



Práctica Evaluación Continua

Administración de Sistemas

Grupo PA2

Álvaro García Sánchez - 70924450V - agarsan@usal.es

Script de arranque en Linux para notificar arranque y apagado del sistema por Discord

Índice

1. Introducción.....	2
2. Script en Python.....	2
3. Servicio de Arranque.....	5
4. Servicio de Parada.....	6
5. Prueba.....	7

1. Introducción

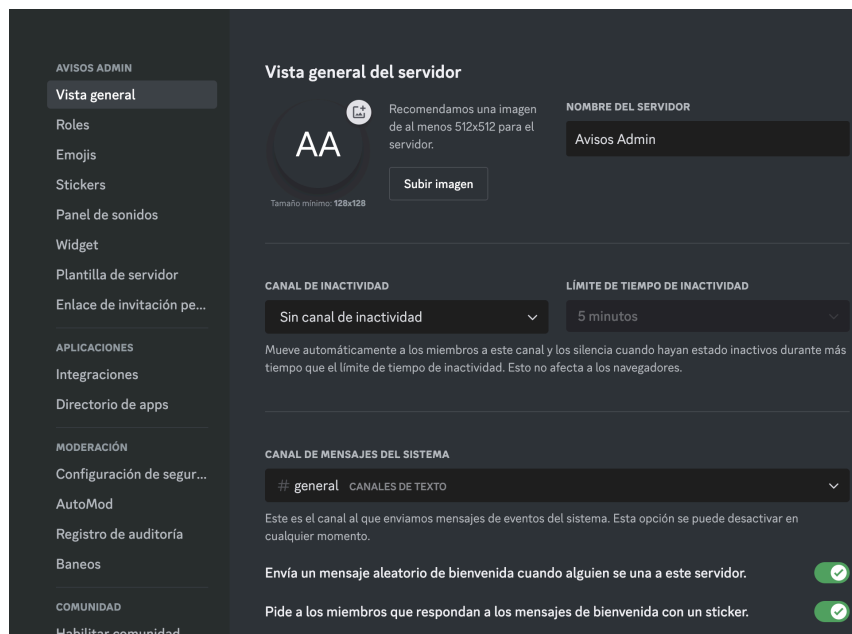
En esta breve práctica se pide la creación y configuración de un script de arranque que nos permita registrar la fecha y hora en la que se ha encendido y se ha apagado el sistema operativo. En mi caso estaré usando la distribución de Linux, Ubuntu, en su versión “**Fossa**”, que tiene un funcionamiento similar al sistema Debian empleado en clase en las sesiones de prácticas.

Por otra parte, dado que no hay preferencia de lenguaje a la hora de hacer el script, he decidido crearlo en Python debido a su sencillez y facilidad de comprensión.

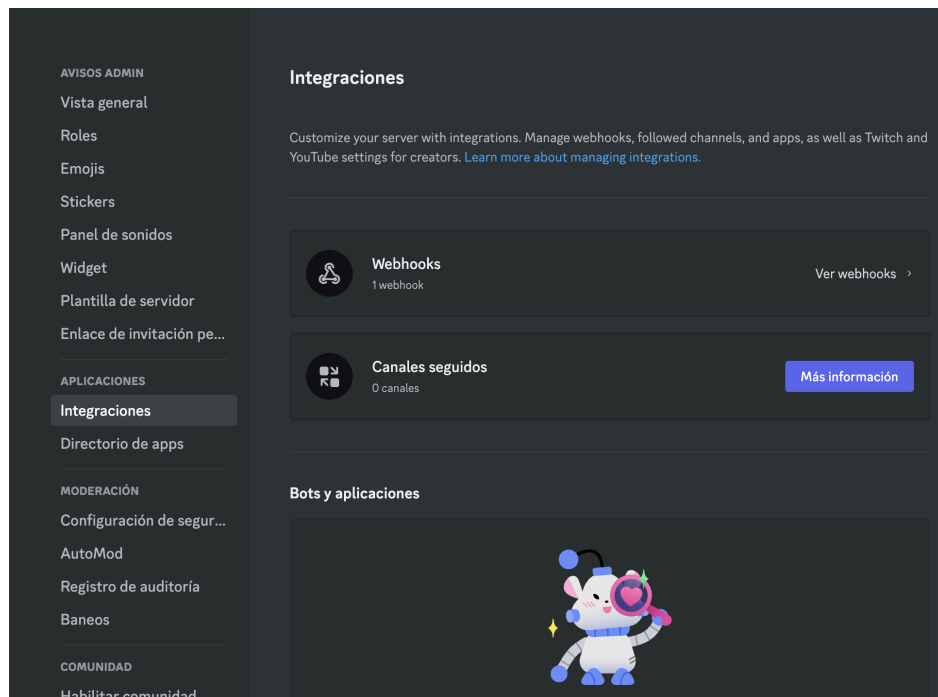
2. Script en Python

Como comento en el apartado introductorio, utilizaré el lenguaje Python para crear mi script de arranque. Para ello, estaré haciendo uso de las librerías requests y datetime para gestionar tanto las peticiones al webHook de Discord como las fechas.

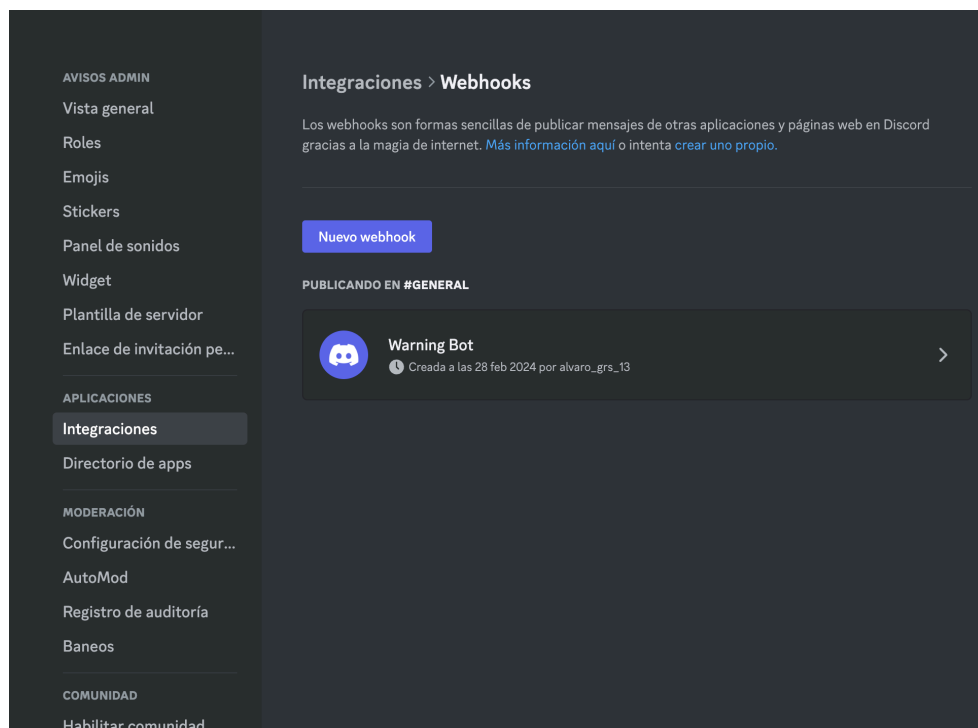
Antes que nada, hay que crear un Servidor en Discord y crear dentro de este un WebHook que será como un “**bot**” por así decirlo. Discord nos proporciona estas herramientas y nos proporciona una URL única para el WebHook que nos permitirá mandar y recibir mensajes a través de nuestro script. Los pasos para crear el webHook son los siguientes:



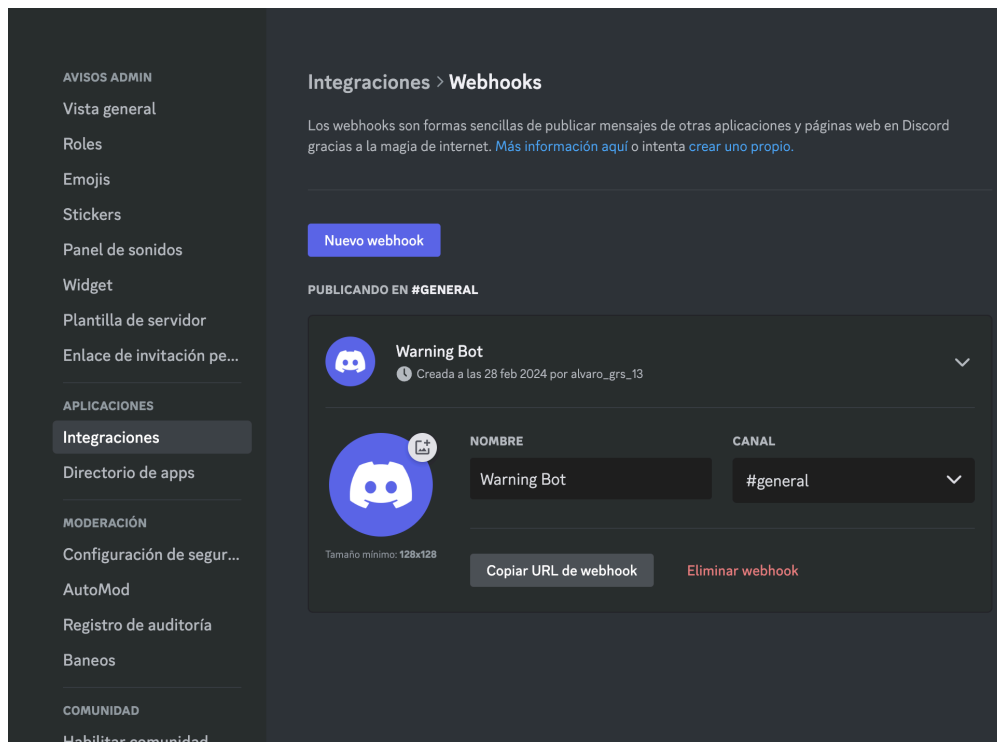
Una vez creado el Servidor, accedemos a los ajustes de este y seleccionamos la pestaña de integraciones.



Dentro de las integraciones, podemos ver la pestaña de WebHooks. Accedemos a ella.



Aquí, podemos ver nuestros WebHooks y crear nuevos. Simplemente creamos uno haciendo click en el botón de “Nuevo WebHook”. Al añadir un nuevo webhook, nos pedirá un nombre y automáticamente le asignará una URL:



Esta URL será la que estaremos utilizando en nuestro Script de Python:
<https://discord.com/api/webhooks/1212451665072889937/e6FuoDIfBFIdDRqQqcx-SscBPAOKFZg35Xv7Lw30WzZR0tiILLxoyJ9z0CIGnR0kRj4>

Una vez se ha completado este paso, vamos a crear el script:

```

AvisoDiscord.py x
Users > alvarogarcia > Desktop > AvisoDiscord.py
1 import requests
2 from datetime import datetime
3
4 # URL del webhook de Discord
5 WEBHOOK_URL = 'https://discord.com/api/webhooks/1212451665072889937/e6FuoDIfBFIdDRqQqcx-SscBPAOKFZg35Xv7Lw30WzZR0tiILLxoyJ9z0CIGnR0kRj4'
6
7 def send_message(startup=True):
8     now = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
9     status = "arrancado" if startup else "apagado"
10    message = f"El sistema operativo ha sido {status} a las {now}"
11    data = {"content": message}
12    requests.post(WEBHOOK_URL, json=data)
13
14 if __name__ == "__main__":
15     import sys
16     action = sys.argv[1] if len(sys.argv) > 1 else "startup"
17     if action == "shutdown":
18         send_message(False)
19     else:
20         send_message()
21

```

En 20 líneas completamos el script. Explicado por partes, tenemos en primer lugar, la variable WEBHOOK_URL que nos permitirá simplificar el código.

Posteriormente, creamos la función `send_message` que será la encargada de enviar el mensaje al webhook si el sistema operativo se enciende o se apaga. Para ello, utilizaremos las variables "now" con el datetime del momento de la ejecución, y status, que detectará si el ordenador se ha apagado o se ha encendido. Por otro lado tenemos el estándar de mensaje que se va a enviar, que es un estándar que se rellenará con los datos de las variables anteriormente mencionadas.

Finalmente, tenemos una variable de tipo "data" que contendrá el mensaje, y la llamada a la función `requests` que se encargará de enviar el tipo "data" al webhook de Discord.

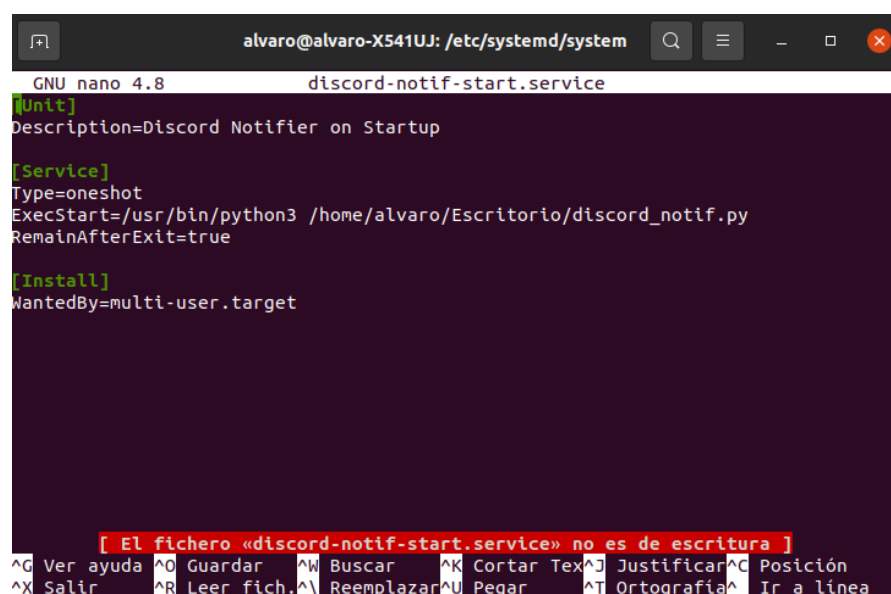
En la función principal del script (main), haremos un import de la librería `sys`, que nos permitirá conocer el estado del sistema.

Mediante la variable "action" podremos saber si el argumento startup está presente en `sys.argv`, y por tanto, action se establece en "arrancado", de lo contrario, se asume "apagado" o "shutdown" y se cambia el valor parámetro de tipo booleano de la función `send_message()` a "false".

3. Servicio de Arranque

Para crear el servicio de arranque, tenemos que entender como funciona todo el sistema de arranque del sistema operativo, en mi caso, Ubuntu emplea Systemd al igual que las máquinas debían del laboratorio de informática. Por tanto para crear el servicio de arranque, nos situamos en la ruta `/etc/systemd/system` y ejecutamos los siguientes comandos.

sudo nano discord-notif-startup.service -> Este comando sirve para crear el fichero de servicio de arranque, acto seguido creamos la unidad con el formato estándar dentro de este fichero de servicio. Escribimos:



```
alvaro@alvaro-X541UJ: /etc/systemd/system
GNU nano 4.8 discord-notif-start.service
[Unit]
Description=Discord Notifier on Startup

[Service]
Type=oneshot
ExecStart=/usr/bin/python3 /home/alvaro/Escritorio/discord_notif.py
RemainAfterExit=true

[Install]
WantedBy=multi-user.target

[ El fichero «discord-notif-start.service» no es de escritura ]
^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar Tex ^J Justificar ^C Posición
^X Salir ^R Leer fich. ^\ Reemplazar ^U Pegar ^T Ortografía ^_ Ir a línea
```

“La alerta que se muestra es porque lo abrí sin permiso para hacer la captura de pantalla”

En la unidad, recogemos una breve descripción de lo que hace (Notificar por discord al arrancar), el servicio que hace, que es ejecutar el script, en mi caso ubicado en el escritorio y, finalmente, indicar el qué modo de ejecución del sistema operativo se va a ejecutar, en este caso, en cualquier modo multi usuario.

Para terminar este proceso, habilitamos el servicio mediante **systemctl enable discord-notif-startup.service**.

```
alvaro@alvaro-X541UJ: /etc/systemd/system$ sudo systemctl enable discord-notif-startup.service
Created symlink /etc/systemd/system/multi-user.target.wants/discord-notif-startup.service → /etc/systemd/system/discord-notif-startup.service.
alvaro@alvaro-X541UJ: /etc/systemd/system$ sudo systemctl start discord-notif-startup.service
alvaro@alvaro-X541UJ: /etc/systemd/system$
```

4. Servicio de Parada

Al igual que en el servicio de arranque, es necesario crear una unidad diferente que ejecute el script en el momento en que se apague el sistema operativo. Del mismo modo nos situamos en **/etc/systemd/system** y hacemos:

sudo nano discord-notif-stop.service -> Este comando sirve para crear el fichero de servicio de parada, vamos a editarlo del mismo modo que en el caso anterior:

```
GNU nano 4.8 discord-notif-stop.service
[Unit]
Description=Discord Notifier on Shutdown
DefaultDependencies=no
Before=shutdown.target reboot.target halt.target

[Service]
Type=oneshot
ExecStart=/usr/bin/python3 /home/alvaro/Escritorio/discord_notif.py shutdown
RemainAfterExit=true

[Install]
WantedBy=halt.target reboot.target shutdown.target

12 líneas leídas
^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar Text ^J Justificar ^C Posición
^X Salir ^R Leer fich. ^\ Reemplazar ^U Pegar ^T Ortografía ^_ Ir a línea
```

En este caso, es interesante indicarle a la unidad que, evidentemente, tiene que ejecutar el script antes de que se apague por completo, es decir, antes de que se produzca un apagado, un reinicio, o un halt. También se contempla la posibilidad de que, a parte de que se ejecute el script en modo multiusuario, también lo haga para los modos 0 y 6, apagado y reboot respectivamente.

Una vez terminada y guardada la unidad, habilitamos el servicio con **systemctl enable discord-notif-stop.service** (Siempre y cuando estemos utilizando un usuario con permisos, de lo contrario usar **sudo**).

```
alvaro@alvaro-XS41UJ:/etc/systemd/system$ sudo systemctl enable discord-notif-stop.service
Created symlink /etc/systemd/system/halt.target.wants/discord-notif-stop.service → /etc/systemd/system/discord-notif-stop.service.
Created symlink /etc/systemd/system/reboot.target.wants/discord-notif-stop.service → /etc/systemd/system/discord-notif-stop.service.
Created symlink /etc/systemd/system/shutdown.target.wants/discord-notif-stop.service → /etc/systemd/system/discord-notif-stop.service.
alvaro@alvaro-XS41UJ:/etc/systemd/system$
```

5. Prueba

En la siguiente captura de pantalla, podemos observar el formato que imprime el bot de discord, notificando así la parada y arranque del sistema operativo. Existen 10 min de diferencia entre mensajes en lo que terminaba de configurar el servicio de parada. Aún así, notifica y funciona correctamente.

