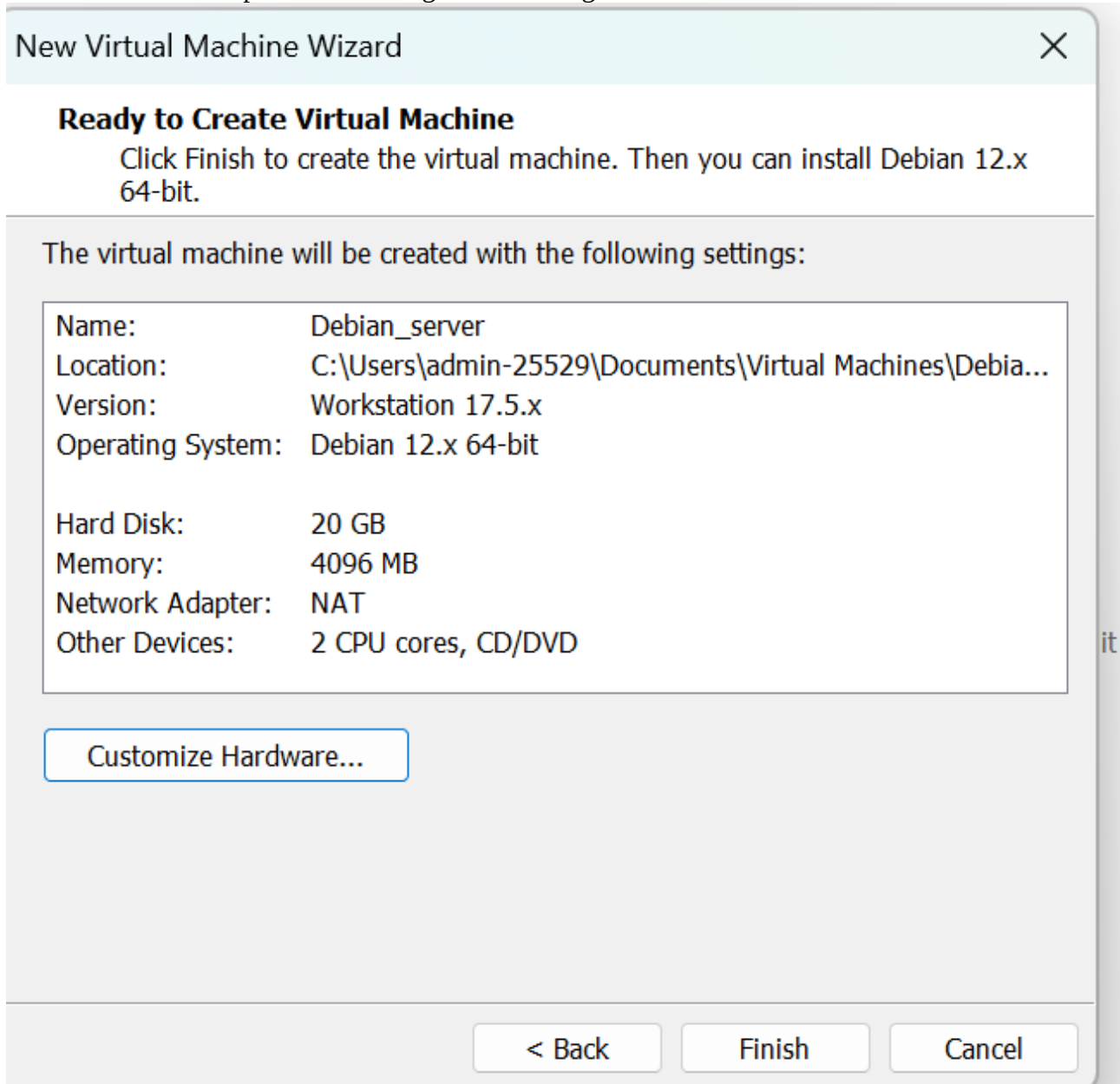


Systemes, Scripts et Sécurité

1. Création d'une VM Debian

création d'une VM à partir d'une image iso téléchargée Debian 12



2. Commandes de recherche avancée

On peut créer un script qui va créer les fichiers texte avec le même contenu.

Pour trouver le mot contenant « force », on utilise cette commande :

```
find . -type f -name "mon_texte.txt" -exec grep -l "force" {} +
```

Cette commande va rechercher tous les fichiers « mon_texte.txt » à partir de mon répertoire et afficher les noms des fichiers qui contiennent le mot « force »

```

GNU nano 7.2 mon_script.sh
#script pour automatiser la création de fichier
fichier_texte="mon_texte.txt"
contenu_texte="Que la force soit avec toi."

#boucle pour créer les fichiers dans chaque répertoire
for repertoire in Bureau Documents Téléchargements Vidéos Images; do
    echo "$contenu_texte" > "$repertoire/$fichier_texte"
done

```

[Lecture de 8 lignes]

^G Aide ^O Écrire ^W Chercher ^K Couper ^T Exécuter ^C Emplacement
 ^X Quitter ^R Lire fich. ^\ Remplacer ^U Coller ^J Justifier ^_ Aller ligne

3. Compression et décompression de fichiers

Pour dupliquer le fichier mon_texte.txt quatre fois dans le répertoire Plateforme, on peut utiliser la commande :

```
for i in {1..4}; do cp mon_texte.txt "mon_texte_$i.txt"; done
```

```

GNU nano 7.2 mon_texte.txt *
#Dupliquer le fichier "mon_texte.txt" quatre fois dans le même répertoire.

for i in {1..4}; do
    cp mon_texte.txt "mon_texte_$i.txt"
done

```

ou utiliser un script.

Pour la compression du répertoire Plateforme on utilise « tar » et l'option « -z » pour la compression gzip :

```
tar -czf Plateforme.tar.gz Plateforme
-j : utilise la compression bzip2 au lieu de gzip.
```

- - J : utilise la compression xz.
- - Z : niveau de compression gzip (1-9, par défaut 6).
- - - lzma : utilise la compression LZMA.

Décompression de l'archive

```
tar -xf Plateforme.tar.gz
```

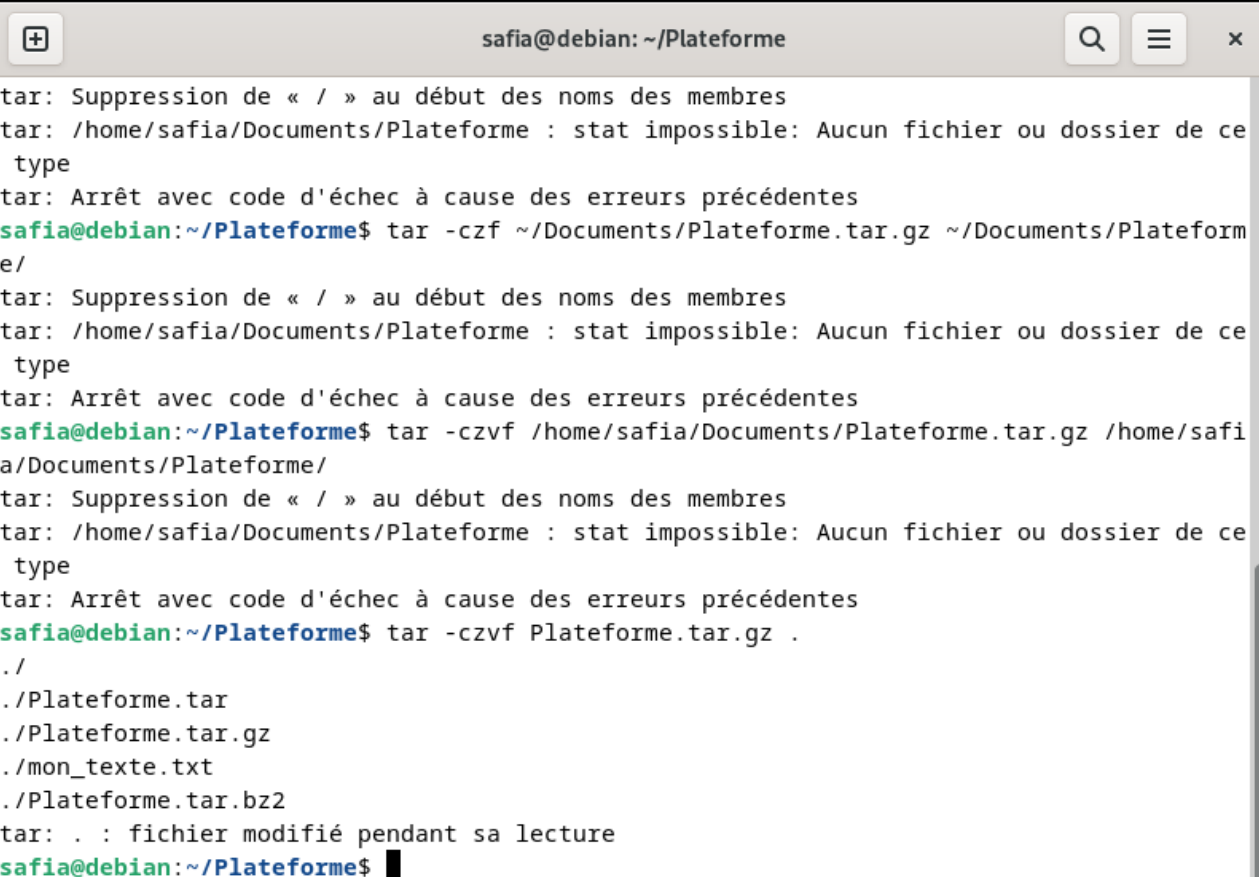
-C : affiche le contenu de l'archive sans la décompresser.

- -t : affiche la table des matières de l'archive.
- -v : affiche des informations détaillées pendant la décompression.
- -p : permet de décompresser l'archive dans un chemin spécifique.

```
gzip mon_texte.txt gunzip mon_texte.txt.gz
```

```
bzip2 mon_texte.txt bunzip2 mon_texte.txt.bz2
```

```
xz mon_texte.txt xz -d mon_texte.txt.xz
```



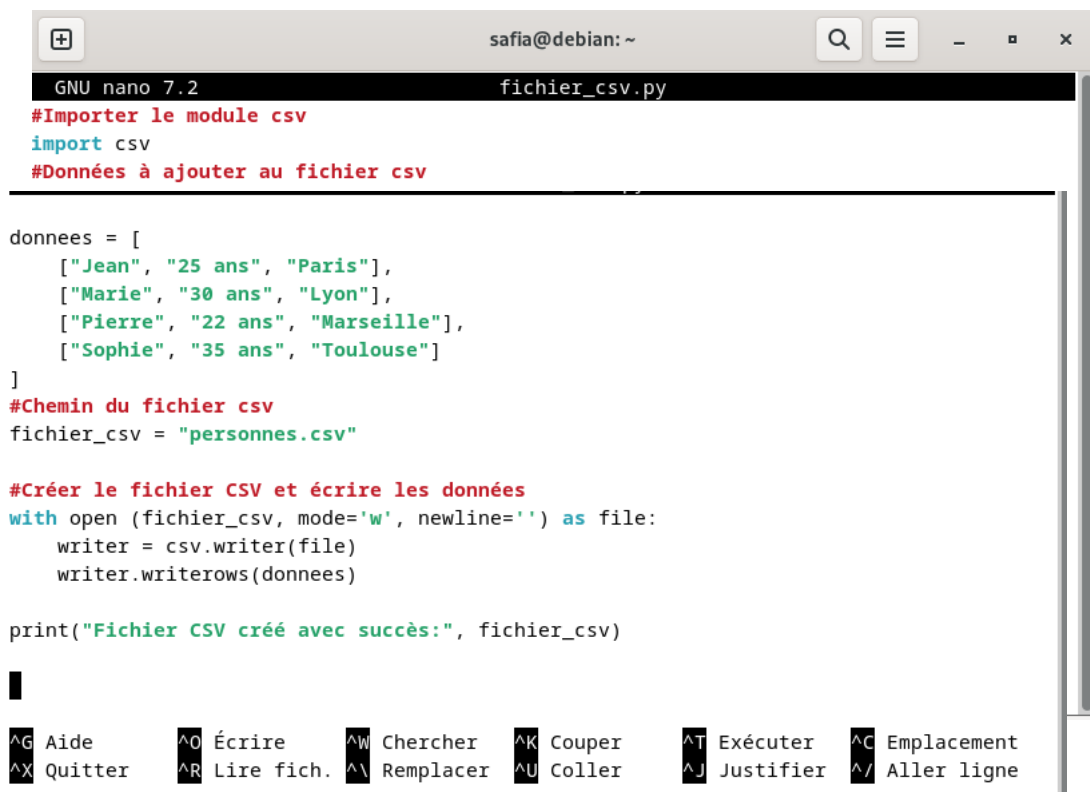
A terminal window titled 'safia@debian: ~/Plateforme' with search, menu, and close buttons. The terminal shows the following sequence of commands and output:

```
tar: Suppression de « / » au début des noms des membres
tar: /home/safia/Documents/Plateforme : stat impossible: Aucun fichier ou dossier de ce type
tar: Arrêt avec code d'échec à cause des erreurs précédentes
safia@debian:~/Plateforme$ tar -czf ~/Documents/Plateforme.tar.gz ~/Documents/Plateforme/
tar: Suppression de « / » au début des noms des membres
tar: /home/safia/Documents/Plateforme : stat impossible: Aucun fichier ou dossier de ce type
tar: Arrêt avec code d'échec à cause des erreurs précédentes
safia@debian:~/Plateforme$ tar -czvf /home/safia/Documents/Plateforme.tar.gz /home/safia/Documents/Plateforme/
tar: Suppression de « / » au début des noms des membres
tar: /home/safia/Documents/Plateforme : stat impossible: Aucun fichier ou dossier de ce type
tar: Arrêt avec code d'échec à cause des erreurs précédentes
safia@debian:~/Plateforme$ tar -czvf Plateforme.tar.gz .
./
./Plateforme.tar
./Plateforme.tar.gz
./mon_texte.txt
./Plateforme.tar.bz2
tar: . : fichier modifié pendant sa lecture
safia@debian:~/Plateforme$
```

```
--format=gnu -f -b20 --quoting-style=escape --rmt-command=/usr/sbin/rmt
--rsh-command=/usr/bin/rsh
safia@debian:~/Plateforme$ file -b mon_fichier.txt
cannot open `mon_fichier.txt' (No such file or directory)
safia@debian:~/Plateforme$ tar -tvf Plateforme.tar.gz
safia@debian:~/Plateforme$ ls
mon_texte.txt  Plateforme.tar  Plateforme.tar.bz2  Plateforme.tar.gz
safia@debian:~/Plateforme$ file Plateforme.tar.gz
Plateforme.tar.gz: gzip compressed data, from Unix, original size modulo 2^32 10240
safia@debian:~/Plateforme$ file Plateforme.tar
Plateforme.tar: gzip compressed data, from Unix, original size modulo 2^32 10240
safia@debian:~/Plateforme$ file Plateforme.tar.bz2
Plateforme.tar.bz2: bzip2 compressed data, block size = 900k
safia@debian:~/Plateforme$ gunzip Plateforme.tar.gz
gzip: Plateforme.tar already exists; do you wish to overwrite (y or n)? y
safia@debian:~/Plateforme$ tar -xvf Plateforme.tar.gz
tar: Plateforme.tar.gz : open impossible: Aucun fichier ou dossier de ce type
tar: Error is not recoverable: exiting now
safia@debian:~/Plateforme$ ls
mon_texte.txt  Plateforme.tar  Plateforme.tar.bz2
safia@debian:~/Plateforme$ tar -xvf Plateforme.tar
safia@debian:~/Plateforme$ bzip2 -d Plateforme.tar.bz2
bzip2: Output file Plateforme.tar already exists.
safia@debian:~/Plateforme$
```

4. Manipulation de texte

Création d'un fichier CSV avec python



```
GNU nano 7.2      fichier_csv.py
#Importer le module csv
import csv
#Données à ajouter au fichier csv

donnees = [
    ["Jean", "25 ans", "Paris"],
    ["Marie", "30 ans", "Lyon"],
    ["Pierre", "22 ans", "Marseille"],
    ["Sophie", "35 ans", "Toulouse"]
]
#Chemin du fichier csv
fichier_csv = "personnes.csv"

#Créer le fichier CSV et écrire les données
with open (fichier_csv, mode='w', newline='') as file:
    writer = csv.writer(file)
    writer.writerows(donnees)

print("Fichier CSV créé avec succès:", fichier_csv)

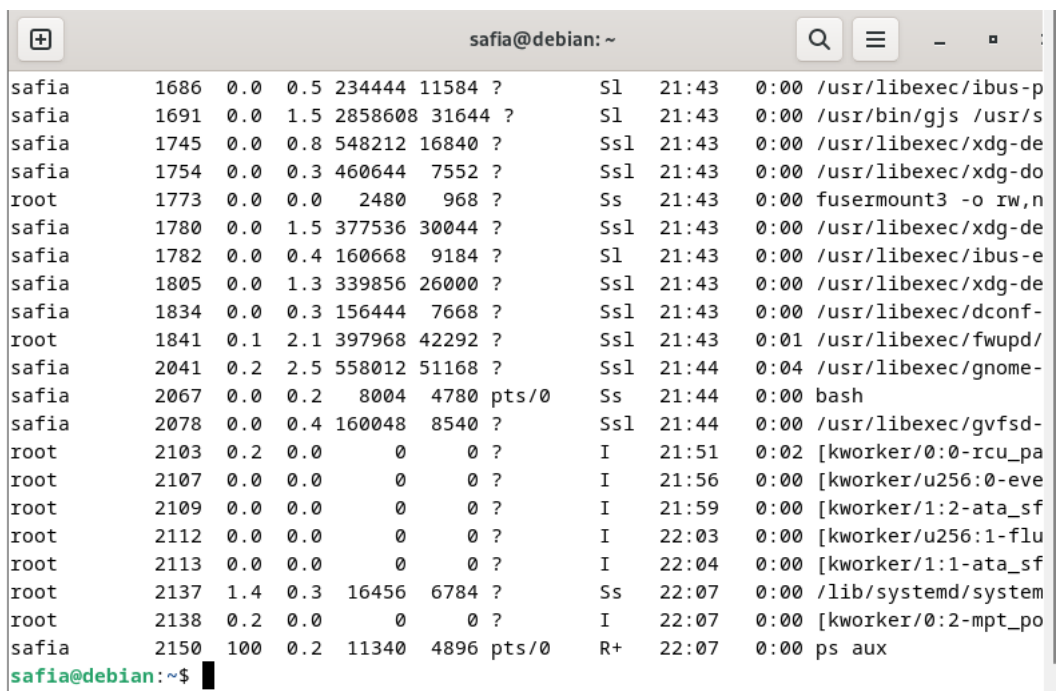
^G Aide      ^O Écrire    ^W Chercher  ^K Couper    ^T Exécuter  ^C Emplacement
^X Quitter    ^R Lire fich.^_ Remplacer  ^U Coller    ^J Justifier ^/ Aller ligne
```

Extraction des informations relatives aux villes en utilisant AWK

```
safia@debian:~$ nanofichier_csv.py
bash: nanofichier_csv.py : commande introuvable
safia@debian:~$ nano fichier_csv.py
safia@debian:~$ python3 fichier_csv.py
Fichier CSV créé avec succès: personnes.csv
safia@debian:~$ awk -F ',' '{print $3}' personnes.csv
Paris
Lyon
Marseille
Toulouse
safia@debian:~$ █
```

5. Gestion des processus

La commande **ps aux** permet de recenser tous les processus actifs sur le système.



```
safia@debian: ~
safia 1686 0.0 0.5 234444 11584 ? S1 21:43 0:00 /usr/libexec/ibus-p
safia 1691 0.0 1.5 2858608 31644 ? S1 21:43 0:00 /usr/bin/gjs /usr/s
safia 1745 0.0 0.8 548212 16840 ? Ssl 21:43 0:00 /usr/libexec/xdg-de
safia 1754 0.0 0.3 460644 7552 ? Ssl 21:43 0:00 /usr/libexec/xdg-do
root 1773 0.0 0.0 2480 968 ? Ss 21:43 0:00 fusemount3 -o rw,n
safia 1780 0.0 1.5 377536 30044 ? Ssl 21:43 0:00 /usr/libexec/xdg-de
safia 1782 0.0 0.4 160668 9184 ? S1 21:43 0:00 /usr/libexec/ibus-e
safia 1805 0.0 1.3 339856 26000 ? Ssl 21:43 0:00 /usr/libexec/xdg-de
safia 1834 0.0 0.3 156444 7668 ? Ssl 21:43 0:00 /usr/libexec/dconf-
root 1841 0.1 2.1 397968 42292 ? Ssl 21:43 0:01 /usr/libexec/fwupd/
safia 2041 0.2 2.5 558012 51168 ? Ssl 21:44 0:04 /usr/libexec/gnome-
safia 2067 0.0 0.2 8004 4780 pts/0 Ss 21:44 0:00 bash
safia 2078 0.0 0.4 160048 8540 ? Ssl 21:44 0:00 /usr/libexec/gvfsd-
root 2103 0.2 0.0 0 0 ? I 21:51 0:02 [kworker/0:0-rcu_pa
root 2107 0.0 0.0 0 0 ? I 21:56 0:00 [kworker/u256:0-eve
root 2109 0.0 0.0 0 0 ? I 21:59 0:00 [kworker/1:2-ata_sf
root 2112 0.0 0.0 0 0 ? I 22:03 0:00 [kworker/u256:1-flu
root 2113 0.0 0.0 0 0 ? I 22:04 0:00 [kworker/1:1-ata_sf
root 2137 1.4 0.3 16456 6784 ? Ss 22:07 0:00 /lib/systemd/system
root 2138 0.2 0.0 0 0 ? I 22:07 0:00 [kworker/0:2-mpt_po
safia 2150 100 0.2 11340 4896 pts/0 R+ 22:07 0:00 ps aux
safia@debian:~$ █
```

Pour fermer un processus spécifique, on utilise la commande **kill**.

Exemple :

```
safia@debian:~$ kill 1234
bash: kill: (1234) - Aucun processus de ce type
safia@debian:~$ █
```

Si on doit fermer de manière forcée un processus, on peut utiliser la commande « kill » avec le signal SIGKILL (9), exemple :

```
kill -9 1234
```

6. Surveillance des ressources système

On utilise la commande « **top** » ou « **htop** » si installée.

```
safia@debian: ~  
Tâches: 204 total, 1 en cours, 203 en veille, 0 arrêté, 0 zombie  
%Cpu(s): 2,0 ut, 0,6 sy, 0,0 ni, 97,2 id, 0,0 wa, 0,0 hi, 0,2 si, 0,0 st  
MiB Mem : 1932,4 total, 450,0 libr, 1107,0 util, 544,4 tamp/cache  
MiB Éch : 975,0 total, 975,0 libr, 0,0 util. 825,5 dispo Mem  


| PID  | UTIL. | PR | NI  | VIRT    | RES    | SHR    | S | %CPU | %MEM | TEMPS+  | COM.            |
|------|-------|----|-----|---------|--------|--------|---|------|------|---------|-----------------|
| 1353 | safia | 20 | 0   | 3734480 | 270472 | 128640 | S | 3,3  | 13,7 | 0:48.45 | gnome-shell     |
| 2041 | safia | 20 | 0   | 558308  | 51388  | 38588  | S | 0,7  | 2,6  | 0:06.06 | gnome-terminal- |
| 2197 | safia | 20 | 0   | 11824   | 5580   | 3412   | R | 0,7  | 0,3  | 0:00.06 | top             |
| 34   | root  | 20 | 0   | 0       | 0      | 0      | S | 0,3  | 0,0  | 0:00.32 | kcompactd0      |
| 2138 | root  | 20 | 0   | 0       | 0      | 0      | I | 0,3  | 0,0  | 0:03.52 | kworker/0:2-ev+ |
| 1    | root  | 20 | 0   | 167916  | 12500  | 9200   | S | 0,0  | 0,6  | 0:02.76 | systemd         |
| 2    | root  | 20 | 0   | 0       | 0      | 0      | S | 0,0  | 0,0  | 0:00.01 | kthreadd        |
| 3    | root  | 0  | -20 | 0       | 0      | 0      | I | 0,0  | 0,0  | 0:00.00 | rcu_gp          |
| 4    | root  | 0  | -20 | 0       | 0      | 0      | I | 0,0  | 0,0  | 0:00.00 | rcu_par_gp      |
| 5    | root  | 0  | -20 | 0       | 0      | 0      | I | 0,0  | 0,0  | 0:00.00 | slub_flushwq    |
| 6    | root  | 0  | -20 | 0       | 0      | 0      | I | 0,0  | 0,0  | 0:00.00 | netns           |
| 8    | root  | 0  | -20 | 0       | 0      | 0      | I | 0,0  | 0,0  | 0:00.00 | kworker/0:0H-k+ |
| 10   | root  | 0  | -20 | 0       | 0      | 0      | I | 0,0  | 0,0  | 0:00.00 | mm_percpu_wq    |
| 11   | root  | 20 | 0   | 0       | 0      | 0      | I | 0,0  | 0,0  | 0:00.00 | rcu_tasks_kthr+ |
| 12   | root  | 20 | 0   | 0       | 0      | 0      | I | 0,0  | 0,0  | 0:00.00 | rcu_tasks_rude+ |

  
safia@debian: ~$
```

Pour enregistrer ces informations dans un fichier CSV, on peut utiliser des outils de redirection de sortie « **awk** », ou « **csvkit** ».

```
GNU nano 7.2 surveillance_système.sh *  
#!/bin/bash  
while true; do  
    top -d1 -u all >> system_stats.csv  
    sleep 1  
done  
  
nom du fichier à écrire: surveillance_système.sh  
G Aide M-D Format DOS M-A Ajout (à la fin) M-B Copie de sécu.  
C Annuler M-M Format Mac M-P Ajout (au début) ^T Parcourir
```

Pour afficher le contenu du fichier « **system_ressources.csv** », on utilise la commande :
`cat system_ressources.csv`

```
safia@debian: ~  
1681,0,0,1,7,0:04.33,ibus-ex+  
1685,0,0,0,9,0:00.13,gsd-pri+  
1686,0,0,0,6,0:00.09,ibus-po+  
1691,0,0,1,6,0:00.38,gjs  
1745,0,0,0,9,0:00.33,xdg-des+  
1754,0,0,0,4,0:00.05,xdg-doc+  
1773,0,0,0,0,0:00.00,fuseimo+  
1780,0,0,1,5,0:00.38,xdg-des+  
1805,0,0,1,3,0:00.28,xdg-des+  
1834,0,0,0,4,0:00.01,dconf-s+  
1841,0,0,2,1,0:01.86,fwupd  
2041,0,0,2,6,0:08.49,gnome-t+  
2067,0,0,0,3,0:00.11,bash  
2078,0,0,0,4,0:00.02,gvfsd-m+  
2138,0,0,0,0,0:05.10,kworker+  
2172,0,0,0,0,0:00.11,kworker+  
2201,0,0,0,0,0:00.00,kworker+  
2203,0,0,0,0,0:00.04,kworker+  
2206,0,0,0,0,0:00.02,kworker+  
2207,0,0,0,0,0:00.04,kworker+  
2224,0,0,0,1,0:00.00,awk  
safia@debian:~$
```

7. Scripting avancé

Pour automatiser la sauvegarde périodique du répertoire « Plateforme » et gérer l'historique des sauvegardes on écrit un script bash :

```
safia@debian: ~/Plateforme  
GNU nano 7.2 sauvegarde_Plateforme.sh *  
# Définition du répertoire source  
SOURCE_DIR="/home/utilisateur/Plateforme"  
  
# Définition du répertoire de destination  
DEST_DIR="/home/utilisateur/Sauvegardes/Plateforme"  
  
# Date et heure actuelle  
DATE_TIME=$(date +"%Y-%m-%d_%H-%M-%S")  
  
# Nom du fichier de sauvegarde  
BACKUP_FILE="sauvegarde-${DATE_TIME}.tar.gz"  
  
# Compression et archivage du répertoire  
tar -zcvf "${DEST_DIR}/${BACKUP_FILE}" "${SOURCE_DIR}"  
  
# Rotation des sauvegardes/suppression (7 jours)  
find "${DEST_DIR}" -type f -mtime +7 -exec rm -f {} \;  
  
# Historique des sauvegardes enregistrement dans un fichier  
echo "${DATE_TIME} : ${BACKUP_FILE}" >> "${DEST_DIR}/historique.log"  
  
^G Aide      ^O Écrire    ^W Chercher  ^K Couper    ^T Exécuter  ^C Emplacement  
^X Quitter   ^R Lire fich.^N Remplacer  ^U Coller    ^J Justifier  ^_ Aller ligne
```

8. Automatisation des mises à jour logicielles

On peut automatiser la recherche et la mise à jour des logiciels existants sur le système avec un script shell qui utilise le gestionnaire de package apt :

```
GNU nano 7.2 mise_a_jour.sh *
# Obtenir la liste des paquets à mettre à jour
liste_paquets=$(apt list --upgradable | grep -v "^Inst" | awk '{print $1}')
# Si aucun paquet n'est à mettre à jour, quitter le script
if [ -z "$liste_paquets" ]; then
    echo "Aucun paquet à mettre à jour."
    exit 0
fi
# Afficher le nombre de paquets à mettre à jour
nb_paquets=$(echo "$liste_paquets" | wc -l)
echo "il y a $nb_paquets paquets à mettre à jour."
# Demander à l'utilisateur s'il souhaite installer les mises à jour
choix=$(zenity --question --text="Voulez-vous installer les mises à jour ?")
if [ "$choix" = "Oui" ]; then
    # Mettre à jour la liste des paquets disponibles
    apt update
    # Installer les mises à jour
    apt upgrade -y
    # Afficher un message de confirmation
    echo "Les mises à jour ont été installées avec succès."
else
    echo "les mises à jour n'ont pas été installées."
fi

^G Aide      ^O Écrire    ^W Chercher  ^K Couper    ^T Exécuter  ^C Emplacement M-U Annuler
^X Quitter   ^R Lire fich. ^V Remplacer ^U Coller    ^J Justifier ^_ Aller ligne M-E Refaire
```

9. Gestion des dépendances logicielles

Un script shell permet d'installer les dépendances logicielles pour un projet web :


```

GNU nano 7.2                                gestion_logicielles.sh *
# Installation du serveur web

echo "Installation du serveur web..."
sudo apt update
sudo apt install apache2      #Pour Apache
# sudo apt install nginx      #Pour Nginx
echo "Serveur web installé avec succès."

# Installation de PHP et phpMyAdmin
echo "Installation de PHP et phpMyAdmin..."
sudo apt install php php-mysql phpmyadmin
sudo phpenmod mysqli
sudo systemctl restart apache2
echo "PHP et phpMyAdmin installés avec succès."

# Installation de MySQL ou MariaDB
echo "Installation de la base de données..."
sudo apt install mariadb-server

```

```

GNU nano 7.2                                gestion_logicielles.sh
# Installation de MySQL ou MariaDB
echo "Installation de la base de données..."
sudo apt install mariadb-server
# sudo apt install mysql-server
sudo mysql_secure_installation
echo "Base de données installées avec succès."

#Installation de Node.js et npm
echo "Installation de Node.js et npm..."
sudo apt install nodejs npm
echo "Node.js et npm installés avec succès."

# Installation de Git
echo "Installation de Git..."
sudo apt install git
echo "Git installé avec succès."

echo "Toutes les dépendances ont été installées avec succès."

^G Aide      ^O Écrire    ^W Chercher  ^K Couper    ^T Exécuter  ^C Emplacement
^X Quitter   ^R Lire fich.^_ Remplacer  ^U Coller    ^J Justifier ^_ Aller ligne

```

```

safia@debian:~$ nano gestion_logicielles.sh
safia@debian:~$ nano gestion_logicielles.sh
safia@debian:~$ nano gestion_logicielles.sh
safia@debian:~$ chmod +x gestion_logicielles.sh
safia@debian:~$ ./gestion_logicielles.sh
Installation du serveur web...
[sudo] Mot de passe de safia : █

```

10. Sécuriser ses scripts

Il existe des scripts spécifiques pour sécuriser les scripts :

- 1. Script de validation d'entrée:** Ce script peut être utilisé pour valider toutes les entrées du script et empêcher l'injection de données malveillantes.
- 2. Script d'échappement:** Ce script peut être utilisé pour échapper les caractères spéciaux dans les entrées et les sorties du script pour empêcher l'exécution de code malveillant.
- 3. Script de cryptage:** Ce script peut être utilisé pour crypter les données sensibles stockées dans le script.
- 4. Script de contrôle d'accès:** Ce script peut être utilisé pour limiter l'accès au script aux utilisateurs qui en ont besoin.
- 5. Script d'analyse de sécurité:** Ce script peut être utilisé pour identifier les vulnérabilités de sécurité dans le script.

Quelques exemples de frameworks de sécurité :

- **OWASP Zed Attack Proxy**

(ZAP): <https://www.linguee.fr/francais-anglais/traduction/non+valide.html>

- **Nessus:** <https://www.linguee.fr/francais-anglais/traduction/non+valide.html>

- **QualysGuard:** <https://www.linguee.fr/francais-anglais/traduction/non+valide.html>

La sécurité des scripts est un processus continu, notamment la mise à jour régulière pour corriger les vulnérabilités connues et utiliser les outils d'analyse de sécurité pour identifier les nouvelles vulnérabilités.

11. Utilisation d'API Web dans un script

Comment exploiter une API Web dans un script Shell de manière sécurisée, avec gestion des erreurs et journalisation.

```
GNU nano 7.2                                api.sh
# Définir l'URL de l'API et la clé API
API_URL="https://api.openweathermap.org/data/2.5/weather"
API_KEY="YOUR_API_KEY"

# Fonction pour envoyer une requête à l'API
function get_weather() {
    city="$1"

# Envoyer une requête GET à l'API
    response=$(curl -s -XGET "$API_URL?q=$city&appid=$API_KEY")

# Vérifier le code de retour HTTP
    if [[ $? -ne 0 ]]; then
        echo "Erreur lors de requête à l'API" >&2
        return 1
    fi

# Valider les données JSON
    if ! jq -e ' .main.temp' <<< "$response" > /dev/null; then
        echo "Données JSON invalides reçues de l'API" >&2
        return 1
    fi

# Extraire la température
    temperature=$(jq -r ' .main.temp' <<< "$response")

# Journaliser la requête et la réponse
    logger -i "Requête: $API_URL?q=$city&appid=$API_KEY"
    logger -i "réponse: $response"

    echo "La température à $city est de $temperature °C"
}

# Demander la ville à l'utilisateur
echo "Entrez le nom d'une ville: "
read city

# Appeler la fonction pour obtenir la météo
get_weather "$city"

# Gérer les erreurs potentielles
if [[ $? -ne 0 ]]; then
    echo "Une erreur est survenue." >&2
    exit 1
fi


```

^G Aide	^O Écrire	^W Chercher	^K Couper	^T Exécuter	^C Emplacement
^X Quitter	^R Lire fich.	^V Remplacer	^U Coller	^J Justifier	^_ Aller ligne