4.
   * ip link show | grep ether pour l'adresse MAC.
2. **Écriture du script :**
   * Utilisez des variables pour stocker les résultats des commandes.
   * Formatez les résultats pour les rendre lisibles et compréhensibles.
   * Utilisez des boucles, des conditions et des commandes d'assignation pour organiser et formater les informations collectées.
3. **Affichage sur les terminaux :**
   * Utilisez la commande wall pour diffuser les informations collectées sur tous les terminaux connectés.
   * Assurez-vous que les informations sont bien formatées et lisibles.
4. **Planification avec cron :**
   * Ajoutez une entrée dans cron pour exécuter le script toutes les 10 minutes.
   * Utilisez crontab -e pour éditer la table cron et ajouter votre commande.

# 6.3 Réflexion et débogage

Pendant le développement, assurez-vous de :

* Tester chaque commande individuellement pour vous assurer qu'elle renvoie les résultats attendus.
* Utiliser des variables pour stocker les résultats intermédiaires afin de les manipuler et de les formater facilement.
* Vérifier que le script fonctionne sans erreurs en le testant sur votre machine virtuelle.
* Utiliser des outils de débogage comme echo pour vérifier les valeurs des variables à différentes étapes du script.

En suivant ces étapes et en comprenant chaque partie du script, vous serez en mesure de le développer efficacement et de répondre aux exigences de votre projet.

# 6.4 Lectures et Ressources Recommandées

Pour mieux comprendre les concepts utilisés dans ce script et les commandes associées, voici quelques lectures et ressources que je recommande :

1. **Manuels et Pages de manuel (man pages)** :
   * Les pages de manuel sont une excellente ressource pour comprendre les commandes Unix/Linux. Par exemple, man uname, man grep, man awk, etc.
2. **Livres** :
   * **"The Linux Command Line" par William Shotts** : Un excellent livre pour débuter avec les commandes Linux.
   * **"Advanced Bash-Scripting Guide" par Mendel Cooper** : Un guide détaillé pour apprendre le scripting bash.
   * **"Linux System Programming" par Robert Love** : Une ressource avancée pour comprendre le noyau Linux et la programmation système.
3. **Cours en ligne** :
   * **"Introduction to Linux" sur edX par la Linux Foundation** : Un cours complet pour apprendre les bases de Linux.
   * **"Bash Scripting and Shell Programming" sur Udemy** : Un cours vidéo pour apprendre le scripting en bash.
4. **Documentation en ligne** :
   * **The Linux Documentation Project (TLDP)** : Une ressource en ligne avec de nombreux guides et how-tos.
   * **Stack Overflow et forums Linux** : Des communautés en ligne où tu peux poser des questions et apprendre des autres.
5. **Pratique et Expérimentation** :
   * Créer de petits scripts pour automatiser des tâches simples et expérimenter avec différentes commandes et options.
   * Utiliser des environnements virtuels pour tester des scripts sans risquer de perturber ton système principal.

### **Comment coder le script sans aide**

Pour coder le script sans aide, voici une approche détaillée :

1. **Décomposer les exigences** : Lire les consignes et identifier chaque information demandée.
2. **Rechercher les commandes nécessaires** : Utiliser des manuels (man command), rechercher en ligne et lire des guides pour trouver les commandes Unix/Linux qui récupèrent les informations demandées.
3. **Tester chaque commande individuellement** : Avant d'intégrer les commandes dans le script, les exécuter dans le terminal pour s'assurer qu'elles fonctionnent comme prévu.
4. **Écrire le script étape par étape** : Commencer par un simple script qui récupère et affiche une seule information, puis ajouter progressivement les autres éléments.
5. **Déboguer et optimiser** : Utiliser des messages de débogage (echo) pour vérifier que chaque partie du script fonctionne correctement, puis optimiser le script en éliminant les redondances et en améliorant l'efficacité.
6. **Automatiser avec cron** : Une fois le script fonctionnel, configurer cron pour l'exécuter automatiquement à intervalles réguliers.

# 6.5 Réalisation du script

## 6.5.1 Installation de cron

sudo apt install cron

Après l'installation, cron devrait normalement démarrer automatiquement. Si ce n'est pas le cas, vous pouvez démarrer et activer le service cron en utilisant les commandes suivantes :  
sudo systemctl start cron  
sudo systemctl enable cron

## 6.5.2 Configuration de cron

Configuration de Cron pour exécuter le script toutes les 10 minutes  
Ouvrir crontab pour l'utilisateur root

sudo crontab -u root -e

Ajouter la ligne suivante pour exécuter le script toutes les 10 minutes :  
\*/10 \* \* \* \* /usr/local/bin/monitoring.sh

Check root's scheduled cron jobs via sudo crontab -u root -l.

sudo crontab -u root -l

**Avantages** :

* Direct et simple : il suffit d'ajouter une ligne dans crontab.
* Pas besoin de spécifier l'interpréteur de script (sh ou bash) car la commande complète est spécifiée.

OU j’ai choisi plutôt

Création du script, la modification de sudoers pour l'exécuter sans mot de passe, et la configuration de cron dans un ensemble d'instructions plus détaillées.

**Avantages** :

* Détaillé et inclusif pour ceux qui pourraient nécessiter des étapes supplémentaires comme la création du script, les permissions sudo, etc.
* Couvre toutes les étapes du processus, y compris le reboot après la modification de sudoers.

dpkg -s net-tools pour vérifier si j’ai netstat tools sinon apt-get install -y net-tools  
  
création du script et modif des droits  
cd /usr/local/bin/  
sudo touch monitoring.sh  
sudu chmod 777 monitoring.sh

Connexion via mon terminal hote  
ssh username@127.0.0.1 -p 4243  
  
Edition du script sur terminal hôte (car tu peux copie colle)  
cd /usr/local/bin  
nano monitoring.sh

exit  
  
Sur Virtual Machine (not iTerm)   
sudo visudo

Ajoute cette ligne   
your\_username ALL=(ALL) NOPASSWD: /usr/local/bin/monitoring.sh endesous %sudo ALL=(ALL:ALL) ALL

sudo reboot

sudo /usr/local/bin/monitoring.sh pour exécuter le script en su (super user)

sudo crontab -u root -e

Ajouter la ligne suivante pour exécuter le script toutes les 10 minutes :  
\*/10 \* \* \* \* /usr/local/bin/monitoring.sh

A SAVOIR : \*/1 \* \* \* \* bash /chemin/vers/votre/script/monitor.sh ( = pour 1 min)  
Toutes les 30 secondes :  
\* \* \* \* \* sleep 30 && bash /chemin/vers/votre/script/monitor.sh

### **Explication :**

* \* \* \* \* \* : Cette partie spécifie la fréquence à laquelle la tâche cron sera exécutée. Chaque astérisque représente une unité de temps différente (minute, heure, jour du mois, mois, jour de la semaine).
  + Le premier \* : minute (chaque minute)
  + Le deuxième \* : heure (chaque heure)
  + Le troisième \* : jour du mois (chaque jour du mois)
  + Le quatrième \* : mois (chaque mois)
  + Le cinquième \* : jour de la semaine (chaque jour de la semaine)
* sleep 30 && bash /chemin/vers/votre/script/monitor.sh : sleep 30 permet d'attendre 30 secondes avant d'exécuter la commande suivante (&& bash /chemin/vers/votre/script/monitor.sh).

## 6.5.3 Script détaillé

#!/bin/bash

echo "Début de l'exécution du script monitoring.sh"

# Architecture et version du kernel

echo "Récupération de l'architecture du système et de la version du kernel"  
arc=$(uname -a)

echo "Architecture et version du kernel : $arc"  
  
Alternative : arc=$(uname -srm) pour obtenir des informations plus concises.

Gestion des erreurs :  
if [ $? -ne 0 ]; then   
echo "Erreur : Impossible de récupérer l'architecture du système." >&2   
exit 1   
fi

# Nombre de processeurs physiques

echo "Récupération du nombre de processeurs physiques"  
pcpu=$(grep "physical id" /proc/cpuinfo | sort | uniq | wc -l)

echo "Nombre de processeurs physiques : $pcpu"

# Nombre de processeurs virtuels

echo "Récupération du nombre de processeurs virtuels"  
vcpu=$(grep "^processor" /proc/cpuinfo | wc -l)

echo "Nombre de processeurs virtuels : $vcpu"  
  
Alternative : Utilisation de lscpu pour obtenir des informations détaillées sur le CPU.  
pcpu=$(lscpu | grep "^Socket(s):" | awk '{print $2}')  
vcpu=$(lscpu | grep "^CPU(s):" | awk '{print $2}')

Gestion des erreurs :  
if [ $? -ne 0 ]; then   
echo "Erreur : Impossible de récupérer le nombre de processeurs physiques." >&2   
exit 1   
fi

# Mémoire vive et taux d'utilisation

echo "Récupération de la mémoire vive et de son taux d'utilisation"  
fram=$(free -m | awk '$1 == "Mem:" {print $2}')  
uram=$(free -m | awk '$1 == "Mem:" {print $3}')  
pram=$(free | awk '$1 == "Mem:" {printf("%.2f"), $3/$2\*100}')

echo "Mémoire totale : ${fram}MB, Mémoire utilisée : ${uram}MB, Pourcentage d'utilisation : ${pram}%"

Alternative : Utilisation de vmstat pour obtenir des statistiques de mémoire.  
fram=$(vmstat -s | grep "total memory" | awk '{print $1}')  
uram=$(vmstat -s | grep "used memory" | awk '{print $1}')  
pram=$(echo "scale=2; $uram/$fram\*100" | bc)  
  
Gestion des erreurs :  
if [ -z "$fram" ] || [ -z "$uram" ] || [ -z "$pram" ]; then   
echo "Erreur : Impossible de récupérer les informations sur la mémoire." >&2   
exit 1   
fi

# Espace disque et taux d'utilisation

echo "Récupération de l'espace disque et de son taux d'utilisation"  
fdisk=$(df -BG | grep '^/dev/' | grep -v '/boot$' | awk '{ft += $2} END {print ft}')  
udisk=$(df -BM | grep '^/dev/' | grep -v '/boot$' | awk '{ut += $3} END {print ut}')  
pdisk=$(df -BM | grep '^/dev/' | grep -v '/boot$' | awk '{ut += $3} {ft+= $2} END {printf("%d"), ut/ft\*100}')

echo "Espace disque total : ${fdisk}GB, Espace disque utilisé : ${udisk}MB, Pourcentage d'utilisation : ${pdisk}%"

Alternative : Utilisation de stat -f pour obtenir des informations sur le système de fichiers.  
fdisk=$(stat -f / | awk '{print $4 \* $2 / 1024 / 1024}')  
udisk=$(df -m / | awk 'NR==2 {print $3}')  
pdisk=$(df -m / | awk 'NR==2 {print $5}')

Gestion des erreurs :  
if [ -z "$fdisk" ] || [ -z "$udisk" ] || [ -z "$pdisk" ]; then   
echo "Erreur : Impossible de récupérer les informations sur le disque." >&2 exit 1 fi

# Utilisation du CPU

echo "Récupération du taux d'utilisation du CPU"

cpul=$(top -bn1 | grep '^%Cpu' | cut -c 9- | xargs | awk '{printf("%.1f%%"), $1 + $3}')

echo "Taux d'utilisation du CPU : $cpul"

Alternative : Utilisation de mpstat pour obtenir des informations sur l'utilisation du CPU.  
cpul=$(mpstat | awk '$12 ~ /[0-9.]+/ { printf("%.1f%%\n", 100 - $12) }')

if [ -z "$cpul" ]; then   
echo "Erreur : Impossible de récupérer la charge CPU." >&2   
exit 1   
fi

# Date et heure du dernier redémarrage

echo "Récupération de la date et l'heure du dernier redémarrage"

lb=$(who -b | awk '$1 == "system" {print $3 " " $4}')

echo "Dernier redémarrage : $lb"

Alternative : Utilisation de uptime pour obtenir la durée depuis le dernier redémarrage.  
lb=$(uptime -s)

Gestion des erreurs :  
if [ -z "$lb" ]; then   
echo "Erreur : Impossible de récupérer la date de dernier démarrage." >&2   
exit 1   
fi

# LVM actif ou non

echo "Vérification si LVM est actif ou non"

lvmu=$(if [ $(lsblk | grep "lvm" | wc -l) -eq 0 ]; then echo no; else echo yes; fi)

echo "LVM actif : $lvmu"

Gestion des erreurs :

lvmu=$(if [ $(lsblk | grep "lvm" | wc -l) -eq 0 ]; then echo no; else echo yes; fi)

# Nombre de connexions actives

echo "Récupération du nombre de connexions TCP actives"  
ctcp=$(ss -Ht state established | wc -l)

echo "Nombre de connexions TCP actives : $ctcp"

Gestion des erreurs :   
if [ -z "$ctcp" ]; then

echo "Erreur : Impossible de récupérer le nombre de connexions TCP établies." >&2   
exit 1   
fi

# Nombre d'utilisateurs connectés

echo "Récupération du nombre d'utilisateurs connectés"

ulog=$(users | wc -w)

echo "Nombre d'utilisateurs connectés : $ulog"

Gestion des erreurs :

if [ -z "$ulog" ]; then   
echo "Erreur : Impossible de récupérer le nombre d'utilisateurs connectés." >&2   
exit 1   
fi

# Adresse IPv4 et MAC

echo "Récupération de l'adresse IPv4 et de l'adresse MAC"

ip=$(hostname -I)

mac=$(ip link show | grep "ether" | awk '{print $2}')

echo "Adresse IPv4 : $ip, Adresse MAC : $mac"

Gestion des erreurs :

if [ -z "$ip" ] || [ -z "$mac" ]; then  
echo "Erreur : Impossible de récupérer l'adresse IP ou l'adresse MAC." >&2  
exit 1  
fi

# Nombre de commandes sudo exécutées

echo "Récupération du nombre de commandes exécutées avec sudo"

cmds=$(journalctl \_COMM=sudo | grep COMMAND | wc -l)

echo "Nombre de commandes sudo exécutées : $cmds"

Gestion des erreurs :

if [ -z "$cmds" ]; then

echo "Erreur : Impossible de récupérer le nombre de commandes sudo exécutées." >&2

exit 1

fi

# Affichage des informations

echo "Affichage des informations récupérées avec wall"

wall " #Architecture: $arc

#CPU physical: $pcpu

#vCPU: $vcpu

#Memory Usage: $uram/${fram}MB ($pram%)

#Disk Usage: $udisk/${fdisk}Gb ($pdisk%)

#CPU load: $cpul

#Last boot: $lb

#LVM use: $lvmu

#Connections TCP: $ctcp ESTABLISHED

#User log: $ulog

#Network: IP $ip ($mac)

#Sudo: $cmds cmd"

echo "Fin de l'exécution du script monitoring.sh"

## 6.5.4 Vérification

1. **Tester le script manuellement** :  
   sudo /usr/local/bin/monitoring.sh
2. **OPTION : Vérifier les logs de cron** :

* Les logs de cron peuvent être consultés dans /var/log/syslog ou /var/log/cron.log (selon la configuration du système).

sudo tail -f /var/log/syslog | grep CRON

## 6.5.5 **Explications pour la Soutenance**

1. **Comment le script récupère et affiche chaque information demandée** :
   * Le script utilise des commandes Unix/Linux (uname, grep, awk, free, df, top, who, lsblk, ss, hostname, ip, journalctl) pour récupérer les informations demandées et les affiche à l'aide de la commande wall.
2. **Pourquoi chaque commande est utilisée** :
   * uname -a : Récupère l'architecture du système et la version du kernel.
   * grep et awk sur /proc/cpuinfo : Comptent les processeurs physiques et virtuels.
   * free -m : Affiche l'utilisation de la mémoire en Mo.
   * df -BG et df -BM : Affichent l'utilisation de l'espace disque en Go et Mo.
   * top -bn1 : Affiche l'utilisation du CPU.
   * who -b : Affiche la date et l'heure du dernier redémarrage.
   * lsblk : Vérifie si LVM est actif.
   * ss -Ht state established : Compte les connexions TCP actives.
   * users : Compte les utilisateurs connectés.
   * hostname -I et ip link show : Affichent l'adresse IP et MAC.
   * journalctl \_COMM=sudo : Compte les commandes sudo exécutées.
   * wall " " dans le script permet d'envoyer un message à tous les terminaux connectés. Cela affiche les informations récupérées par le script sur tous les écrans des utilisateurs connectés, ce qui est conforme à la consigne du projet de diffuser les informations toutes les 10 minutes.
3. **Comment cron est configuré** :
   * La ligne \*/10 \* \* \* \* /usr/local/bin/monitoring.sh dans crontab configure l'exécution du script toutes les 10 minutes.
4. **Vérification du bon fonctionnement** :
   * Tester le script manuellement.
   * Vérifier les logs de cron pour s'assurer que le script s'exécute sans erreur.
5. **Configuration de sudoers** :
   * Modifier le fichier sudoers pour permettre l'exécution du script sans mot de passe avec la ligne : <your\_username> ALL=(ALL) NOPASSWD: /usr/local/bin/monitoring.sh.

7 - Signature.txt

⚠️ Warning: before you generate a signature number, turn off your Virtual Machine. ⚠️

1. **Arrête ta machine virtuelle** :
   * Assure-toi que ta machine virtuelle est complètement éteinte avant de générer la signature.
2. **Accède au dossier de ta VM** :
   * Ouvre le terminal (iTerm si tu es sur macOS).
   * Tape cd pour revenir à ton répertoire home.
   * Navigue vers le dossier contenant ta VM : cd /sgoinfre/goinfre/Perso/aumartin
3. **Génère la signature** :
   * Utilise la commande shasum pour calculer la signature de ton fichier .vdi :  
     shasum VirtualBox.vdi  
     Remplace VirtualBox.vdi par le nom exact de ton fichier .vdi.
4. **Crée et soumets le fichier signature.txt** :

Copie le numéro de la signature générée.

Crée un fichier signature.txt dans le répertoire racine de ton dépôt Git cloné.

Colle le numéro de la signature dans ce fichier :  
echo "<signature\_number>" > signature.txt

Assure-toi que le fichier signature.txt est bien présent à la racine de ton dépôt Git et soumets-le.

La commande shasum est utilisée pour calculer et vérifier les sommes de contrôle SHA (Secure Hash Algorithm) pour les fichiers. Il s'agit d'un outil de hachage qui génère une empreinte numérique unique pour un fichier, ce qui permet de vérifier l'intégrité des fichiers.

Voici ce que fait la commande shasum et quelques détails supplémentaires :

**Utilisation de shasum**

* **Calcul de la somme de contrôle** : Lorsque tu exécutes shasum sur un fichier, elle calcule une somme de contrôle SHA pour ce fichier et affiche le résultat.
* **Vérification de la somme de contrôle** : Tu peux également utiliser shasum pour vérifier une somme de contrôle par rapport à une valeur connue pour assurer que le fichier n'a pas été modifié.

**Syntaxe**

shasum [options] [fichier]

**Options courantes**

* **-a** : Spécifie l'algorithme SHA à utiliser (par exemple, 1, 224, 256, 384, 512). Par défaut, shasum utilise SHA-1.

shasum -a 256 fichier.txt

**-c** : Vérifie les sommes de contrôle par rapport à celles répertoriées dans un fichier.  
shasum -c fichier\_de\_somme\_de\_controle

**Conclusion**

La commande shasum est utile pour assurer l'intégrité des fichiers en générant et vérifiant des sommes de contrôle. Pour ton projet, l'utilisation de shasum permet de créer un fichier signature.txt contenant l'empreinte numérique unique de ton fichier VM, ce qui est crucial pour valider l'authenticité et l'intégrité de ta machine virtuelle.

8. Résumé des commandes essentielles

* **Comparer les signatures** : diff signature.txt .vdi\_signature\_file
* **Vérifier les services** : sudo systemctl status ufw, sudo systemctl status ssh
* **Ajouter un utilisateur** : sudo adduser <username>
* **Ajouter un groupe** : sudo groupadd evaluating, sudo usermod -aG evaluating <username>
* **Voir les partitions** : lsblk, df -h
* **Vérifier les logs sudo** : ls /var/log/sudo/, cat /var/log/sudo/some\_log\_file
* **Configurer cron** : sudo crontab -u root -e  
  Modifs toutes les 30 secondes :  
  \* \* \* \* \* sleep 30 && bash /chemin/vers/votre/script/monitor.sh
* **Vérifier les logs de cron** : sudo tail -f /var/log/cron.log
* **Gestion UFW** : sudo ufw status, sudo ufw allow 8080, sudo ufw delete allow 8080
* **Configurer et vérifier SSH** : sudo nano /etc/ssh/sshd\_config, sudo systemctl restart ssh, ssh <username>@<ip\_address> -p 4243