

Procédure de création de la VM (Configuration)

Cette section détaille les étapes suivies pour mettre en place l'environnement de base du CTF avant l'injection des vulnérabilités. L'objectif est d'obtenir une installation "propre" et fonctionnelle.

Caractéristiques de la Machine Virtuelle

- **Système d'exploitation** : Debian 13 (Trixie) 64-bit.
- **Ressources** : 1 vCPU, 1024 Mo de RAM, 10 Go de disque (allocation dynamique).
- **Interface** : Console uniquement (pas d'interface graphique).

Étapes de Configuration

Étape 1 : Installation du Système

1. **Montage de l'ISO** : Utilisation de l'image debian-13.3.0-amd64-DVD-1.iso.
2. **Partitionnement** : Choix du partitionnement assisté ("Tout dans une seule partition") pour simplifier la gestion des fichiers du challenge.
3. **Logiciels** :
 - **Désactiver** : Environnement de bureau Debian, GNOME.
 - **Activer** : Serveur SSH, Utilitaires usuels du système.

Étape 2 : Configuration Réseau et Accès

Pour que le CTF soit accessible, la machine doit être joignable par l'attaquant.

1. **Mode Réseau** : Configuration de la carte réseau en mode **Accès par pont (Bridge)** ou **Réseau privé hôte (Host-only)** dans les réglages de la VM.
2. **Activation du SSH** :
 - Vérification du statut : `systemctl status ssh`
 - Recherche de l'adresse IP : `ip a`

Étape 3 : Installation des Outils de Base

Installation des paquets nécessaires pour l'administration et la mise en place des futurs niveaux :

Bash

Mise à jour des dépôts

`apt update && apt upgrade -y`

Installation des outils essentiels

apt install -y sudo nano vim net-tools openssh-server coreutils

```
bouboulito@debian: ~  
su: Échec de l'authentification  
bouboulito@debian:~$ su -  
Mot de passe :  
root@debian:~# apt update && apt install apache2 -y  
Atteint : 1 http://security.debian.org/debian-security trixie-security InRelease  
Atteint : 2 http://deb.debian.org/debian trixie InRelease  
Atteint : 3 http://deb.debian.org/debian trixie-updates InRelease  
25 paquets peuvent être mis à jour. Exécutez « apt list --upgradable » pour les voir.  
Installation de :  
  apache2  
  
Installation de dépendances :  
  apache2-data apache2-utils  
  
Paquets suggérés :  
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom ufw  
  
Sommaire :  
  Mise à niveau de : 0. Installation de : 3 Supprimé : 0. Non mis à jour : 25  
Taille du téléchargement : 601 kB  
Espace nécessaire : 1 916 kB / 90,0 GB disponible
```

```
bouboulito@debian: ~  
Réception de : 1 http://deb.debian.org/debian trixie/main amd64 apache2-data all 2.4.66-1~deb13u2 [160 kB]  
Réception de : 2 http://deb.debian.org/debian trixie/main amd64 apache2-utils amd64 2.4.66-1~deb13u2 [216 kB]  
Réception de : 3 http://deb.debian.org/debian trixie/main amd64 apache2 amd64 2.4.66-1~deb13u2 [225 kB]  
601 ko réceptionnés en 1s (1 106 ko/s)  
Sélection du paquet apache2-data précédemment désélectionné.  
(Lecture de la base de données... 162896 fichiers et répertoires déjà installés.)  
Préparation du dépaquetage de .../apache2-data_2.4.66-1~deb13u2_all.deb ...  
Dépaquetage de apache2-data (2.4.66-1~deb13u2) ...  
Sélection du paquet apache2-utils précédemment désélectionné.  
Préparation du dépaquetage de .../apache2-utils_2.4.66-1~deb13u2_amd64.deb ...  
Dépaquetage de apache2-utils (2.4.66-1~deb13u2) ...  
Sélection du paquet apache2 précédemment désélectionné.  
Préparation du dépaquetage de .../apache2_2.4.66-1~deb13u2_amd64.deb ...  
Dépaquetage de apache2 (2.4.66-1~deb13u2) ...  
Paramétrage de apache2-data (2.4.66-1~deb13u2) ...  
Paramétrage de apache2-utils (2.4.66-1~deb13u2) ...  
Paramétrage de apache2 (2.4.66-1~deb13u2) ...  
Enabling module mpm_event.  
Enabling module authz_core.  
Enabling module authz_host.
```

```
bouboulito@debian: ~  
Enabling module authn_core.  
Enabling module auth_basic.  
Enabling module access_compat.  
Enabling module authn_file.  
Enabling module authz_user.  
Enabling module alias.  
Enabling module dir.  
Enabling module autoindex.  
Enabling module env.  
Enabling module mime.  
Enabling module negotiation.  
Enabling module setenvif.  
Enabling module filter.  
Enabling module deflate.  
Enabling module status.  
Enabling module reqtimeout.  
Enabling conf charset.  
Enabling conf localized-error-pages.  
Enabling conf other-vhosts-access-log.  
Enabling conf security.  
Enabling conf serve-cgi-bin.  
Enabling site 000-default.  
Created symlink '/etc/systemd/system/multi-user.target.wants/apache2.service' ->  
'/usr/lib/systemd/system/apache2.service'.
```

Étape 4 : Nettoyage et Snapshot (Crucial)

Une fois la base installée, le système est nettoyé pour éviter de donner des indices sur l'installation au futur participant.

1. **Nettoyage des fichiers temporaires** : apt clean
2. **Snapshot** : Création d'un "Instantané" dans le logiciel de virtualisation (VirtualBox/VMware).
 - *Pourquoi ?* Cela permet de réinitialiser le CTF instantanément si un utilisateur casse le système ou si une erreur est commise lors de la configuration des permissions.

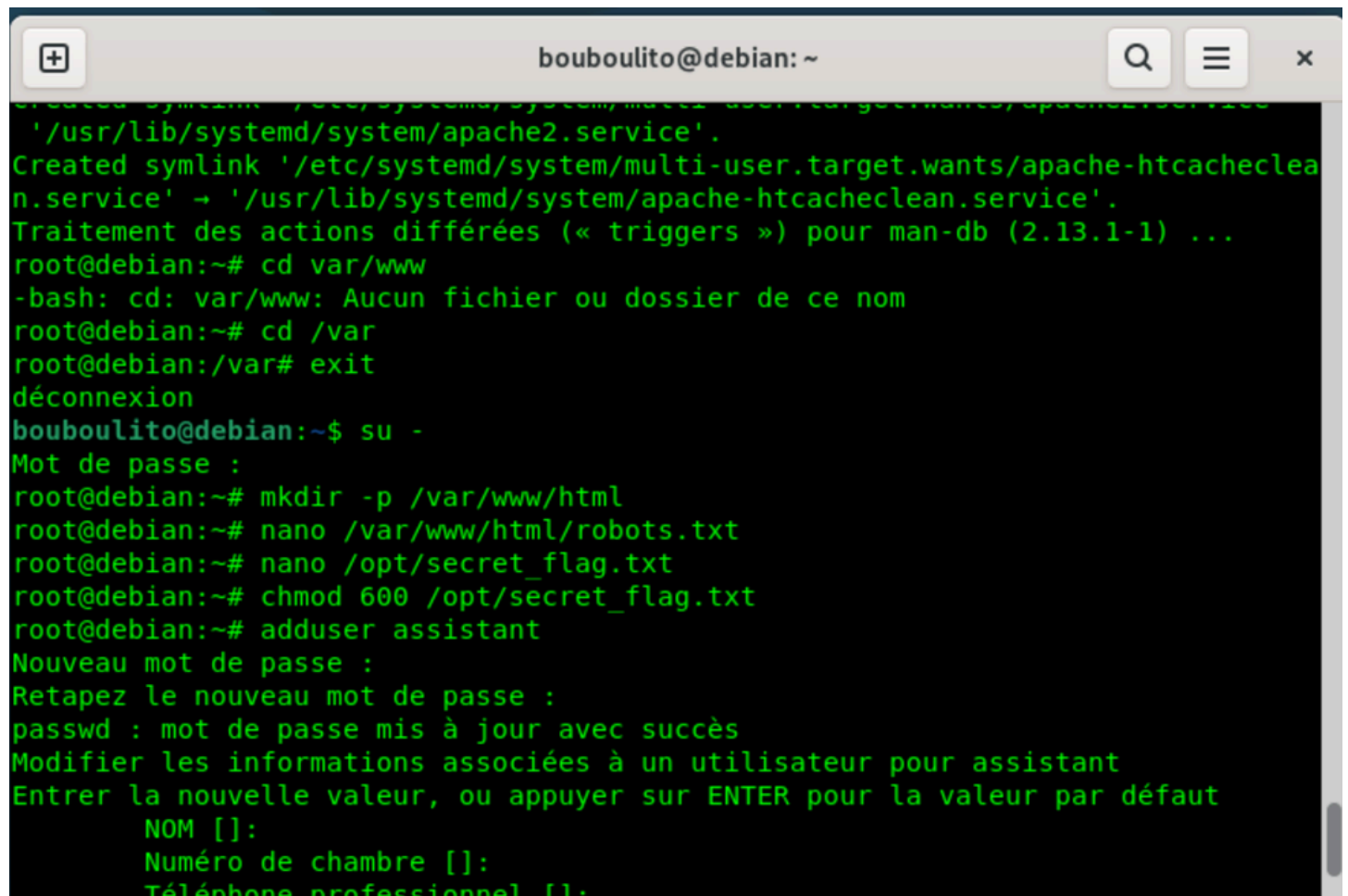
Niveau 1 Reconnaissance et Fuite de données (Information Disclosure)

C'est la porte d'entrée de toute intrusion système.

- **Le concept** : L'attaquant cherche des pistes sur le serveur web (le fichier robots.txt) pour identifier des fichiers sensibles. Il exploite ensuite des erreurs de configuration (mots de passe oubliés dans des scripts de maintenance) pour obtenir un premier accès utilisateur.
- **L'action** : Créer un fichier "indice", configurer un utilisateur "victime" avec un mot de passe caché, et mettre en place une énigme pour l'escalade vers root via un fichier caché.
- **Le défi** : L'attaquant doit d'abord explorer le service Web, puis fouiller le système pour trouver le mot de passe de l'utilisateur assistant avant de résoudre une énigme logique pour devenir root et lire le flag final.

Étape 1 : Préparer les pistes et le Flag (Serveur Web) Pour que l'attaquant trouve une piste sur le serveur web, nous configurons un indice public et le fichier secret que l'attaquant doit tenter de lire.

1. **Créer le dossier manuellement** : `mkdir -p /var/www/html`
2. **Créer le fichier "indice" (Hint)** :
 - Commande : `nano /var/www/html/robots.txt`
 - Écrire à l'intérieur : "Le flag est dans /opt/secret_flag.txt mais tu n'as pas les permissions !"
 - Sauvegarde
3. **Créer le véritable "Flag"** :
 - Commande : `nano /opt/secret_flag.txt`
 - Écrire le code secret : `FLAG{RESEAU_MASTER_2026}`
4. **Configurer les permissions (Très important !)** :
 - En tant que root, taper : `chmod 600 /opt/secret_flag.txt`



```
bouboulito@debian: ~  
Created symlink '/etc/systemd/system/multi-user.target.wants/apache2.service' →  
'/usr/lib/systemd/system/apache2.service'.  
Created symlink '/etc/systemd/system/multi-user.target.wants/apache-htcacheclean.service' →  
'/usr/lib/systemd/system/apache-htcacheclean.service'.  
Traitement des actions différées (« triggers ») pour man-db (2.13.1-1) ...  
root@debian:~# cd /var/www  
-bash: cd: /var/www: Aucun fichier ou dossier de ce nom  
root@debian:~# cd /var  
root@debian:/var# exit  
déconnexion  
bouboulito@debian:~$ su -  
Mot de passe :  
root@debian:~# mkdir -p /var/www/html  
root@debian:~# nano /var/www/html/robots.txt  
root@debian:~# nano /opt/secret_flag.txt  
root@debian:~# chmod 600 /opt/secret_flag.txt  
root@debian:~# adduser assistant  
Nouveau mot de passe :  
Retapez le nouveau mot de passe :  
passwd : mot de passe mis à jour avec succès  
Modifier les informations associées à un utilisateur pour assistant  
Entrer la nouvelle valeur, ou appuyer sur ENTER pour la valeur par défaut  
NOM []:  
Numéro de chambre []:  
Téléphone professionnel []:
```

Étape 2 : Créer l'utilisateur "victime" et cacher le mot de passe L'attaquant va d'abord pirater un compte utilisateur simple avant de devenir root.

1. **Créer l'utilisateur** :
 - Commande : `adduser assistant`
 - Choisis un mot de passe simple à cacher : `password1234`.
2. **Cacher le mot de passe (La piste)** : On utilise un faux script de maintenance.

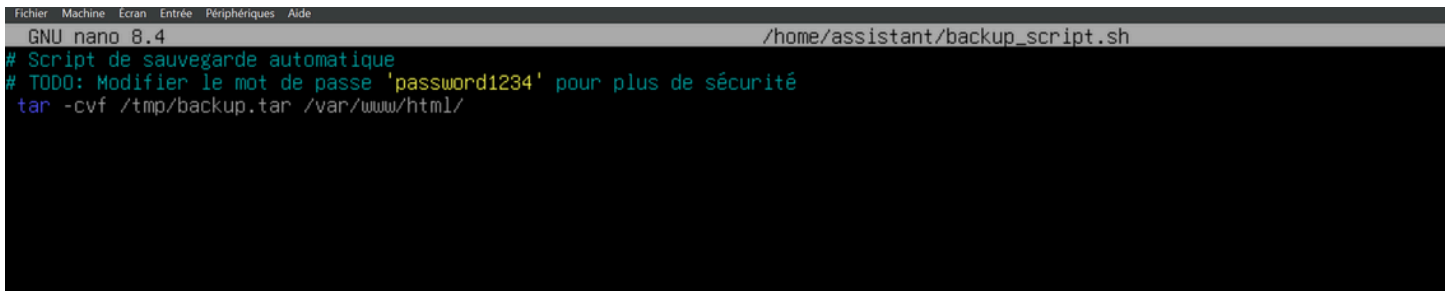
- Commande : nano /home/assistant/backup_script.sh
- Écris ceci :

Bash

Script de sauvegarde automatique

TODO: Modifier le mot de passe 'password1234' pour plus de sécurité

tar -cvf /tmp/backup.tar /var/www/html/

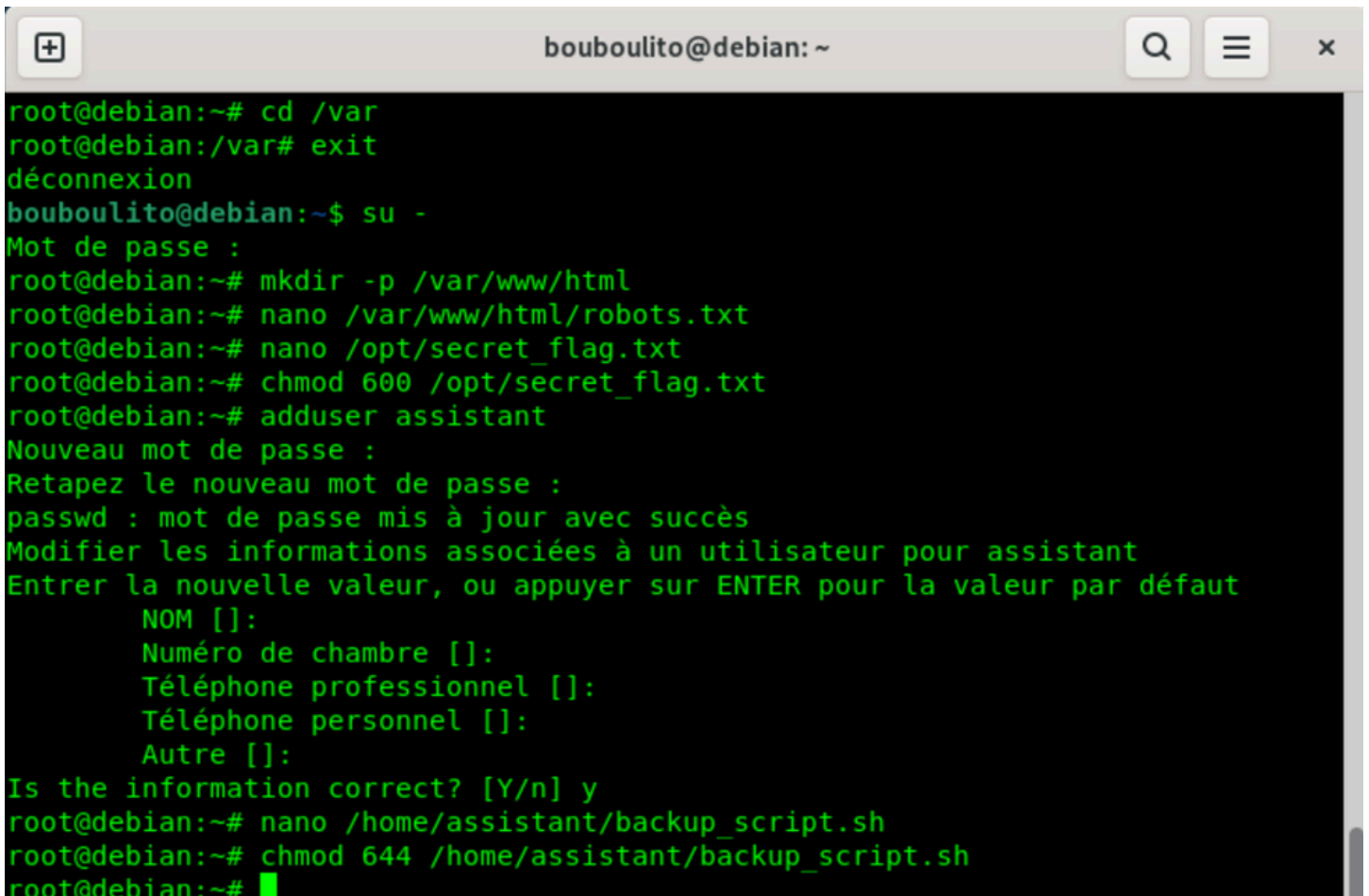


```

Fichier  Machine  Ecran  Entrée  Périphériques  Aide
GNU nano 8.4                                     /home/assistant/backup_script.sh
# Script de sauvegarde automatique
# TODO: Modifier le mot de passe 'password1234' pour plus de sécurité
tar -cvf /tmp/backup.tar /var/www/html/
  
```

1. Ajuster les permissions :

- Commande : chmod 644 /home/assistant/backup_script.sh
- **Pourquoi ?** Pour que l'attaquant puisse trouver ce fichier et le lire sans être encore connecté comme assistant.



```

bouboulito@debian: ~
root@debian:~# cd /var
root@debian:/var# exit
déconnexion
bouboulito@debian:~$ su -
Mot de passe :
root@debian:~# mkdir -p /var/www/html
root@debian:~# nano /var/www/html/robots.txt
root@debian:~# nano /opt/secret_flag.txt
root@debian:~# chmod 600 /opt/secret_flag.txt
root@debian:~# adduser assistant
Nouveau mot de passe :
Retapez le nouveau mot de passe :
passwd : mot de passe mis à jour avec succès
Modifier les informations associées à un utilisateur pour assistant
Entrer la nouvelle valeur, ou appuyer sur ENTER pour la valeur par défaut
  NOM []:
  Numéro de chambre []:
  Téléphone professionnel []:
  Téléphone personnel []:
  Autre []:
Is the information correct? [Y/n] y
root@debian:~# nano /home/assistant/backup_script.sh
root@debian:~# chmod 644 /home/assistant/backup_script.sh
root@debian:~#
  
```

Étape 3 : Configurer l'énigme de l'escalade (Accès Root) Pour passer de assistant à root, l'attaquant doit trouver un indice caché menant au mot de passe root.

1. **Créer un fichier caché (Dotfile) :**

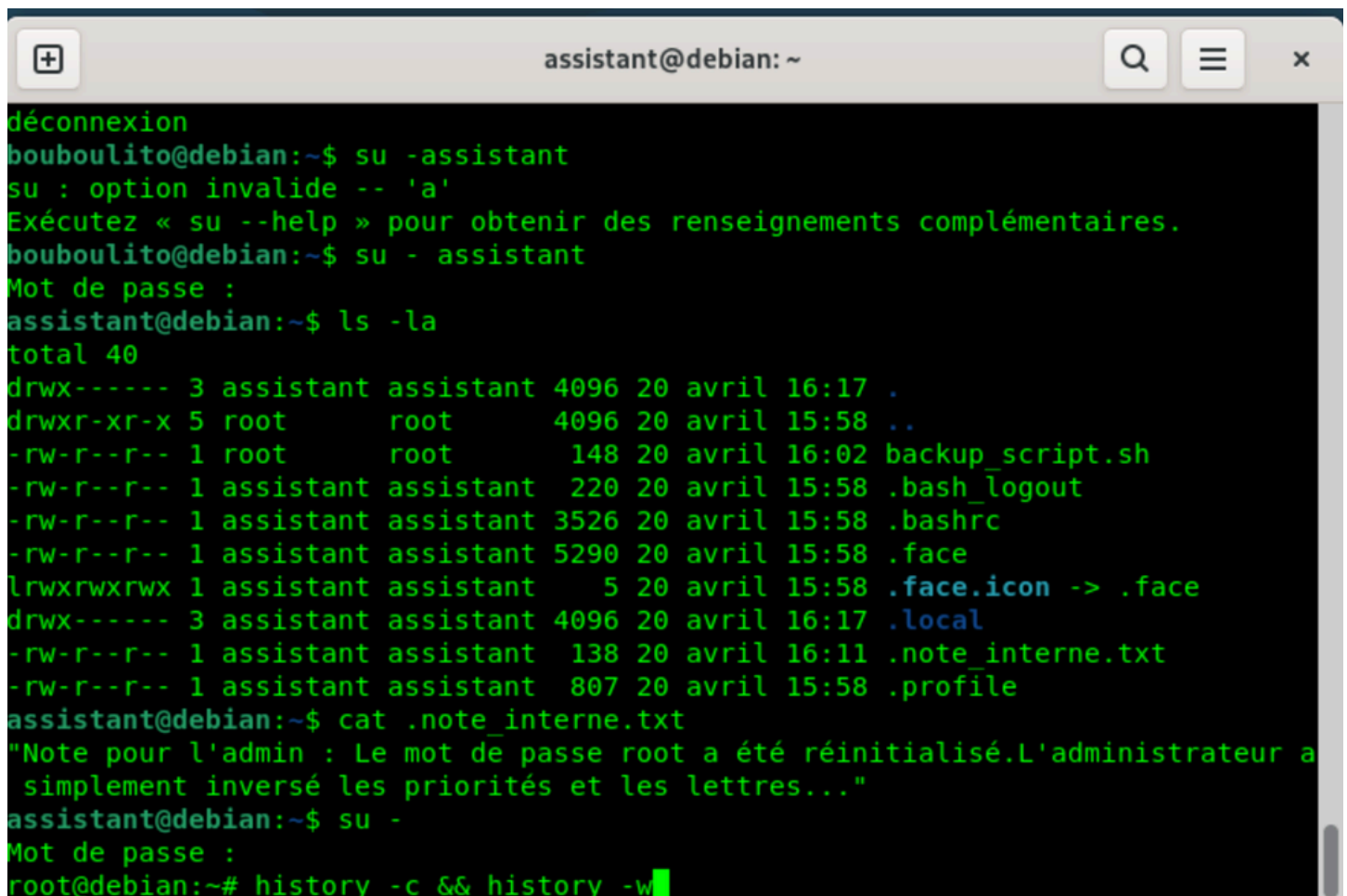
- Assure-toi d'être dans le dossier de l'assistant : `cd /home/assistant`
- Commande : `nano .note_interne.txt` (Le point au début rend le fichier caché).
- Écris : "L'administrateur a simplement inversé les priorités... et les lettres."

2. **Configurer le mot de passe Root :**

- Commande : `passwd root`
- Action : Tape toor (l'inverse de root), puis valide deux fois.

3. **Vérifier les permissions du fichier caché :**

- Commande : `chown assistant:assistant /home/assistant/.note_interne.txt`
- Commande : `chmod 644 /home/assistant/.note_interne.txt`
- **Pourquoi ?** En français : "On s'assure que l'utilisateur a les droits de lecture sur sa propre note."



```
assistant@debian: ~  
déconnexion  
bouboulito@debian:~$ su -assistant  
su : option invalide -- 'a'  
Exécutez « su --help » pour obtenir des renseignements complémentaires.  
bouboulito@debian:~$ su - assistant  
Mot de passe :  
assistant@debian:~$ ls -la  
total 40  
drwx----- 3 assistant assistant 4096 20 avril 16:17 .  
drwxr-xr-x 5 root      root      4096 20 avril 15:58 ..  
-rw-r--r-- 1 root      root      148 20 avril 16:02 backup_script.sh  
-rw-r--r-- 1 assistant assistant  220 20 avril 15:58 .bash_logout  
-rw-r--r-- 1 assistant assistant 3526 20 avril 15:58 .bashrc  
-rw-r--r-- 1 assistant assistant 5290 20 avril 15:58 .face  
lrwxrwxrwx 1 assistant assistant   5 20 avril 15:58 .face.icon -> .face  
drwx----- 3 assistant assistant 4096 20 avril 16:17 .local  
-rw-r--r-- 1 assistant assistant  138 20 avril 16:11 .note_interne.txt  
-rw-r--r-- 1 assistant assistant  807 20 avril 15:58 .profile  
assistant@debian:~$ cat .note_interne.txt  
"Note pour l'admin : Le mot de passe root a été réinitialisé.L'administrateur a  
simplement inversé les priorités et les lettres..."  
assistant@debian:~$ su -  
Mot de passe :  
root@debian:~# history -c && history -w
```

Niveau 2 L'abus de droits SUDO (privilege Escalation)

C'est un classique indispensable.

- **Le concept :** Tu autorises l'utilisateur à lancer un programme très spécifique en tant que root, mais ce programme permet de s'échapper vers le terminal.

- **L'action** : Modifie le fichier /etc/sudoers pour permettre à l'utilisateur de lancer find ou vim sans mot de passe.

- **Le défi** : L'attaquant doit trouver comment utiliser sudo find . -exec /bin/sh \; -quit pour devenir root. C'est ce qu'on appelle exploiter les **GTFOBins**.

Étape 1 : Installer l'outil nécessaire

Assure-toi que l'éditeur **Vim** est présent sur la machine cible.

- **Commande** : apt update && apt install vim -y

Étape 2 : Configurer la vulnérabilité (Le fichier Sudoers)

Nous allons autoriser l'utilisateur assistant à lancer Vim avec les droits root, mais **sans demander de mot de passe**. C'est ici que se trouve la faille.

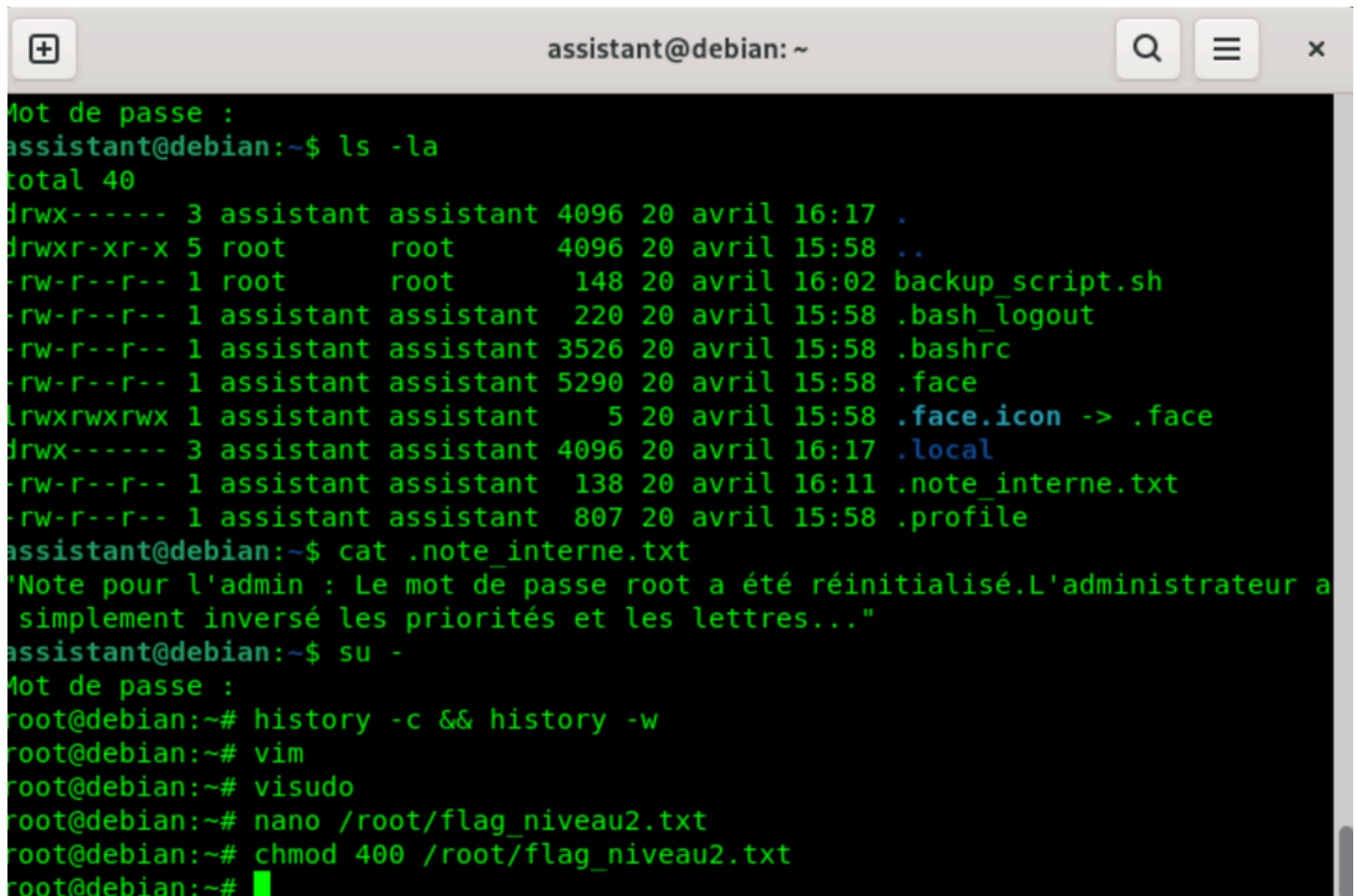
1. Ouvre le fichier de configuration des privilèges avec la commande sécurisée :

- o **Commande** : visudo

2. Ajoute cette ligne tout à la fin du fichier :

- o assistant ALL=(ALL) NOPASSWD: /usr/bin/vim

3. Sauvegarde et quitte (Ctrl+O, Entrée, puis Ctrl+X).

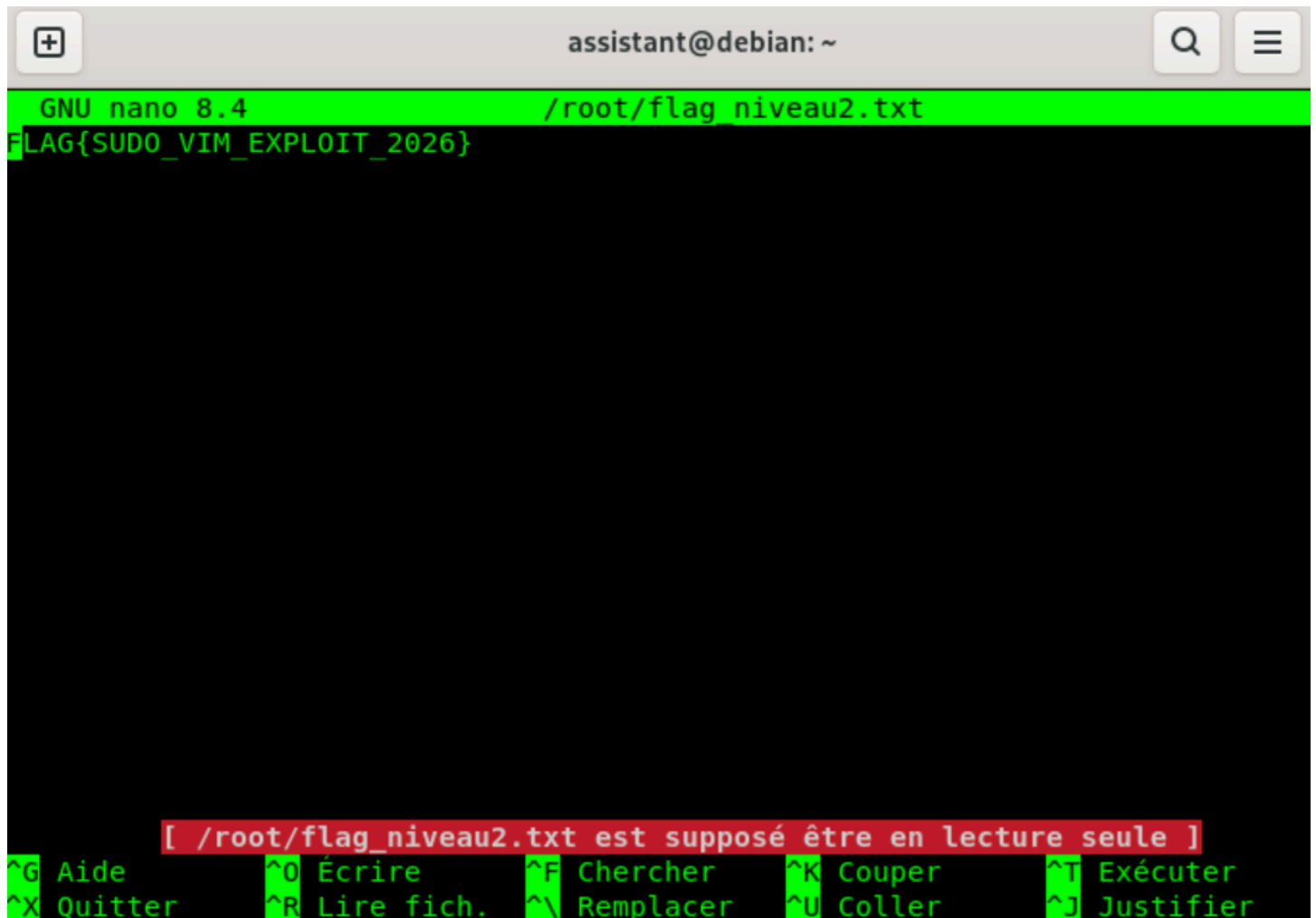


```
assistant@debian: ~
Mot de passe :
assistant@debian:~$ ls -la
total 40
drwx----- 3 assistant assistant 4096 20 avril 16:17 .
drwxr-xr-x 5 root      root      4096 20 avril 15:58 ..
-rw-r--r-- 1 root      root        148 20 avril 16:02 backup_script.sh
-rw-r--r-- 1 assistant assistant  220 20 avril 15:58 .bash_logout
-rw-r--r-- 1 assistant assistant 3526 20 avril 15:58 .bashrc
-rw-r--r-- 1 assistant assistant 5290 20 avril 15:58 .face
lrwxrwxrwx 1 assistant assistant   5 20 avril 15:58 .face.icon -> .face
drwx----- 3 assistant assistant 4096 20 avril 16:17 .local
-rw-r--r-- 1 assistant assistant  138 20 avril 16:11 .note_interne.txt
-rw-r--r-- 1 assistant assistant  807 20 avril 15:58 .profile
assistant@debian:~$ cat .note_interne.txt
'Note pour l'admin : Le mot de passe root a été réinitialisé.L'administrateur a
simplement inversé les priorités et les lettres...'
assistant@debian:~$ su -
Mot de passe :
root@debian:~# history -c && history -w
root@debian:~# vim
root@debian:~# visudo
root@debian:~# nano /root/flag_niveau2.txt
root@debian:~# chmod 400 /root/flag_niveau2.txt
root@debian:~#
```

Étape 3 : Créer le Flag de victoire

Le "Flag" doit se trouver dans un endroit accessible uniquement par **root**.

1. **Commande** : `nano /root/flag_niveau2.txt`
2. Écris ton code secret, par exemple : `FLAG{SUDO_VIM_EXPLOIT_2026}`
3. Protège le fichier : `chmod 400 /root/flag_niveau2.txt` (Seul root pourra le lire).



```
assistant@debian: ~
GNU nano 8.4 /root/flag_niveau2.txt
FLAG{SUDO_VIM_EXPLOIT_2026}

[ /root/flag_niveau2.txt est supposé être en lecture seule ]
^G Aide      ^O Écrire    ^F Chercher  ^K Couper    ^T Exécuter
^X Quitter   ^R Lire fich. ^\ Remplacer  ^U Coller    ^J Justifier
```

Niveau 3 : Exploitation d'une tâche automatisée (Cron Job)

Objectif : Comprendre l'automatisation et les dangers des permissions globales (777).

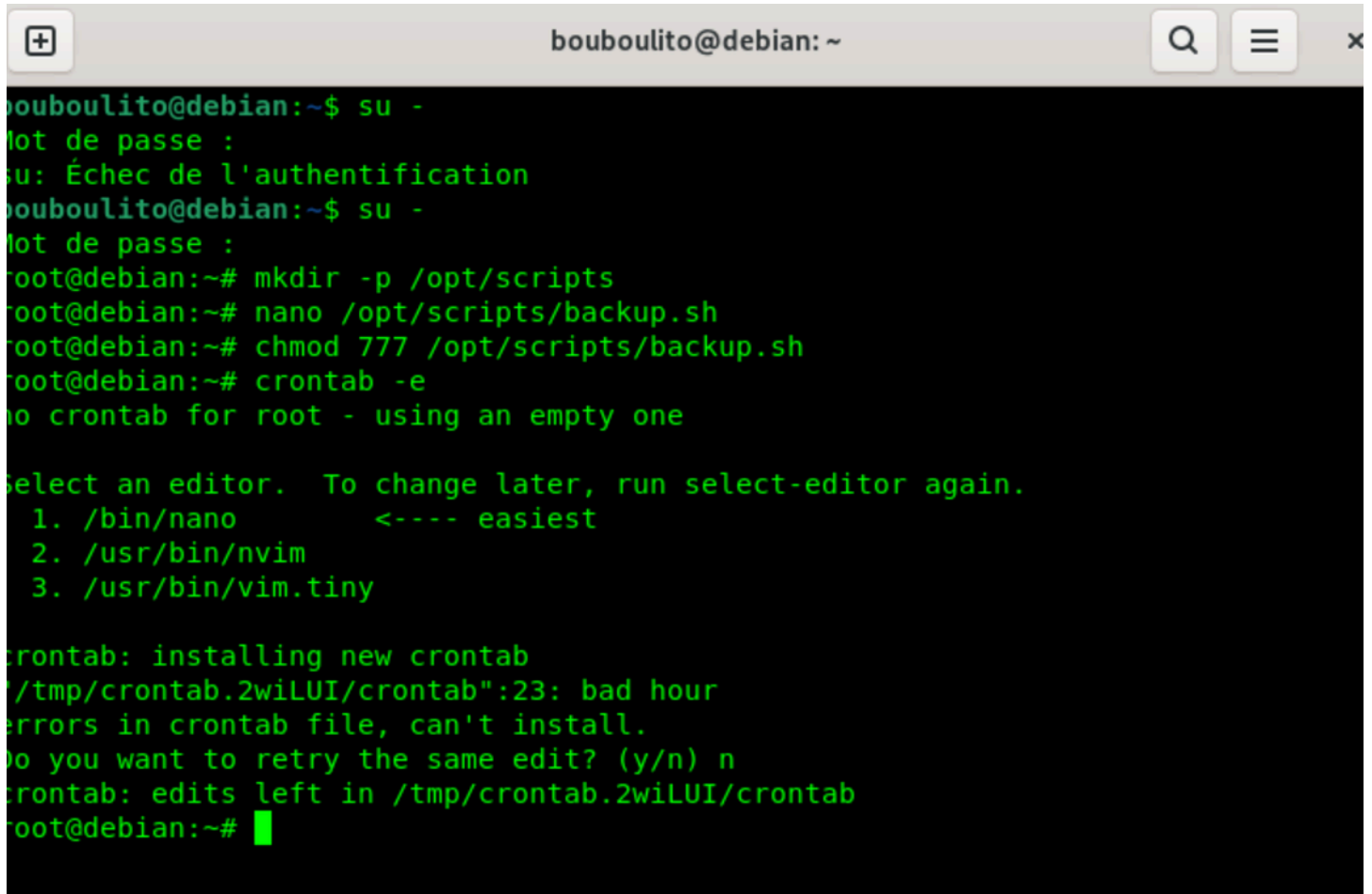
Mise en place (Côté Administrateur)

1. Créer le script vulnérable :

- o `mkdir -p /opt/scripts`
- o `nano /opt/scripts/backup.sh`
- o Contenu : `#!/bin/bash \n cp /var/www/html/robots.txt /tmp/robots_backup.txt`

2. Rendre le script modifiable par l'utilisateur "assistant" :

o `chmod 777 /opt/scripts/backup.sh`

A terminal window titled 'bouboulito@debian: ~' with search and menu icons in the top right. The terminal shows a user attempting to switch to root with 'su -' and failing due to a missing password. They then successfully switch to root. The root user runs a series of commands: 'mkdir -p /opt/scripts', 'nano /opt/scripts/backup.sh', 'chmod 777 /opt/scripts/backup.sh', and 'crontab -e'. The crontab editor shows a list of editors with nano selected as the easiest. After saving, an error message appears: '/tmp/crontab.2wiLUI/crontab':23: bad hour'. The user is asked if they want to retry the same edit, and they respond with 'n'. The terminal ends with the root prompt.

```
bouboulito@debian:~$ su -
Mot de passe :
su: Échec de l'authentification
bouboulito@debian:~$ su -
Mot de passe :
root@debian:~# mkdir -p /opt/scripts
root@debian:~# nano /opt/scripts/backup.sh
root@debian:~# chmod 777 /opt/scripts/backup.sh
root@debian:~# crontab -e
no crontab for root - using an empty one

Select an editor. To change later, run select-editor again.
 1. /bin/nano          <---- easiest
 2. /usr/bin/nvim
 3. /usr/bin/vim.tiny

crontab: installing new crontab
'/tmp/crontab.2wiLUI/crontab':23: bad hour
errors in crontab file, can't install.
Do you want to retry the same edit? (y/n) n
crontab: edits left in /tmp/crontab.2wiLUI/crontab
root@debian:~#
```

3. Automatiser l'exécution en root :

o `crontab -e`

o Ajoute la ligne : `* * * * * /opt/scripts/backup.sh` (s'exécute chaque minute).

```
bouboulito@debian: ~
GNU nano 8.4 /tmp/crontab.tDzCR2/crontab *
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
* * * * * opt/scripts/backup.sh
^G Aide      ^O Écrire    ^F Chercher  ^K Couper    ^T Exécuter  ^C Emplacement
^X Quitter   ^R Lire fich.^N Remplacer  ^U Coller    ^J Justifier ^_ Aller ligne
```

```
bouboulito@debian: ~
root@debian:~# crontab -e
no crontab for root - using an empty one
No modification made
root@debian:~# exit
exit
root@debian:~# exit
déconnexion
bouboulito@debian:~$ crontab -e
no crontab for bouboulito - using an empty one
Select an editor. To change later, run select-editor again.
 1. /bin/nano          <---- easiest
 2. /usr/bin/nvim
 3. /usr/bin/vim.tiny

Choose 1-3 [1]:
crontab: installing new crontab
bouboulito@debian:~$ crontab -e
No modification made
bouboulito@debian:~$ su -
Mot de passe :
root@debian:~# crontab -e
no crontab for root - using an empty one
crontab: installing new crontab
root@debian:~#
```

Description du challenge (Pour le Wiki)

"Un script de maintenance tourne chaque minute sur le serveur avec des privilèges élevés. Si seulement vous pouviez 'aider' ce script à faire autre chose que sa sauvegarde habituelle..."

Niveau 4 : Détournement de tâche Cron (Cron Hijacking)

C'est une technique redoutable basée sur l'automatisation.

- **Le concept** : Le système exécute automatiquement des scripts de maintenance en tant que root. Si un utilisateur peut modifier l'un de ces scripts, il peut forcer le système à exécuter ses propres commandes avec les privilèges maximum.
- **L'action** : Créer un script avec des permissions trop larges (777) et l'ajouter à la table de planification (crontab) de l'administrateur.
- **Le défi** : L'attaquant doit identifier les scripts qui tournent régulièrement et injecter un code (comme un "Reverse Shell" ou un changement de permissions) pour prendre le contrôle.

Étape 1 : Créer le script vulnérable Nous allons simuler un script de sauvegarde qui appartient à root mais que tout le monde peut modifier.

- **Commande** : `mkdir -p /opt/scripts`
- **Commande** : `nano /opt/scripts/backup.sh`
- **Écris à l'intérieur** :

```
Bash
```

```
#!/bin/bash
```

```
# Sauvegarde automatique des logs
```

```
cp /var/log/auth.log /tmp/auth_backup.log
```

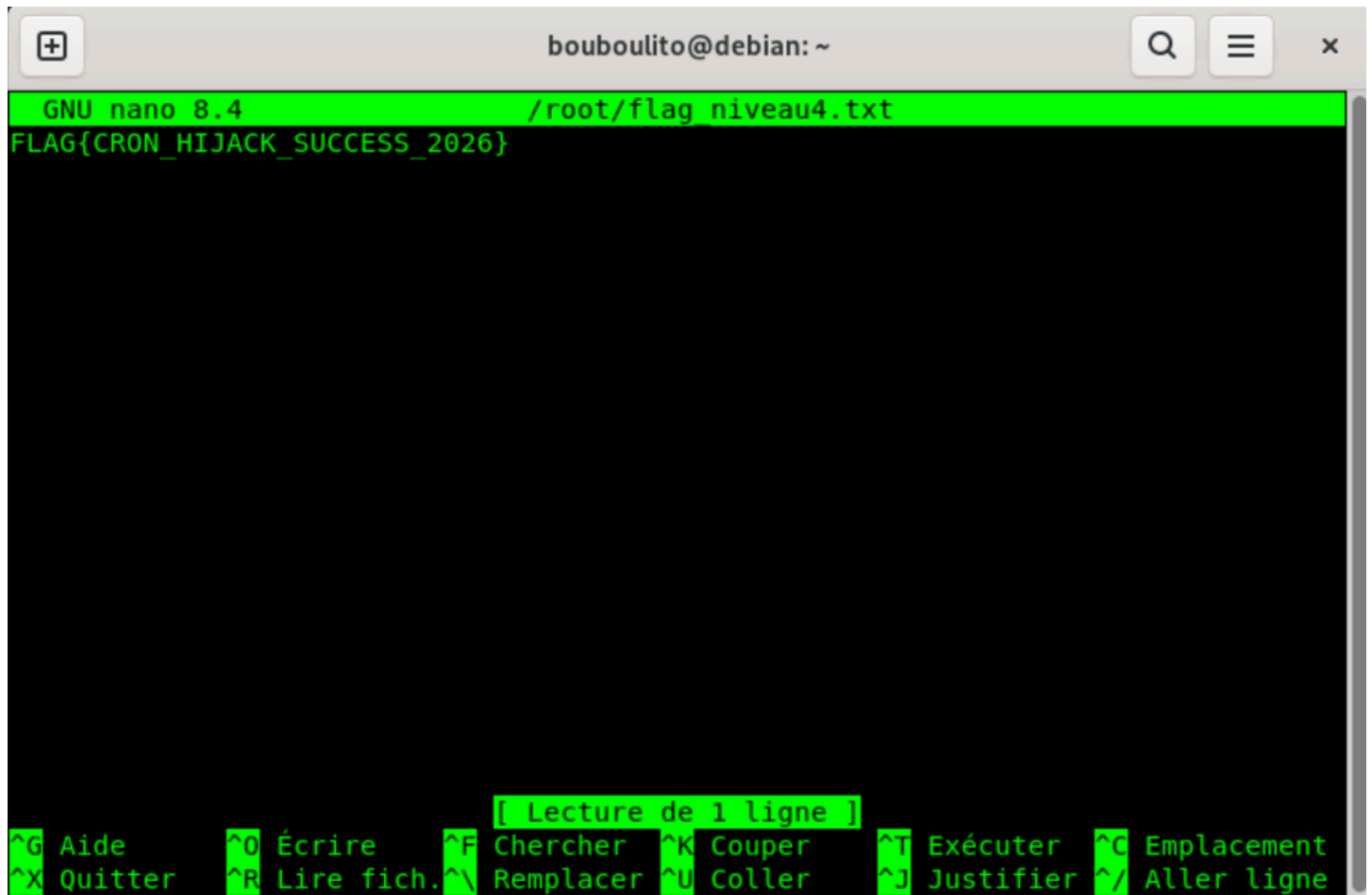
- **Permissions (La faille)** : `chmod 777 /opt/scripts/backup.sh`

Étape 2 : Configurer l'automatisation (Cron) Il faut maintenant dire au système de lancer ce script chaque minute.

1. **Ouvre le planificateur** : `crontab -e`
2. **Ajoute cette ligne tout à la fin** : `* * * * * /opt/scripts/backup.sh`
3. **Sauvegarde et quitte.**

Étape 3 : Créer le Flag Le flag sera placé dans le répertoire personnel de root.

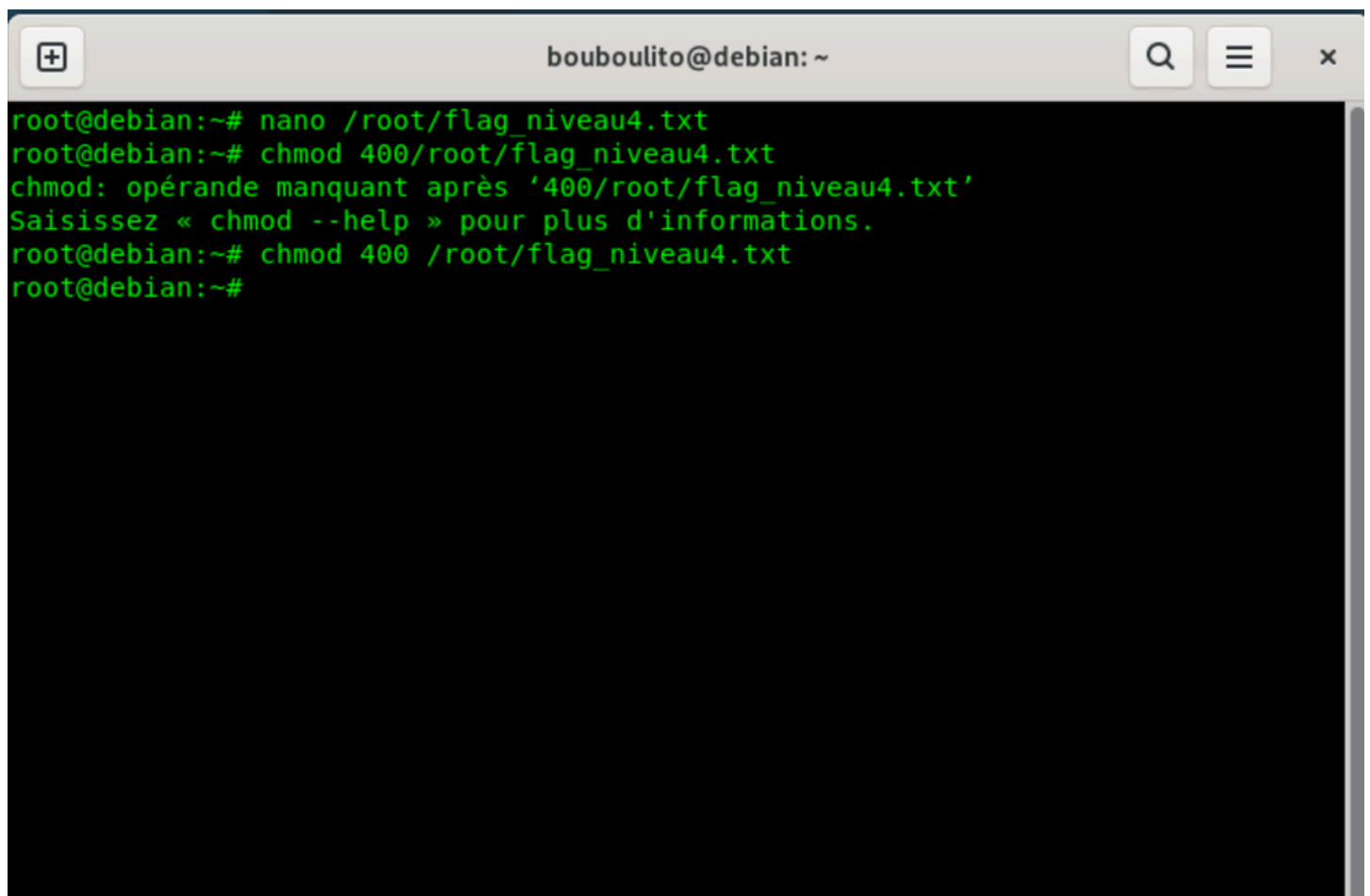
1. **Commande** : `nano /root/flag_niveau4.txt`
2. **Écris le code secret** : `FLAG{CRON_HIJACK_SUCCESS_2026}`



```
GNU nano 8.4 /root/flag_niveau4.txt
FLAG{CRON_HIJACK_SUCCESS_2026}

[ Lecture de 1 ligne ]
^G Aide      ^O Écrire    ^F Chercher  ^K Couper    ^T Exécuter  ^C Emplacement
^X Quitter   ^R Lire fich.^N Remplacer  ^U Coller    ^J Justifier  ^_ Aller ligne
```

3. **Protège le fichier** : `chmod 400 /root/flag_niveau4.txt`



```
root@debian:~# nano /root/flag_niveau4.txt
root@debian:~# chmod 400/root/flag_niveau4.txt
chmod: opérande manquant après '400/root/flag_niveau4.txt'
Saisissez « chmod --help » pour plus d'informations.
root@debian:~# chmod 400 /root/flag_niveau4.txt
root@debian:~#
```

Niveau 5 L'exploitation du Bit SUID (Set User ID)

C'est le défi final qui utilise les permissions avancées d'Unix.

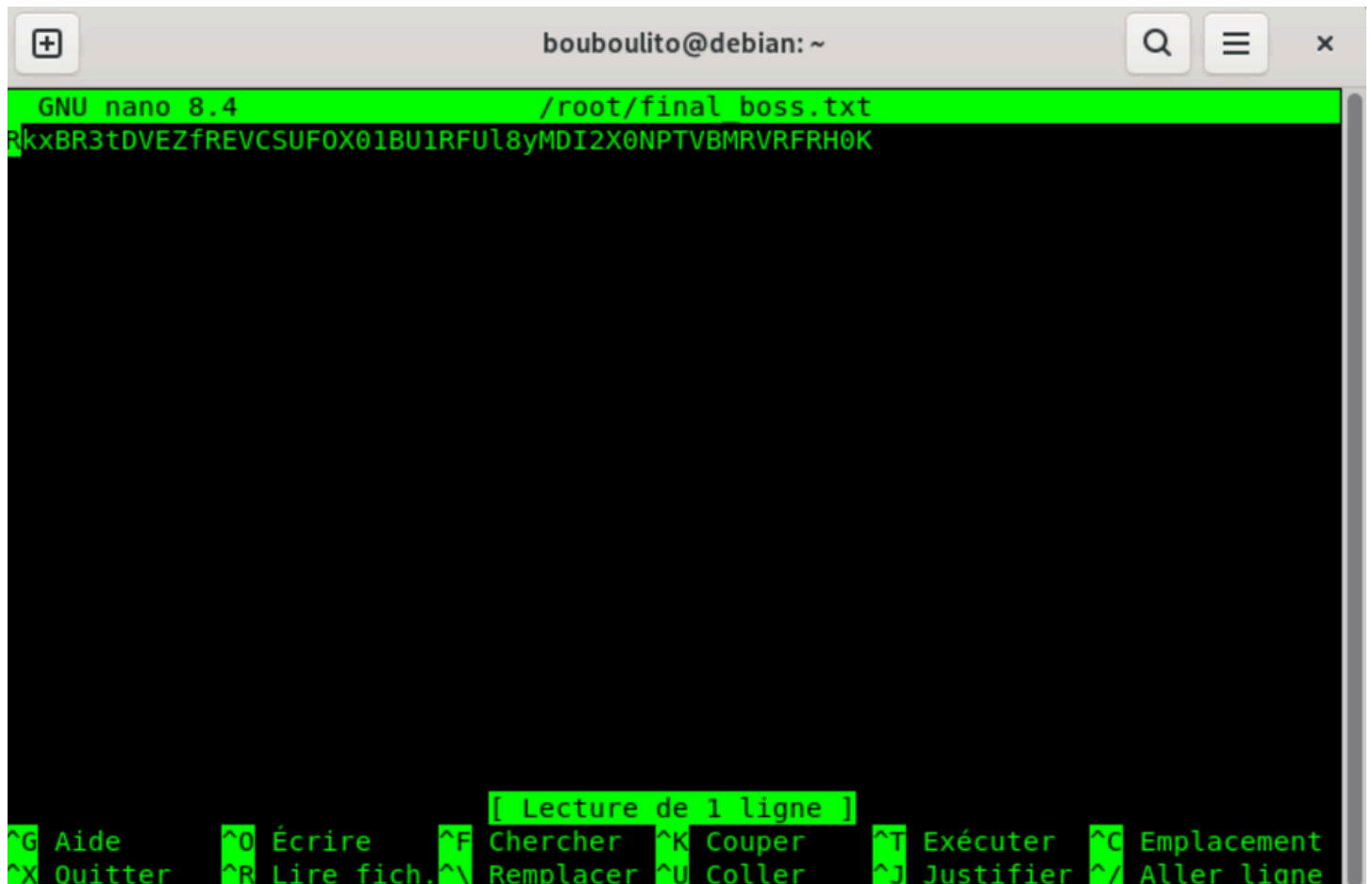
- **Le concept** : Le bit SUID permet à un utilisateur de lancer un programme avec les permissions du propriétaire (root). Si on active ce bit sur un outil de lecture, l'utilisateur peut lire n'importe quel secret sur le disque.
- **L'action** : Créer une copie d'un utilitaire système (cat) et lui donner des droits spéciaux pour qu'il devienne une "porte dérobée".
- **Le défi** : L'attaquant doit scanner le système pour trouver les fichiers SUID et découvrir qu'il peut lire le flag final, qui est en plus encodé pour corser l'épreuve.

Étape 1 : Créer le binaire à privilèges Nous allons créer une version de cat nommée super_reader qui a des pouvoirs de root.

- **Commande** : `cp /bin/cat /usr/bin/super_reader`
- **Commande** : `chown root:root /usr/bin/super_reader`
- **Commande (Activer le SUID)** : `chmod 4755 /usr/bin/super_reader`

Étape 2 : Créer et encoder le Flag Final Pour ce dernier niveau, nous allons utiliser l'obfuscation.

1. **Génère ton flag en Base64** : `echo "FLAG{CTF_DEBIAN_MASTER_2026_COMPLETED}" | base64`
2. **Copie le résultat obtenu** (exemple :
RkxBR3tDVEZfREVCSUFOX01BU1RFUI8yMDI2X0NPTVBMRVRFRH0K).



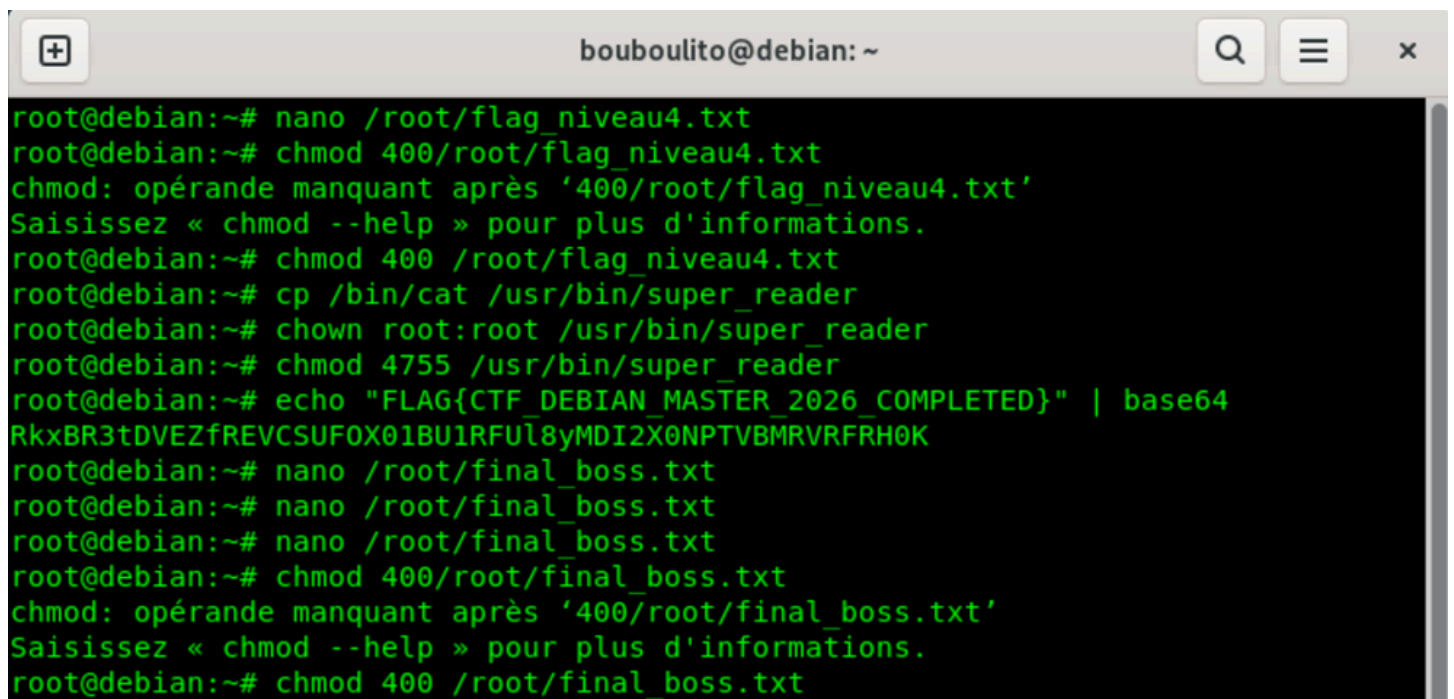
```
GNU nano 8.4 /root/final_boss.txt
RkxBR3tDVEZfREVCSUF0X01BU1RFU8yMDI2X0NPTVBMRVRFRH0K

[ Lecture de 1 ligne ]
^G Aide      ^O Écrire    ^F Chercher  ^K Couper    ^T Exécuter  ^C Emplacement
^X Quitter   ^R Lire fich.^N Remplacer  ^U Coller    ^J Justifier ^_ Aller ligne
```

3. **Crée le fichier secret** : `nano /root/final_boss.txt`

4. **Colle la chaîne Base64 à l'intérieur**, sauvegarde et quitte.

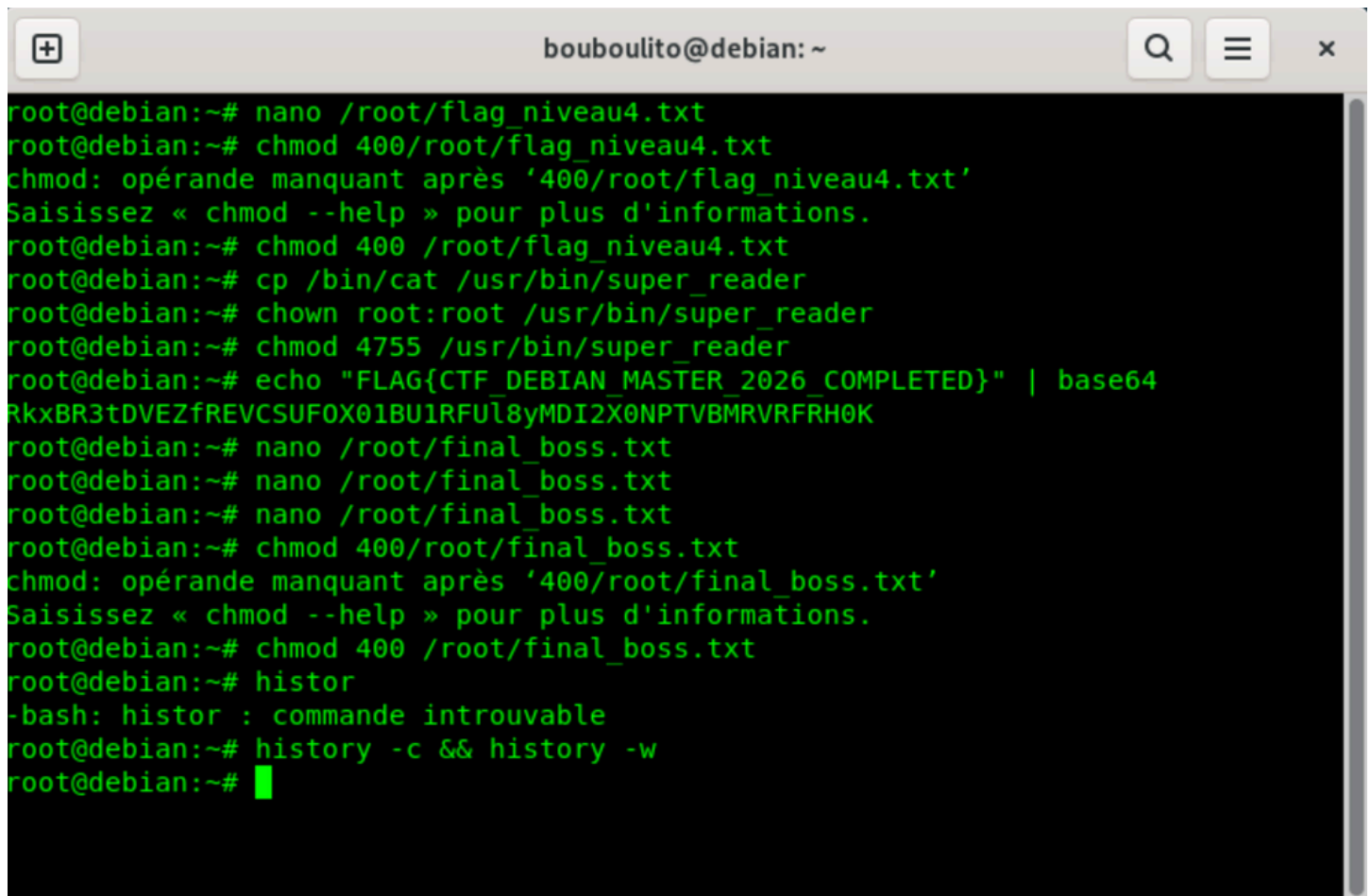
5. **Protège le fichier** : `chmod 400 /root/final_boss.txt`



```
root@debian:~# nano /root/flag_niveau4.txt
root@debian:~# chmod 400 /root/flag_niveau4.txt
chmod: opérande manquant après '400/root/flag_niveau4.txt'
Saisissez « chmod --help » pour plus d'informations.
root@debian:~# chmod 400 /root/flag_niveau4.txt
root@debian:~# cp /bin/cat /usr/bin/super_reader
root@debian:~# chown root:root /usr/bin/super_reader
root@debian:~# chmod 4755 /usr/bin/super_reader
root@debian:~# echo "FLAG{CTF_DEBIAN_MASTER_2026_COMPLETED}" | base64
RkxBR3tDVEZfREVCSUF0X01BU1RFU8yMDI2X0NPTVBMRVRFRH0K
root@debian:~# nano /root/final_boss.txt
root@debian:~# nano /root/final_boss.txt
root@debian:~# nano /root/final_boss.txt
root@debian:~# chmod 400 /root/final_boss.txt
chmod: opérande manquant après '400/root/final_boss.txt'
Saisissez « chmod --help » pour plus d'informations.
root@debian:~# chmod 400 /root/final_boss.txt
```

Étape 3 : Nettoyer les traces (Optionnel mais recommandé) Pour que le participant ne puisse pas simplement tricher en regardant ce que tu as fait :

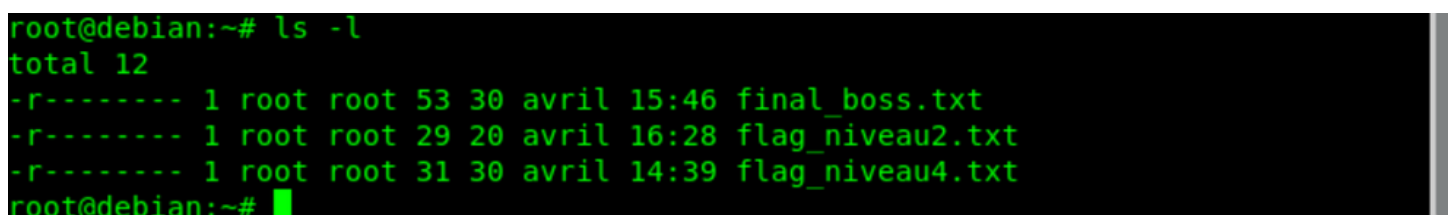
· **Commande** : `history -c && history -w`

A terminal window titled 'bouboulito@debian: ~' with search, menu, and close buttons. The terminal shows a series of commands to create files and clean up history. The commands are: `nano /root/flag_niveau4.txt`, `chmod 400 /root/flag_niveau4.txt` (which fails with a message about a missing operand), `chmod 400 /root/flag_niveau4.txt` (which succeeds), `cp /bin/cat /usr/bin/super_reader`, `chown root:root /usr/bin/super_reader`, `chmod 4755 /usr/bin/super_reader`, `echo "FLAG{CTF_DEBIAN_MASTER_2026_COMPLETED}" | base64` (outputting `RkxBR3tDVEZfREVCSUFOX01BU1RFU18yMDI2X0NPTVBMRVRFH0K`), `nano /root/final_boss.txt` (repeated three times), `chmod 400 /root/final_boss.txt` (which fails with the same message as before), `chmod 400 /root/final_boss.txt` (which succeeds), `history` (outputting `-bash: history : commande introuvable`), and finally `history -c && history -w`. The prompt returns to `root@debian:~#`.

```
root@debian:~# nano /root/flag_niveau4.txt
root@debian:~# chmod 400 /root/flag_niveau4.txt
chmod: opérande manquant après '400/root/flag_niveau4.txt'
Saisissez « chmod --help » pour plus d'informations.
root@debian:~# chmod 400 /root/flag_niveau4.txt
root@debian:~# cp /bin/cat /usr/bin/super_reader
root@debian:~# chown root:root /usr/bin/super_reader
root@debian:~# chmod 4755 /usr/bin/super_reader
root@debian:~# echo "FLAG{CTF_DEBIAN_MASTER_2026_COMPLETED}" | base64
RkxBR3tDVEZfREVCSUFOX01BU1RFU18yMDI2X0NPTVBMRVRFH0K
root@debian:~# nano /root/final_boss.txt
root@debian:~# nano /root/final_boss.txt
root@debian:~# nano /root/final_boss.txt
root@debian:~# chmod 400 /root/final_boss.txt
chmod: opérande manquant après '400/root/final_boss.txt'
Saisissez « chmod --help » pour plus d'informations.
root@debian:~# chmod 400 /root/final_boss.txt
root@debian:~# history
-bash: history : commande introuvable
root@debian:~# history -c && history -w
root@debian:~#
```

Conseils de mise en œuvre (Best Practices)

- **Audit des fichiers** : Vérifiez toujours avec `ls -l` que vos fichiers `flag.txt` ne sont pas lisibles par un utilisateur simple sans l'exploit prévu.
- **Nettoyage final** : Avant de distribuer la VM, effacez vos traces :
- Bash
- `history -c && history -w`
- **Stabilité** : Testez la chaîne complète de l'utilisateur `level0` jusqu'à `root` avant de valider la Snapshot finale.

A terminal window showing the output of the `ls -l` command. The output lists three files: `final_boss.txt` (permissions `-r-----`, size 53, date avril 15:46), `flag_niveau2.txt` (permissions `-r-----`, size 29, date avril 16:28), and `flag_niveau4.txt` (permissions `-r-----`, size 31, date avril 14:39). The prompt returns to `root@debian:~#`.

```
root@debian:~# ls -l
total 12
-r----- 1 root root 53 30 avril 15:46 final_boss.txt
-r----- 1 root root 29 20 avril 16:28 flag_niveau2.txt
-r----- 1 root root 31 30 avril 14:39 flag_niveau4.txt
root@debian:~#
```