créer un script bash pour surveiller un service et alerter en cas de panne

Phase de préparation

Ligne de code

```
#installation du serveur web Apache2
sudo apt update && sudo apt install apache2
#vérifier le status de Apache2
sudo systemctl status apache2
#Simulation d'une panne/arreter le service sur Apache2
sudo systemctl stop apache2
#Démarrer le service Apache2
sudo systemctl start apache2
```

Phase écriture du script

ligne de commande

```
#création du fichier script et mettre le contenu du script dedans
nano surveillance_service.sh
#permission d'exécution du script
chmod +x surveillance_service.sh
#Création d'un fichier log pour la bonne journalisation des tâches
sudo touch /var/log/surveillance_service.log
```

```
#accordé les bonne permissions au fichier log
sudo chmod 664 /var/log/surveillance_service.log
#exécution du script
sudo ./surveillance_service.sh
#consultation du journal
cat /var/log/surveillance_service.log
```

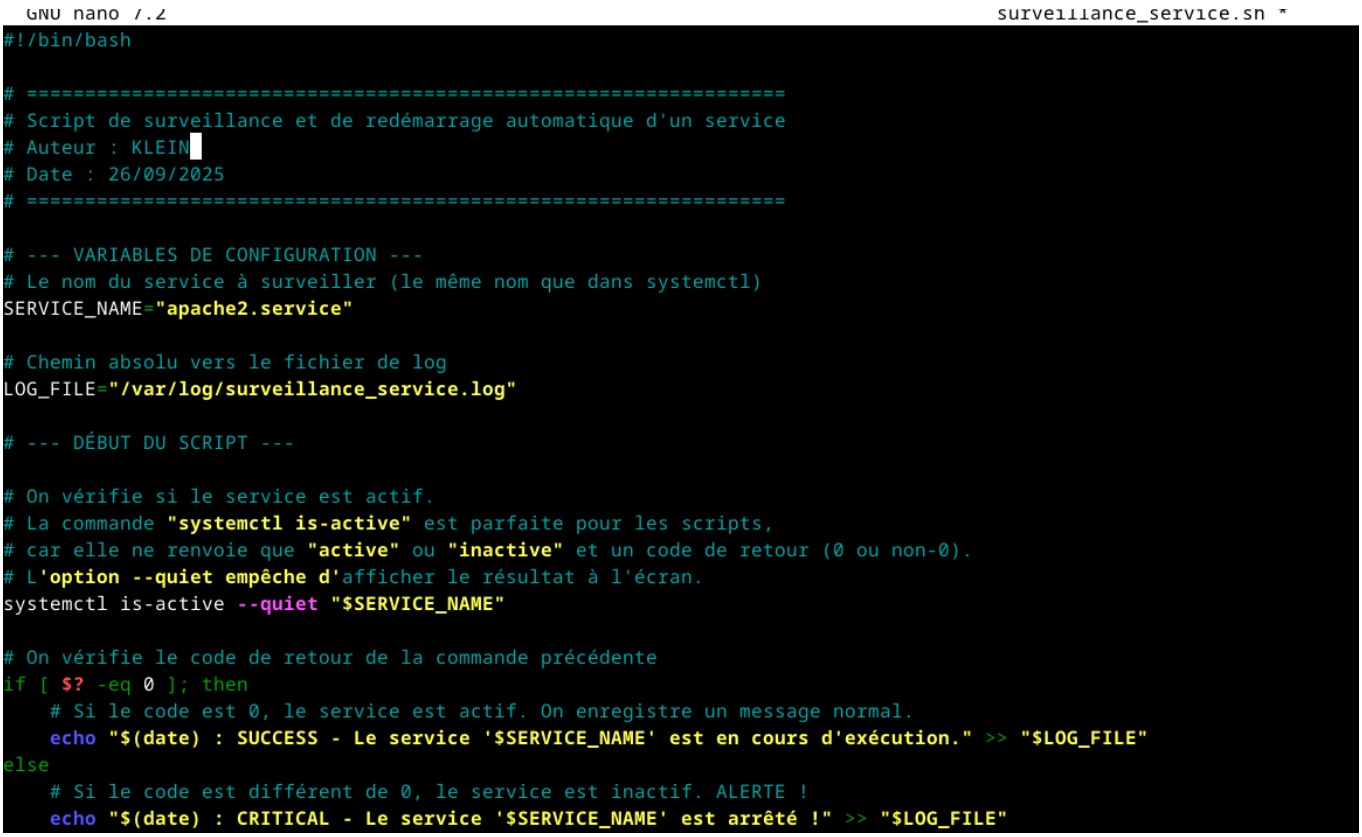
Contenu du script

```

```

```
#!/bin/bash

# ================================================================
# Script de surveillance et de redémarrage automatique d'un service
# Auteur : [Votre Nom]
# Date : 06/08/2025
# ================================================================

# --- VARIABLES DE CONFIGURATION ---
# Le nom du service à surveiller (le même nom que dans systemctl)
SERVICE_NAME="apache2.service"

# Chemin absolu vers le fichier de log
LOG_FILE="/var/log/surveillance_service.log"

# --- DÉBUT DU SCRIPT ---

# On vérifie si le service est actif.
# La commande "systemctl is-active" est parfaite pour les scripts,
# car elle ne renvoie que "active" ou "inactive" et un code de retour (0 ou
non-0).
# L'option --quiet empêche d'afficher le résultat à l'écran.
systemctl is-active --quiet "$SERVICE_NAME"

# On vérifie le code de retour de la commande précédente
if [ $? -eq 0 ]; then
    # Si le code est 0, le service est actif. On enregistre un message normal.
    echo "$(date) : SUCCESS - Le service '$SERVICE_NAME' est en cours
d'exécution." >> "$LOG_FILE"
else
    # Si le code est différent de 0, le service est inactif. ALERTE !
    echo "$(date) : CRITICAL - Le service '$SERVICE_NAME' est arrêté !" >>
"$LOG_FILE"
```

```
    # On tente de redémarrer le service
    echo "$(date) : ACTION - Tentative de redémarrage du service
'$SERVICE_NAME'..." >> "$LOG_FILE"
    systemctl start "$SERVICE_NAME"

    # On vérifie si le redémarrage a fonctionné
    if [ $? -eq 0 ]; then
        echo "$(date) : RECOVERY - Le service '$SERVICE_NAME' a été redémarré
avec succès." >> "$LOG_FILE"
        # C'est ici qu'on pourrait ajouter une commande pour envoyer un email
d'alerte
        # Exemple : mail -s "Alerte Service : $SERVICE_NAME redémarré"
votre.email@domaine.com
    else
        echo "$(date) : FAILURE - Échec du redémarrage du service
'$SERVICE_NAME'." >> "$LOG_FILE"
    fi
fi

exit 0
```

Capture d'écran



```
root@debian:~# nano surveillance_service.sh
root@debian:~# chmod +x surveillance_service.sh
root@debian:~# sudo touch /var/log/surveillance_service.log
root@debian:~# sudo ./surveillance_service.sh
root@debian:~# cat /var/log/surveillance_service.log
ven. 26 sept. 2025 22:14:25 CEST : SUCCESS - Le service 'apache2.service' est en cours d'exécution.
root@debian:~#
```

(test) Simulation d'une panne

ligne de commande

```
#Arreter le service
sudo systemctl stop apache2
#exécution du script
sudo ./surveillance_service.sh
#consultation du journal
cat /var/log/surveillance_service.log
#Vérification du service
sudo systemctl status apache2
#Le script a bien fait son travail
```

Capture d'écran

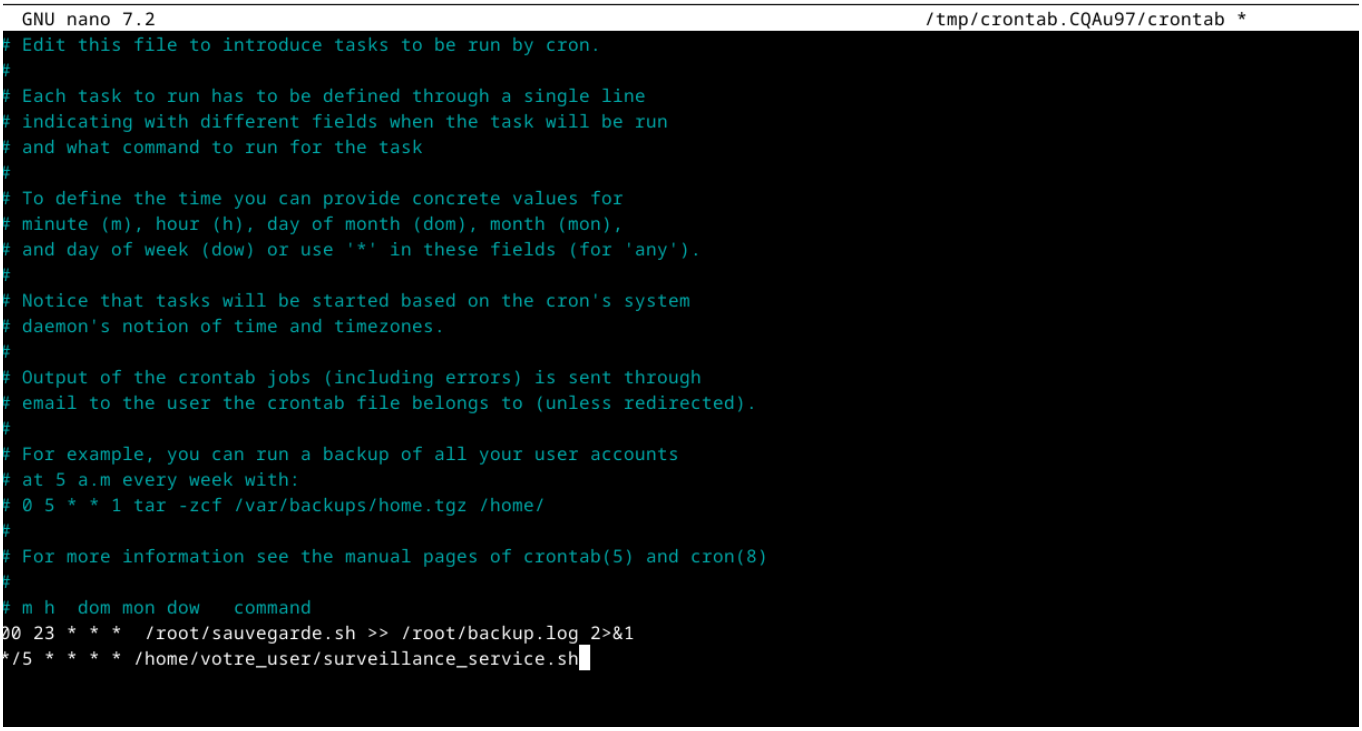
```
root@debian:~# sudo systemctl stop apache2
root@debian:~# sudo ./surveillance_service.sh
root@debian:~# cat /var/log/surveillance_service.log
ven. 26 sept. 2025 22:14:25 CEST : SUCCESS - Le service 'apache2.service' est en cours d'exécution.
ven. 26 sept. 2025 22:18:14 CEST : CRITICAL - Le service 'apache2.service' est arrêté !
ven. 26 sept. 2025 22:18:14 CEST : ACTION - Tentative de redémarrage du service 'apache2.service'...
ven. 26 sept. 2025 22:18:14 CEST : RECOVERY - Le service 'apache2.service' a été redémarré avec succès.
root@debian:~# sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Fri 2025-09-26 22:18:14 CEST; 2min 48s ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 18502 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 18506 (apache2)
      Tasks: 7 (limit: 3397)
     Memory: 27.2M
        CPU: 829ms
     CGroup: /system.slice/apache2.service
             ├─18506 /usr/sbin/apache2 -k start
             ├─18508 /usr/sbin/apache2 -k start
             ├─18509 /usr/sbin/apache2 -k start
             ├─18510 /usr/sbin/apache2 -k start
             ├─18511 /usr/sbin/apache2 -k start
             ├─18512 /usr/sbin/apache2 -k start
             └─18516 /usr/sbin/apache2 -k start
```

Automatisation avec crontab

Ligne de commande

```
#accéder à crontab
sudo crontab -e
#ajouter la ligne de planification pour une éxécution toute les 5 minutes
*/5 * * * * /home/votre_user/surveillance_service.sh
```

Capture d'écran



- **Envoyer une vraie alerte par e-mail** en cas de panne (avec la commande `mail`).
- **Adapter le script** pour qu'il puisse surveiller plusieurs services.
- **Le transformer en un outil plus générique** en passant le nom du service en argument (`./surveillance_service.sh apache2.service`).
- Découvrir les outils de supervision professionnels comme **Zabbix, Nagios ou Prometheus**, qui sont basés sur ces mêmes principes mais à une échelle beaucoup plus grande.