



Systeme, Scripts et Sécurité



Samet ARI
B1-PREPA

Mai-2025

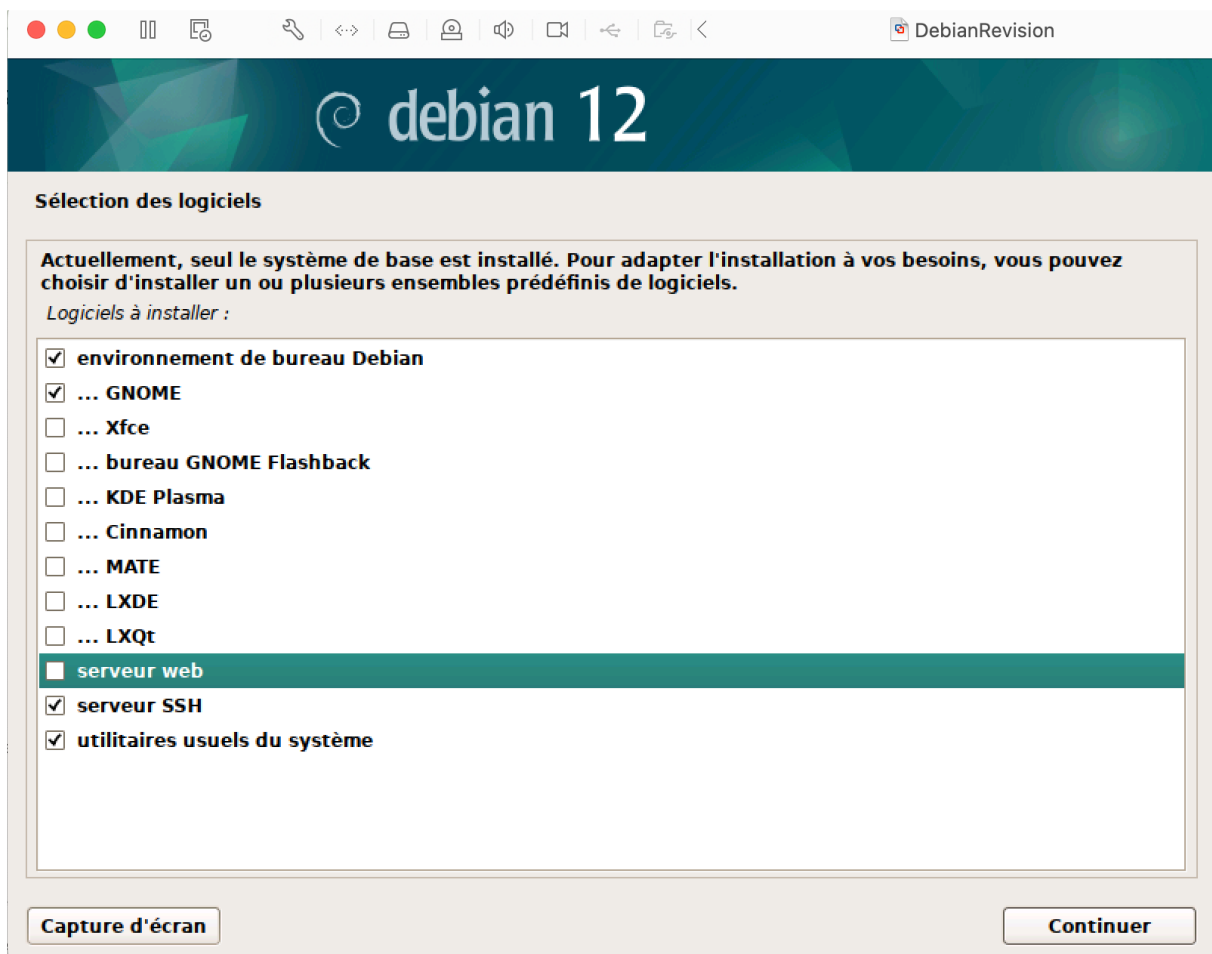
CONTENU

	Page
1. Création de la VM Debian.....	1
2. Commande de recherche avancées.....	3
3. Compression et décompression de fichiers.....	4
4. Manipulation de texte.....	6
5. Gestion de processus.....	7
6. Surveillance des ressources système.....	9
7. Scripting avancé.....	10
8. Automatisation des mises à jour logicielles.....	11
9. Gestion des dépendances logicielles.....	12
10. Sécuriser ses scripts.....	14
11. Utilisation d'API dans un script.....	15
12. CONCLUSION.....	17

CRÉATION DE LA VM DEBIAN

Pour commencer, je choisis un hyperviseur pour créer ma machine virtuelle Debian avec interface graphique. Dans ce projet, j'ai utilisé **VMWare** car je l'avais déjà utilisé auparavant et il répond aux exigences de mon projet.


Ensuite, je crée une nouvelle machine virtuelle dans mon hyperviseur. Je configure la machine virtuelle selon mes préférences, en allouant suffisamment de mémoire vive et d'espace disque.



Ensuite, je démarre ma machine virtuelle et j'installe le système d'exploitation **Debian 12** en utilisant l'image que j'ai téléchargée précédemment.

Lors de l'installation, je crée un utilisateur nommé « **La_Plateforme** » avec le mot de passe « **LAPlateforme_** » comme demandé dans la consigne.


Je configure également l'interface graphique pour qu'elle démarre automatiquement au démarrage de la machine virtuelle.



Créer les utilisateurs et choisir les mots de passe

Veillez choisir un identifiant (« login ») pour le nouveau compte. Votre prénom est un choix possible. Les identifiants doivent commencer par une lettre minuscule, suivie d'un nombre quelconque de chiffres et de lettres minuscules.

Identifiant pour le compte utilisateur :



Créer les utilisateurs et choisir les mots de passe

Un bon mot de passe est composé de lettres, chiffres et signes de ponctuation. Il devra en outre être changé régulièrement.

Mot de passe pour le nouvel utilisateur :

☒ **Afficher le mot de passe en clair**

Veillez entrer à nouveau le mot de passe pour l'utilisateur, afin de vérifier que votre saisie est correcte.

Confirmation du mot de passe :

☐ **Afficher le mot de passe en clair**

En ce qui concerne la connexion internet, mon hyperviseur VMWare la configure automatiquement.

Cependant, j'ai vérifié que la connexion fonctionnait correctement en utilisant la commande suivante dans le terminal de ma machine virtuelle.

```
laplateforme@laplateforme:~$ ping google.com
PING google.com (142.251.37.206) 56(84) bytes of data.
64 bytes from mrs09s15-in-f14.1e100.net (142.251.37.206): icmp_seq=1 ttl=128 time=5.95 ms
64 bytes from mrs09s15-in-f14.1e100.net (142.251.37.206): icmp_seq=2 ttl=128 time=4.71 ms
```

COMMANDES DE RECHERCHES AVANCÉES

Pour commencer, je vais utiliser les commandes du terminal pour créer cinq fichiers texte nommés **"mon_texte.txt"** contenant le texte **"Que la force soit avec toi."** et les répartir dans les répertoires **"Bureau"**, **"Documents"**, **"Téléchargements"**, **"Vidéos"** et **"Images"**.

Tout d'abord, j'utilise la commande **"echo"** pour créer les fichiers texte et ajouter le texte demandé. Pour ce faire, j'exécute les commandes suivantes :

```
laplateforme@laplateforme:~$ echo "Que la force soit avec toi" > Bureau/mon_texte.txt
laplateforme@laplateforme:~$ echo "Que la force soit avec toi" > Documents/mon_texte.txt
laplateforme@laplateforme:~$ echo "Que la force soit avec toi" > Téléchargements/mon_texte.txt
laplateforme@laplateforme:~$ echo "Que la force soit avec toi" > Vidéos/mon_texte.txt
laplateforme@laplateforme:~$ echo "Que la force soit avec toi" > Images/mon_texte.txt
```

Ces commandes créent les fichiers **"mon_texte.txt"** dans chaque répertoire et ajoutent le texte **"Que la force soit avec toi."** à l'intérieur.

Ensuite, j'utilise la commande **"grep"** pour localiser les fichiers contenant le mot **"force"** à partir du répertoire de ma session. Pour ce faire, j'exécute la commande suivante :

```
laplateforme@laplateforme:~$ grep -r "force"
Vidéos/mon_texte.txt:Que la force soit avec toi
grep: .cache/tracker3/files/http%3A%2F%2Ftracker.
grep: .cache/tracker3/files/meta.db-wal : fichier
.cache/gnome-software/odrs/ratings.json: "io.g
.cache/gnome-software/odrs/ratings.json: "io.g
.cache/gnome-software/odrs/ratings.json: "io.s
.cache/gnome-software/odrs/ratings.json: "io.s
.cache/gnome-software/odrs/ratings.json: "io.s
grep: .cache/gnome-software/appstream/components.
Téléchargements/mon_texte.txt:Que la force soit a
Documents/mon_texte.txt:Que la force soit avec to
.bashrc:#force_color_prompt=yes
.bashrc:if [ -n "$force_color_prompt" ]; then
.bashrc:unset color_prompt force_color_prompt
Images/mon_texte.txt:Que la force soit avec toi
Bureau/mon_texte.txt:Que la force soit avec toi
```

COMPRESSION ET DÉCOMPRESSION DE FICHIERS

Maintenant que j'ai créé cinq (5) fichiers texte nommés "**mon_texte.txt**" contenant le texte "**Que la force soit avec toi.**" et que je les ai répartis dans les répertoires "**Bureau**", "**Documents**", "**Téléchargement**", "**Vidéos**" et "**Images**".

Je vais créer un répertoire nommé "**Plateforme**" dans le dossier "**Documents**" de ma session et y ajouter le fichier "**mon_texte.txt**" que j'ai précédemment créé.

Pour ce faire, j'utilise la commande "**mkdir**" pour créer le répertoire "**Plateforme**" dans le dossier "**Documents**" et je copie le fichier "**mon_texte.txt**" dans ce nouveau répertoire "**Plateforme**" avec la commande "**cp**" :

```
laplateforme_@laplateforme:~/Documents$ mkdir Plateforme
laplateforme_@laplateforme:~/Documents$ cp mon_texte.txt Plateforme
```

Ensuite, j'ai copié le fichier "**mon_texte.txt**" dans le répertoire "**Plateforme**", je vais dupliquer ce fichier quatre fois dans ce même répertoire, formant ainsi un total de cinq fichiers dans le dossier "**Plateforme**".

Pour ce faire, j'utilise la commande "**cp**" avec l'option "**-n**" afin d'éviter d'écraser les fichiers déjà existants :

```
laplateforme_@laplateforme:~/Documents/Plateforme$ ls
mon_texte.txt
laplateforme_@laplateforme:~/Documents/Plateforme$ cp -n mon_texte.txt mon_texte1.txt
laplateforme_@laplateforme:~/Documents/Plateforme$ cp -n mon_texte.txt mon_texte2.txt
laplateforme_@laplateforme:~/Documents/Plateforme$ cp -n mon_texte.txt mon_texte3.txt
laplateforme_@laplateforme:~/Documents/Plateforme$ cp -n mon_texte.txt mon_texte4.txt
laplateforme_@laplateforme:~/Documents/Plateforme$ ls
mon_texte1.txt mon_texte2.txt mon_texte3.txt mon_texte4.txt mon_texte.txt
```

J'ai maintenant cinq (5) fichiers texte dans le répertoire "**Plateforme**".

Je vais maintenant archiver ce répertoire "**Plateforme**" en utilisant les commandes "**tar**" et "**gzip**".

Pour explorer différentes options de compression, je vais créer trois archives avec différents niveaux de compression.

```
laplateforme_@laplateforme:~/Documents/Plateforme$ tar -czvf ../Plateforme.tar.gz
./
./mon_texte3.txt
./mon_texte1.txt
./mon_texte2.txt
./mon_texte4.txt
./mon_texte.txt
```

```
laplateforme@laplateforme:~/Documents/Plateforme$ tar -cjvf ../Plateforme.tar.bz2 .  
./  
./mon_texte3.txt  
./mon_texte1.txt  
./mon_texte2.txt  
./mon_texte4.txt  
./mon_texte.txt  
laplateforme@laplateforme:~/Documents/Plateforme$ tar -cJvf ../Plateforme.tar.xz .  
./  
./mon_texte3.txt  
./mon_texte1.txt  
./mon_texte2.txt  
./mon_texte4.txt  
./mon_texte.txt
```

J'ai maintenant trois archives compressées du répertoire **"Plateforme"**.

Pour décompresser ces archives, je vais utiliser les commandes appropriées pour chaque type de compression :

```
laplateforme@laplateforme:~$ tar -xzf Documents/Plateforme.tar.gz -C Documents/  
./  
./mon_texte3.txt  
./mon_texte1.txt  
./mon_texte2.txt  
./mon_texte4.txt  
./mon_texte.txt  
laplateforme@laplateforme:~$ tar -xjvf Documents/Plateforme.tar.bz2 -C Documents/  
./  
./mon_texte3.txt  
./mon_texte1.txt  
./mon_texte2.txt  
./mon_texte4.txt  
./mon_texte.txt  
laplateforme@laplateforme:~$ tar -xJvf Documents/Plateforme.tar.xz -C Documents/  
./  
./mon_texte3.txt  
./mon_texte1.txt  
./mon_texte2.txt  
./mon_texte4.txt  
./mon_texte.txt
```

MANIPULATION DE TEXTE

Tout d'abord, j'ai créé un script(bash)nommé **personnes.csv** dans lequel j'écris le code suivant :

```
GNU nano 7.2      personnes.csv
Prénom,Âge,Ville
Jean,25 ans,Paris
Marie,30 ans,Lyon
Pierre,22 ans,Marseille
Sophie,35 ans,Toulouse
```

Ensuite, j'ai utilisé la commande **awk** avec l'option **-F** pour spécifier le séparateur de colonnes (dans ce cas, une virgule) et la commande **{print \$3}** pour extraire la troisième colonne du fichier CSV, qui contient les noms des villes. J'ai exécuté cette commande dans le répertoire où se trouve le fichier CSV et j'ai vérifié que les résultats correspondaient aux noms des villes dans le fichier. Enfin, j'ai utilisé la commande **cat** pour afficher le contenu du fichier CSV et vérifier que les données avaient été correctement écrites.

```
la_plateforme@debian:~$ cat personnes.csv
Prénom,Âge,Ville
Jean,25 ans,Paris
Marie,30 ans,Lyon
Pierre,22 ans,Marseille
Sophie,35 ans,Toulouse
la_plateforme@debian:~$ awk -F',' '{print $3}' personnes.csv
Ville
Paris
Lyon
Marseille
Toulouse
la_plateforme@debian:~$ █
```


GESTION DE PROCESSUS

```
la_plateforme@debian:~$ ps aux
```

JSER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
root	1	0.0	0.5	168184	11052	?	Ss	mai24	0:10	/lib/systemd/
root	2	0.0	0.0	0	0	?	S	mai24	0:00	[kthreadd]
root	3	0.0	0.0	0	0	?	I<	mai24	0:00	[rcu_gp]
root	4	0.0	0.0	0	0	?	I<	mai24	0:00	[rcu_par_gp]
root	5	0.0	0.0	0	0	?	I<	mai24	0:00	[slub_flushwq]
root	6	0.0	0.0	0	0	?	I<	mai24	0:00	[netns]
root	8	0.0	0.0	0	0	?	I<	mai24	0:00	[kworker/0:0H]
root	10	0.0	0.0	0	0	?	I<	mai24	0:00	[mm_percpu_wq]
root	11	0.0	0.0	0	0	?	I	mai24	0:00	[rcu_tasks_kt]
root	12	0.0	0.0	0	0	?	I	mai24	0:00	[rcu_tasks_ru]
root	13	0.0	0.0	0	0	?	I	mai24	0:00	[rcu_tasks_tr]
root	14	0.0	0.0	0	0	?	S	mai24	0:01	[ksoftirqd/0]
root	15	0.0	0.0	0	0	?	I	mai24	0:05	[rcu_preempt]
root	16	0.0	0.0	0	0	?	S	mai24	0:01	[migration/0]
root	18	0.0	0.0	0	0	?	S	mai24	0:00	[cpuhp/0]
root	19	0.0	0.0	0	0	?	S	mai24	0:00	[cpuhp/1]
root	20	0.0	0.0	0	0	?	S	mai24	0:01	[migration/1]
root	21	0.0	0.0	0	0	?	S	mai24	0:01	[ksoftirqd/1]
root	23	0.0	0.0	0	0	?	I<	mai24	0:00	[kworker/1:0H]
root	26	0.0	0.0	0	0	?	S	mai24	0:00	[kdevtmpfs]
root	27	0.0	0.0	0	0	?	I<	mai24	0:00	[inet_frag_wq]
root	28	0.0	0.0	0	0	?	S	mai24	0:00	[kauditd]

Pour afficher tous les processus en cours d'exécution sur le système, y compris ceux qui n'ont pas de terminal de contrôle, vous pouvez utiliser la commande

ps aux

Cela affichera une liste de tous les processus, avec leur ID de processus (PID), le nom de l'utilisateur qui a lancé le processus, le pourcentage d'utilisation du processeur et de la mémoire, ainsi que la commande qui a lancé le processus.

Pour terminer un processus spécifique, vous pouvez utiliser la commande

kill

suivie de son ID de processus. Par exemple, pour terminer le processus avec l'**ID 1800**, vous pouvez utiliser la commande dans la capture d'écran.

```
la_plat+ 65112 0.0 0.2 11348 4896 pts/0 R+ 17:43 0:00 ps aux
la_plateforme@debian:~$ kill 1800
la_plateforme@debian:~$ █
```

```

la_plat+ 1761 0.0 0.8 598932 16796 ? Ssl mai24 0:00 /usr/libexec/xdg-desktop-portal-gnome
la_plat+ 1781 0.0 0.3 160812 6800 ? Sl mai24 0:00 /usr/libexec/ibus-engine-simple
la_plat+ 1800 0.0 2.3 779080 46760 ? Sssl mai24 0:05 /usr/libexec/tracker-miner-fs-3
la_plat+ 1815 0.0 0.7 340052 14664 ? Ssl mai24 0:00 /usr/libexec/xdg-desktop-portal-gtk
la_plat+ 1872 0.0 0.3 160172 6012 ? Ssl mai24 0:00 /usr/libexec/gvfsd-metadata
la_plat+ 1930 0.0 1.4 559164 28056 ? Ssl mai24 0:06 /usr/libexec/gnome-terminal-server
la_plat+ 1956 0.0 0.2 9576 5764 pts/0 Ss mai24 0:00 bash

```

Si le processus ne se termine pas après quelques secondes, **je peux utiliser** la commande **kill -9** pour forcer sa terminaison.

Cette commande **envoie** un signal **SIGKILL** au processus, ce qui **le force à se terminer immédiatement**.

Je sais que cette méthode doit être utilisée avec précaution, car elle peut entraîner une perte de données ou des problèmes de stabilité du système.

Vous pouvez également utiliser la commande **pskill** pour terminer un processus en fonction de son nom.

Cette commande envoie un signal **SIGTERM** à tous les processus nommés. Vous pouvez également utiliser la commande **pskill -9** pour forcer la terminaison de ces processus.

```

la_plat+ 65634 0.0 3.7 2405904 75004 ? Sl 18:07 0:00 /usr/lib/firefox-esr/firefox-esr -conten
la_plat+ 65687 0.1 3.6 2407064 72668 ? Sl 18:07 0:00 /usr/lib/firefox-esr/firefox-esr -conten
la_plat+ 65715 0.1 3.6 2407064 72684 ? Sl 18:07 0:00 /usr/lib/firefox-esr/firefox-esr -conten
la_plat+ 65736 0.1 2.2 381984 45468 ? Sl 18:07 0:00 /usr/lib/firefox-esr/firefox-esr -conten
root 65786 0.0 0.0 0 0 ? I 18:08 0:00 [kworker/0:3-events]
la_plat+ 65831 50.0 0.2 11348 4884 pts/0 R+ 18:09 0:00 ps aux
la_plateforme@debian:~$ kill -9 65634
la_plateforme@debian:~$

```

SURVEILLANCE DES RESSOURCES SYSTÈME

vmstat est une commande que j'utilise pour rapporter des statistiques sur la mémoire virtuelle, la mémoire physique, le CPU, les processus, les entrées/sorties et les interruptions système. J'ajoute 1 pour spécifier l'intervalle de temps, en secondes, auquel les statistiques doivent être collectées et affichées. Ensuite, je redirige ces données vers un fichier nommé **ressources.csv**.

```
la_plateforme@debian:~/Documents$ top
```

```
top - 18:19:33 up 1 day, 7:33, 1 user, load average: 0,16, 0,25, 0,18
Tâches: 267 total, 1 en cours, 266 en veille, 0 arrêté, 0 zombie
%Cpu(s): 0,7 ut, 0,3 sy, 0,0 ni, 98,8 id, 0,0 wa, 0,0 hi, 0,2 si, 0,0 st
MiB Mem : 1932,4 total, 169,7 libr, 1431,3 util, 533,5 tamp/cache
MiB Éch : 975,0 total, 520,7 libr, 454,3 util. 501,1 dispo Mem
```

PID	UTIL.	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TEMPS+	COM.
65603	la_plat+	20	0	2598804	202132	98060	S	2,3	10,2	0:20.78	Isolated Web Co
1350	la_plat+	20	0	3861872	207000	82220	S	0,3	10,5	6:17.65	gnome-shell
65958	la_plat+	20	0	11844	5596	3400	R	0,3	0,3	0:00.04	top
1	root	20	0	168184	9508	6480	S	0,0	0,5	0:10.53	systemd
2	root	20	0	0	0	0	S	0,0	0,0	0:00.24	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	0:00.01	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-events_highpri
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_kthread
12	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tasks_rude_kthread
13	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_tasks_trace_kthread
14	root	20	0	0	0	0	S	0,0	0,0	0:02.04	ksoftirqd/0
15	root	20	0	0	0	0	I	0,0	0,0	0:06.17	rcu_preempt
16	root	rt	0	0	0	0	S	0,0	0,0	0:01.20	migration/0
18	root	20	0	0	0	0	S	0,0	0,0	0:00.00	cpuhp/0
19	root	20	0	0	0	0	S	0,0	0,0	0:00.00	cpuhp/1

```
la_plateforme@debian:~/Documents$ vmstat 1
procs -----mémoire----- -échange- ----io----- -système- -----cpu-----
r b swpd libre tampon cache si so bi bo in cs us sy id wa st
4 0 465220 170776 13432 532928 0 2 12 29 21 38 0 0 100 0 0
2 0 465220 170776 13432 532928 0 0 0 0 131 508 1 2 97 0 0
2 0 465220 170776 13432 532928 0 0 0 0 109 510 3 1 97 0 0
4 0 465220 170776 13432 532928 0 0 0 0 103 482 3 1 97 0 0
1 0 465220 170776 13432 532928 0 0 0 0 97 464 1 1 98 0 0
1 0 465220 170776 13432 532928 0 0 0 36 136 528 3 2 96 0 0
2 0 465220 170776 13432 532928 0 0 0 0 111 467 1 1 98 0 0
1 0 465220 170776 13432 532928 0 0 0 0 133 515 2 2 96 0 0
3 0 465220 170776 13432 532928 0 0 0 0 160 580 2 1 97 0 0
2 0 465220 170776 13432 532928 0 0 316 0 190 615 3 2 95 1 0
^C
la_plateforme@debian:~/Documents$ vmstat 1 > ressources.csv
^C
la_plateforme@debian:~/Documents$ cat ressources.csv
procs -----mémoire----- -échange- ----io----- -système- -----cpu-----
r b swpd libre tampon cache si so bi bo in cs us sy id wa st
2 0 465220 170776 13444 533240 0 2 12 29 21 38 0 0 100 0 0
2 0 465220 170776 13444 533240 0 0 0 0 97 416 2 1 97 0 0
la_plateforme@debian:~/Documents$
```

SCRIPTING AVANCÉ

J'ai commencé par créer un script Shell nommé **sauvegarde_plateforme.sh** dans mon répertoire personnel. Ce script a pour but de sauvegarder automatiquement le dossier Plateforme dans un dossier backups.

Voici le contenu du script :

```
GNU nano 7.2                               sauvegarde_plateforme.sh
#!/bin/bash

# Définir les variables
SOURCE="$HOME/Plateforme"
DEST="$HOME/backups"
DATE=$(date +%Y-%m-%d-%H%M)
ARCHIVE_NAME="Plateforme_backup_$DATE.tar.gz"

# Message d'information
echo "Sauvegarde en cours..."

# Création de l'archive compressée
tar -czf "$DEST/$ARCHIVE_NAME" "$SOURCE"

# Vérification du succès de la commande précédente
if [ $? -eq 0 ]; then
    echo "Sauvegarde réussie : $ARCHIVE_NAME"
else
    echo "Erreur lors de la sauvegarde."
fi

# (Optionnel) Supprimer les anciennes sauvegardes pour ne garder que les 7 plus récentes
cd "$DEST"
ls -t Plateforme_backup *.tar.gz | tail -n +8 | xargs -r rm --
```

J'ai exécuté le script et Le terminal m'a affiché les messages suivants :

```
la_plateforme@debian:~$ ~/sauvegarde_plateforme.sh
Sauvegarde en cours...
tar: Suppression de « / » au début des noms des membres
Sauvegarde réussie : Plateforme_backup_2025-05-25-18h39.tar.gz
la_plateforme@debian:~$ █
```

J'ai vérifié que le fichier de sauvegarde a bien été généré en listant le contenu du dossier backups :

```
la_plateforme@debian:~$ ls ~/backups
backup.log                               Plateforme_backup_2025-05-25-18h32.tar.gz
Plateforme_2025-05-25_12-49-30.tar.gz    Plateforme_backup_2025-05-25-18h39.tar.gz
la_plateforme@debian:~$
```

Avec cette ligne dans le crontab, **mon script de sauvegarde est exécuté automatiquement tous les jours à 08h00 du matin**. Cela me permet de conserver une copie régulière de mon dossier de travail sans intervention manuelle.

```
GNU nano 7.2                               /tmp/crontab.NmYUIj/crontab
2 * * * /bin/bash /home/la_plateforme/backup_plateforme.sh
8 * * * /home/<la_plateforme>/sauvegarde_plateforme.sh
# Edit this file to introduce tasks to be run by cron.
#
```

AUTOMATISATION DES MISES À JOUR LOGICIELS

```
GNU nano 7.2                                update_script.sh *
#!/bin/bash

echo "Mise à jour des listes de paquets en cours..."
sudo apt update

echo -e "\nLes mises à jour disponibles sont :"
apt list --upgradable

read -p $'\nVoulez-vous installer les mises à jour ? (o/n) : ' choix

if [[ "$choix" == "o" || "$choix" == "O" ]]; then
    echo -e "\nInstallation des mises à jour..."
    sudo apt upgrade -y
    echo -e "\nMise à jour terminée."
else
    echo -e "\nMise à jour annulée."
fi
```

Ce script Bash est conçu pour gérer les mises à jour des paquets sur un système basé sur Debian ou Ubuntu. Il commence par exécuter la commande **sudo apt update** pour actualiser la liste des paquets disponibles, puis affiche les paquets pouvant être mis à jour à l'aide de **apt list --upgradable**.

Ensuite, il demande à l'utilisateur s'il souhaite installer les mises à jour. Si l'utilisateur répond "o" (oui), le script lance **sudo apt upgrade -y** pour installer automatiquement toutes les mises à jour disponibles, puis affiche un message confirmant que la mise à jour est terminée. Si l'utilisateur refuse, le script affiche que la mise à jour a été annulée. C'est un outil simple et interactif permettant de garder son système à jour en toute sécurité.

GESTION DES DÉPENDANCES LOGICIELS

```
GNU nano 7.2                                install_dependencies.sh *
#!/bin/bash

echo "Mise à jour des listes de paquets..."
sudo apt update

echo "Choisissez le serveur Web à installer : (1) Apache (2) Nginx"
read serveur

if [ "$serveur" == "1" ]; then
    echo "Installation d'Apache..."
    sudo apt install -y apache2
elif [ "$serveur" == "2" ]; then
    echo "Installation de Nginx..."
    sudo apt install -y nginx
else
    echo "Choix invalide."
    exit 1
fi

echo "Choisissez le système de gestion de base de données : (1) MySQL (2) MariaDB"
read db

if [ "$db" == "1" ]; then
    echo "Installation de MySQL..."
    sudo apt install -y mysql-server
elif [ "$db" == "2" ]; then
    echo "Installation de MariaDB..."
    sudo apt install -y mariadb-server
else
    echo "Choix invalide."
fi
```

Ce script Bash permet d'automatiser l'installation des dépendances d'un serveur web et d'un système de gestion de base de données sur un système **Debian** ou **Ubuntu**. Il commence par mettre à jour la liste des paquets avec **sudo apt update**.

Ensuite, il demande à l'utilisateur de choisir un serveur web à installer, entre Apache (option 1) et Nginx (option 2). Selon le choix, il installe soit apache2, soit **nginx** via la commande **sudo apt install -y**. Si le choix est invalide, le script affiche un message d'erreur et s'arrête. Ensuite, il propose à l'utilisateur de choisir un système de gestion de base de données : **MySQL** (option 1) ou **MariaDB** (option 2). Il installe alors le paquet correspondant (mysql-server ou mariadb-server) selon le choix effectué.

En cas de saisie incorrecte, un message d'erreur est affiché. Ce script est utile pour configurer rapidement un environnement LAMP ou LEMP en laissant à l'utilisateur le choix des technologies à utiliser.

Le script installe également **Node.js**, **npm** et **Git**.

```

root@debian:~# ./update_script.sh
Mise à jour des listes de paquets en cours...
Atteint :1 http://deb.debian.org/debian bookworm InRelease
Atteint :2 http://security.debian.org/debian-security bookworm-security InRelease
Réception de :3 http://deb.debian.org/debian bookworm-updates InRelease [55,4 kB]
55,4 ko réceptionnés en 1s (100 ko/s)
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Tous les paquets sont à jour.
N: Le dépôt « Debian bookworm » a modifié sa valeur « non-free component » de « non-free » à « non-free non-free-firmware »
N: Plus d'information disponible dans la note de mise à jour ici : https://www.debian.org/releases/bookworm/amd64/release-notes/ch-info.html#non-free-split

Les mises à jour disponibles sont :
En train de lister... Fait

Voulez-vous installer les mises à jour ? (o/n) : o

Installation des mises à jour...
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Calcul de la mise à jour... Fait
Le paquet suivant a été installé automatiquement et n'est plus nécessaire :
  libdbus-glib-1-2
Veuillez utiliser « sudo apt autoremove » pour le supprimer.
0 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.

Mise à jour terminée.
root@debian:~# ls

```

À chaque étape, le script gère les réponses de l'utilisateur et exécute les commandes d'installation appropriées. Une fois toutes les installations terminées, un message indique que l'installation des paquets est terminée.

SÉCURISER SES SCRIPTS

Pour garantir la sécurité des scripts, il est essentiel de respecter certaines pratiques recommandées. L'une des premières consiste à appliquer le principe du moindre privilège : les **droits d'accès aux scripts** doivent être réduits au strict minimum nécessaire à leur exécution. Donner simplement les droits d'exécution avec **chmod +x** n'assure pas une sécurité suffisante. En effet, bien que cela rende le fichier exécutable, un utilisateur sans privilèges d'administration pourrait modifier le contenu du script et y insérer des commandes nécessitant des droits élevés, ce qui peut entraîner des contournements de sécurité.

Il est donc préférable de ne jamais inclure directement de commandes **sudo** dans un script. Si des droits administratifs sont indispensables (par exemple pour installer un paquet), il est recommandé de lancer le script lui-même avec **sudo**, via la commande **sudo ./script.sh**. Ainsi, le contrôle reste entre les mains de l'utilisateur qui initie l'exécution.

Enfin, la mise en place d'un système de **journalisation (log)** permet de suivre l'exécution des scripts, de détecter les comportements suspects et d'intervenir rapidement en cas d'anomalie. Cela renforce considérablement la traçabilité et la sécurité globale du système.

UTILISATION D'API WEB DANS UN SCRIPT

```
GNU nano 7.2                                weather_api.sh
#!/bin/bash

# Ce script récupère la météo actuelle pour la ville de Marseille en utilisant l'API OpenWeatherMap.

# ----- Configuration -----
API_KEY="febad9981aee45ec3bdd016294e39f35"
CITY="Marseille"
UNITS="metric"
LANG="fr"
API_URL="https://api.openweathermap.org/data/2.5/weather?q=$CITY&appid=$API_KEY&units=$UNITS&lang=$LANG"
LOG_FILE="/var/log/weather_api.json"
TEMP_THRESHOLD=30

# ----- Création du fichier log si besoin -----
if [ ! -f "$LOG_FILE" ]; then
    touch "$LOG_FILE"
    chmod 644 "$LOG_FILE"
fi

# ----- Récupération des données -----
response=$(curl -s -w "%{http_code}" -o /tmp/weather_response.json "$API_URL")
http_code="${response: -3}"

if [ "$http_code" != "200" ]; then
    echo "$(date --iso-8601=seconds) - Erreur lors de la récupération des données météo (code HTTP : $http_code)" >&2
    exit 1
fi

# ----- Lecture et extraction des données -----
data=$(cat /tmp/weather_response.json)
```

Ce script shell permet de récupérer et d'afficher les données météorologiques d'une ville donnée, à partir d'une API. Il suit plusieurs étapes bien définies pour assurer à la fois la récupération, l'analyse, l'affichage et la journalisation des informations :

1. Récupération des données météo :

Le script utilise curl pour envoyer une requête **HTTP** vers une API météo (dont l'URL est contenue dans la variable \$API_URL). La réponse est enregistrée dans un fichier temporaire (/tmp/weather_response.json) et le code de retour HTTP est stocké dans la variable http_code.

2. Vérification du code HTTP :

Si le code de réponse n'est pas 200, cela signifie qu'il y a eu une erreur lors de la récupération des données. Le script affiche alors un message d'erreur avec l'heure et termine l'exécution (exit 1).

3. Lecture et extraction des données :

Le script lit le fichier JSON téléchargé et extrait :

- la température (.main.temp),
- les conditions météo (.weather[0].description),
- l'humidité (.main.humidity), en utilisant l'outil jq.

4. **Affichage dans le terminal :**

Les données extraites sont affichées de manière lisible, avec des emojis pour une meilleure visibilité :

- température actuelle,
- conditions météo,
- humidité.

5. **Avertissement en cas de température élevée :**

Si la température dépasse un seuil défini par la variable \$TEMP_THRESHOLD, le script génère un message d'alerte dans le terminal, indiquant que la température est anormalement élevée dans la ville concernée.

6. **Journalisation au format JSON :**

Enfin, les informations météo sont journalisées dans un fichier sous forme d'une ligne JSON contenant :

- la date,
- le nom de la ville,
- la température,
- les conditions météo,
- le taux d'humidité.

Cela permet de garder une trace structurée des données relevées.

```

root@debian:~# jq . /var/log/weather_api.json
{
  "date": "2025-05-25T14:59:52+02:00",
  "city": "Marseille",
  "temperature": 22.27,
  "conditions": "couvert",
  "humidity": 30
}
{
  "date": "2025-05-25T15:00:01+02:00",
  "city": "Marseille",
  "temperature": 22.27,
  "conditions": "couvert",
  "humidity": 30
}
{
  "date": "2025-05-25T15:05:37+02:00",
  "city": "Marseille",
  "temperature": 22.27,
  "conditions": "couvert",
  "humidity": 29
}
{
  "date": "2025-05-25T17:00:02+02:00",
  "city": "Marseille",
  "temperature": 20.29,
  "conditions": "couvert",
  "humidity": 48
}
{
  "date": "2025-05-25T18:00:02+02:00",

```

CONCLUSION

Ce projet "Système, Scripts et Sécurité" m'a permis d'explorer en profondeur les aspects fondamentaux de l'administration système sous Linux, tout en développant des compétences essentielles en scripting et en sécurisation. À travers les différents exercices, j'ai maîtrisé les commandes de base, appris à manipuler les fichiers et processus, automatisé des tâches répétitives grâce aux scripts **Shell** et **Python**, et intégré des bonnes pratiques de sécurité dans mes développements.

Cette approche pratique m'a également sensibilisé à l'importance de la surveillance système, de la gestion des **API web** sécurisées et du logging en cybersécurité. Les compétences que j'ai acquises constituent une base solide pour l'administration et la sécurisation des infrastructures système, me préparant efficacement aux défis professionnels du domaine de la cybersécurité.