# Report on Backup Script Development in Kali Linux:

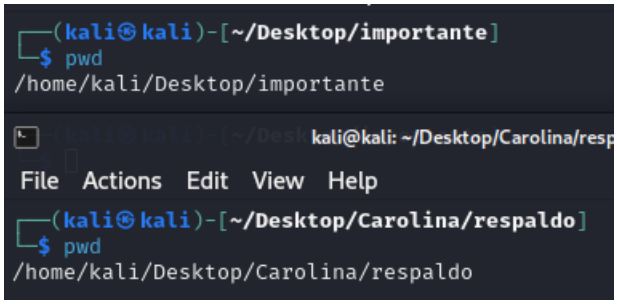
*This script automates the regular backup process for specified directories on a Linux server. It creates timestamped backup files. For educational purposes only.*

*How it works?*

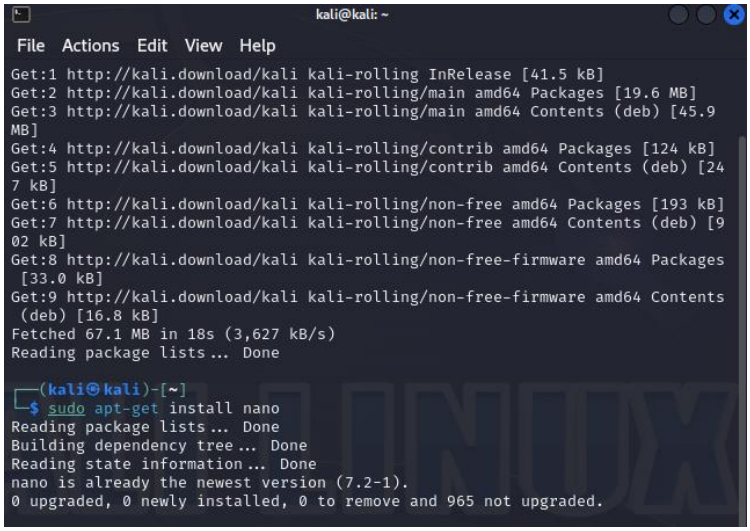
*Prepare the Environment: Ensure you have access to a Linux server or a Linux machine where you want to perform backups. Create the Script: Use a text editor such as nano or vi to create the script. You can copy and paste the script code I mentioned earlier. For example: bash Copy code nano backup.sh Then, paste the script code into the text editor and save it. Give Execute Permissions to the Script: Run the following command to give execute permissions to the script: bash Copy code chmod +x backup.sh Set the Paths: In the script, make sure to modify the origin and destination variables with the paths of the directories you want to back up and the location where you want to store the backups. Manually Run the Script: To manually run the script and perform a backup, simply execute the following command: bash Copy code ./backup.sh The script will compress the files from the source directory and create a backup in the destination directory. Schedule Automatic Execution (Optional): If you want the script to run automatically at regular intervals, you can schedule it using the cron utility. To edit the cron table, run: bash Copy code crontab -e Then, add a line like this to schedule daily execution at a specific time: bash Copy code 0 2 * * * /path/to/backup/script.sh This will run the script every day at 2:00 AM. Adjust the values to suit your specific needs. Remember to customize the paths and scheduling according to your specific requirements and setup. This script is a basic example, and you can customize it to fit your environment and needs.*

Development:

To begin, I initiated the process in Kali Linux by creating a new folder dedicated to holding the backups. Using the pwd command, I determined the source directory I intended to back up and identified the destination directory where the backups would be stored.




Subsequently, the script was crafted using the nano text editor. It's worth noting that the installation of the latest version of nano was necessary for this step.



Following the installation, the script code was copied and pasted into the text editor, and the changes were saved.

```
#!/bin/bash

# Directorio de origen que deseas respaldar
origen="/home/kali/Desktop/importante"

# Directorio de destino donde se almacenarán las copias de seguridad
destino="/home/kali/Desktop/Carolina/respaldo"

# Nombre del archivo de copia de seguridad con marca de tiempo
archivo_destino="$destino/respaldo_$(date +\%Y\%m\%d_\%H\%M\%S).tar.gz"

# Comprimir y copiar los archivos de origen al directorio de destino
tar -czvf "$archivo_destino" "$origen"

# Comprobar si la copia de seguridad se realizó con éxito
if [ $? -eq 0 ]; then
  echo "Copia de seguridad exitosa en $archivo_destino"
else
  echo "Error al realizar la copia de seguridad."
fi
```

Once the script was prepared, I executed the command chmod +x backup.sh to grant execution permissions to the script. To ensure its functionality, I manually ran the script using ./backup.sh.

```
┌──(kali㉿kali)-[~]
└─$ nano backup.sh

┌──(kali㉿kali)-[~]
└─$ chmod +x backup.sh

┌──(kali㉿kali)-[~]
└─$ ./backup.sh
tar: Removing leading `/' from member names
/home/kali/Desktop/importante/
/home/kali/Desktop/importante/apuntes
/home/kali/Desktop/importante/consola
/home/kali/Desktop/importante/hashes.txt
Copia de seguridad exitosa en /home/kali/Desktop/Carolina/respaldo/respaldo_2
0240119_095653.tar.gz

┌──(kali㉿kali)-[~]
└─$
```

As anticipated, the script efficiently compressed the files from the source directory and generated a backup in the designated destination directory.