

Proyecto Final.

Luisa Fernanda Moreno Alfonso.*

* Department of Electronic Engineering, Universidad Santo Tomás,
Bogotá, Colombia (e-mail: { Luisa.Moreno } @usantotomas.edu.co)

Abstract: This report documents the process of implementing and automating a script in a Linux environment, using GitHub for version control and Crontab for scheduled execution. Initially, a repository was set up on GitHub to store and manage the code, allowing tracking of change history. Subsequently, methods for remote execution of the script using Crontab were explored, as well as integration with monitoring and notification tools. System logs were analyzed to verify the correct execution of scheduled tasks and configuration adjustments were made to ensure proper operation. This document details the procedures followed, the challenges encountered and the solutions applied in the development of this automation.

Keywords: Automation, Linux, Bash, GitHub, Crontab, System monitoring, Version control, Postfix, MSMTP, Script execution, Scheduling, Task scheduling

1. INTRODUCTION

En el ámbito de la administración de sistemas y la automatización de tareas, el uso de scripts Bash en entornos Linux es una práctica común para gestionar recursos, ejecutar procesos repetitivos y optimizar flujos de trabajo. Para mejorar la trazabilidad y el control de versiones, herramientas como GitHub permiten almacenar y compartir código de manera eficiente, facilitando la colaboración y el seguimiento de cambios en los proyectos.

Este informe describe el desarrollo e implementación de un conjunto de scripts Bash diseñados para monitorear el estado del sistema, automatizar tareas específicas y garantizar su ejecución programada mediante Crontab. Además, se exploraron métodos de integración con servicios de notificación mediante Postfix y MSMTP, con el objetivo de generar alertas sobre el estado de ejecución de los scripts.

A lo largo del documento, se detallará el proceso de configuración del entorno, la subida del código al repositorio de GitHub, la implementación de Crontab para la ejecución automática y las estrategias utilizadas para la verificación del correcto funcionamiento del sistema. Finalmente, se analizarán los resultados obtenidos y las posibles mejoras que podrían implementarse en el futuro.

2. SCRIPT BASH

2.1 General Script Bash

Para iniciar, se desarrolló un script en Bash con el objetivo de monitorear distintos recursos del sistema. Este script ejecuta comandos del sistema para obtener información sobre:

2.2 Specific Script Bash

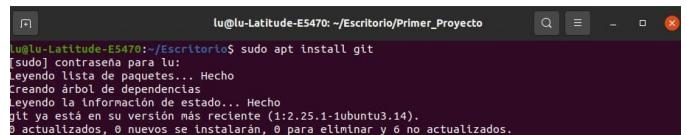
- Uso de CPU, mediante top o uptime.
- Memoria libre, consultada con free -m.
- Uso del disco, verificado con df -h.

- Estado de la batería, si es un equipo portátil, usando acpi.

Una vez escritos los comandos, se encapsularon en un archivo.sh y se le otorgaron permisos de ejecución (chmod +x script.sh), permitiendo que se ejecute fácilmente desde la terminal.

3. PROCEDIMIENTO

Instalar git hub



```
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ sudo apt install git
[sudo] contraseña para lu:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
git ya está en su versión más reciente (1:2.25.1-1ubuntu3.14).
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
```

Fig. 1. Enter Caption

- Se utilizó sudo para obtener privilegios administrativos.
- apt install git instala Git si no está presente en el sistema.

Salida del comando:

- El sistema revisó la lista de paquetes y verificó si Git estaba instalado.
- La salida indica que Git ya estaba instalado y actualizado (git ya está en su versión más reciente).

configuración de git:



```
lu@lu-Latitude-E5470:~/Escritorio$ git config --global user.name "LuMor14"
lu@lu-Latitude-E5470:~/Escritorio$ git config --global user.email "luisamorenoa@usantotomas.edu.co"
lu@lu-Latitude-E5470:~/Escritorio$ resource_monitor.sh
resource_monitor.sh: no se encontró la orden
lu@lu-Latitude-E5470:~/Escritorio$ git clone https://github.com/LuMor14/Primer_Proyecto.git
Clonando en 'Primer_Proyecto'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Desempaquetando objetos: 100% (3/3), 887 bytes | 887.00 KB/s, listo.
```

Fig. 2. Enter Caption

Se establece el nombre de usuario y el correo electrónico globalmente para que Git los use en los commits. Esto es importante para asociar correctamente los cambios con el usuario en **GitHub**.

El sistema responde con **"no se encontró la orden"**, lo que indica que el archivo no existe en el directorio actual o que no tiene permisos de ejecución.

Se clona el repositorio **Primer_Proyecto** desde GitHub al directorio local. La salida muestra que se descargaron tres objetos (archivos o commits previos) con un tamaño total de **887 bytes**.

```
lu@lu-Latitude-E5470:~/Escritorio$ ls
M_Primer_Proyecto
lu@lu-Latitude-E5470:~/Escritorio$ cd Primer_Proyecto
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ nano cpu_monitor
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ nano cpu_monitor.sh
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ ls
cpu_monitor.sh README.md
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ chmod +x cpu_monitor.sh
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ ./cpu_monitor.sh
Uso de CPU:
uso: 32,9%
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ ./cpu_monitor.sh
Uso de CPU:
uso: 28,6%
```

Fig. 3. Enter Caption

Se lista el contenido del directorio actual, mostrando que existe una carpeta llamada **Primer_Proyecto**. e accede al directorio del proyecto clonado desde GitHub.

Se intenta abrir un archivo llamado **cpu_monitor** y luego **cpu_monitor.sh** con el editor **nano**. Este archivo probablemente contenga un script en **Bash** para monitorear el uso de CPU. Se confirma que dentro del directorio están los archivos **cpu_monitor.sh** y **README.md**.

Se otorgan permisos de ejecución al archivo **cpu_monitor.sh** para que pueda ejecutarse como un programa.

Se ejecuta el script y se muestra el **uso actual de CPU** en porcentaje. Se repite la ejecución para verificar los cambios en el consumo de CPU.

```
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ nano mem_monitor.sh
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ chmod +x mem_monitor.sh
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ ./mem_monitor.sh
Uso de Memoria:
total usado libre compartido búfer/caché disponible
Memoria: 7,4Gi 4,4Gi 575Mi 882Mi 2,4Gi 1,8Gi
Swap: 2,0Gi 1,8Gi 159Mi
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ ls
cpu_monitor.sh mem_monitor.sh README.md
```

Fig. 4. Enter Caption

Se abre el editor **nano** para escribir o modificar el script **mem_monitor.sh**, que probablemente obtiene información sobre el uso de la memoria del sistema. Se le otorgan permisos de ejecución al script **mem_monitor.sh** para que pueda ejecutarse como un programa.

Se ejecuta el script y se muestra información detallada sobre la **memoria RAM** y el **swap**:

- **Total:** cantidad total de memoria disponible.
- **Usado:** memoria en uso por el sistema.
- **Libre:** memoria disponible para uso inmediato.
- **Compartido:** memoria usada por múltiples procesos.
- **Búfer/caché:** memoria usada para mejorar el/n rendimiento del sistema.
- **Disponible:** memoria realmente disponible para nuevas aplicaciones.

```
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ rm -rf cpu_monitor.sh
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ rm -rf mem_monitor.sh
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ ls
README.md
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ nano
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ nano nano system_monitor.sh
Use «fg» para volver a nano.
[1]+  Detenido nano nano system_monitor.sh
```

Fig. 5. Enter Caption

Eliminación de los scripts anteriores (**cpu_monitor.sh** y **mem_monitor.sh**) Después de eliminar los scripts, solo queda el archivo **README.md**.

el comando nano por sí solo no hace nada útil, ya que solo abre el editor sin especificar un archivo.

Aquí hay un error en el comando. Se repitió **nano** dos veces, lo que provocó que el proceso quedara **detenido** en segundo plano.

La terminal indica que el proceso está detenido, lo que significa que se debe reanudar o cerrar. Para continuar con el proceso de edición en **nano**, puedes ejecutar:

```
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ ./system_monitor.sh
== MONITOR DE RECURSOS DEL SISTEMA ==
Uso de CPU:
Uso: 31,9%
Memoria libre:
Disponible: 464Mi / Total: 7,4Gi
Uso de disco:
Usado: 70% de 61G
Estado de la batería:
state: fully-charged
percentage: 100%
No se encontró el comando 'sensors'. Instálalo con: sudo apt install lm-sensors
Top 5 procesos por uso de CPU:
PID COMMAND %CPU
112147 Discord 51.0
99615 Isolated Web Co 6.5
68128 Isolated Web Co 3.5
6077 firefox 2.5
127735 Isolated Web Co 1.6
Top 5 procesos por uso de RAM:
PID COMMAND %MEM
99615 Isolated Web Co 14.4
6077 firefox 7.1
112147 Discord 6.5
68128 Isolated Web Co 6.3
66968 Isolated Web Co 4.3
```

Fig. 6. Enter Caption

Uso de CPU: Muestra el porcentaje actual de uso.

Memoria disponible: Indica la memoria libre y la total del sistema.

Uso de disco: Muestra el porcentaje de espacio ocupado.

Estado de la batería: Indica si está cargada y el porcentaje de carga.

Procesos con mayor uso de CPU y RAM: Lista los 5 procesos más exigentes en términos de CPU y RAM.

Luego, puedes usar **sensors** para obtener temperaturas del sistema.

Discord está consumiendo mucha CPU (51%) Si el uso es constante, podrías cerrarlo o reiniciarlo para optimizar el rendimiento.

Firefox y **procesos Isolated Web Co** consumen RAM y CPU Estos procesos corresponden a pestañas o servicios de Firefox. Si necesitas liberar recursos, considera cerrarlas.

```

lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ crontab -e
no crontab for lu - using an empty one

Select an editor. To change later, run 'select-editor'.
1. /bin/nano      <---- easiest
2. /usr/bin/vim.tiny
3. /usr/bin/code
4. /bin/ed

choose 1-4 [1]: 1
crontab: installing new crontab
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ ls
README.md  system_monitor.sh
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ realpath system_monitor.sh
/home/lu/Escritorio/Primer_Proyecto/system_monitor.sh
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ crontab -e
crontab: installing new crontab

```

Fig. 7. Enter Caption

Estás intentando automatizar la ejecución del script `system_monitor.sh` usando `crontab`.

Usaste `realpath system_monitor.sh` para obtener la ruta absoluta del script. Luego abriste `crontab -e` para programar la tarea.

Dentro del archivo de `crontab`, agrega una línea para ejecutar el script automáticamente en un intervalo de tiempo específico. `*/5 * * * * /home/lu/Escritorio/Primer_Proyecto/system_monitor.sh hmod +x /home/lu/Escritorio/Primer_Proyecto/system_monitor.sh`

Luego revisa los logs de `cron` para ver si hay errores:

```

lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ crontab -l
# Edit this file to introduce tasks to be run by cron.

# Cada tarea debe definirse en una sola linea con el formato:
# m h dom mon dow command
# Donde:
#   - m = minuto (0 - 59)
#   - h = hora (0 - 23)
#   - dom = dia del mes (1 - 31)
#   - mon = mes (1 - 12)
#   - dow = dia de la semana (0 - 7, donde 0 y 7 son domingo)
#   - command = comando a ejecutar
#
# -----#
# MONITOREO AUTOMÁTICO DEL SISTEMA
# -----#
# Ejecutar el script cada 5 minutos y guardar el log
*/5 * * * * /home/lu_usuario/Escritorio/system_monitor.sh >> /home/lu_usuario/Escritorio/logs_monitor.txt 2>&1

# Hacer una copia de seguridad de /home cada dia a las 3 AM
0 3 * * * tar -czf /home/lu_usuario/backups/home_backup_$(date +\%F).tar.gz /home/lu_usuario

# Limpiar archivos temporales cada domingo a la medianoche
0 0 * * 0 rm -rf /home/lu_usuario/.cache/*

# Enviar un mensaje de prueba cada dia a las 8 AM
0 8 * * * echo "El sistema sigue funcionando correctamente" | mail -s "Reporte Diario" tu_email@example.com

```

Fig. 8. Enter Caption

Captura el estado del sistema periódicamente y almacena los logs. —2;1— redirige errores al mismo archivo para una mejor depuración.

Crea una copia comprimida con la fecha en el nombre del archivo.

`textbf{Verifica que la carpeta /home/tu_usuario/backups/ exista, o tar fallará.}`

Limpia archivos temporales para liberar espacio.

Necesita que `mailutils` esté instalado (`sudo apt install mailutils`). Verifica la configuración de correo si no recibes el mensaje.

Si aparece como "active (running)", todo está bien.

`grep CRON /var/log/syslog — tail -10`

Deberías ver líneas indicando que `cron` ejecutó `system_monitor.sh`. `cat /home/tu_usuario/Escritorio/logs_monitor.txt`

Si el archivo no existe o está vacío, puede haber un problema con los permisos del script.

```

lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ sudo systemctl start cron
[sudo] contraseña para lu:
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ crontab -e
Crontab: installing new crontab
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ crontab -l
# Edit this file to introduce tasks to be run by cron.

# Cada tarea debe definirse en una sola linea con el formato:
# m h dom mon dow command
# Donde:
#   - m = minuto (0 - 59)
#   - h = hora (0 - 23)
#   - dom = dia del mes (1 - 31)
#   - mon = mes (1 - 12)
#   - dow = dia de la semana (0 - 7, donde 0 y 7 son domingo)
#   - command = comando a ejecutar
#
# -----#
# MONITOREO AUTOMÁTICO DEL SISTEMA
# -----#
# Ejecutar el script cada 5 minutos y guardar el log
*/5 * * * * /home/lu_usuario/Escritorio/system_monitor.sh >> /home/lu_usuario/Escritorio/logs_monitor.txt 2>&1

```

Fig. 9. Enter Caption

`/home/tu_usuario/Escritorio/system_monitor.sh`

Si el archivo no existe o está vacío, puede haber un problema con los permisos del script.

`/home/tu_usuario/Escritorio/system_monitor.sh`

```

crontab: installing new crontab
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ crontab -l
# Edit this file to introduce tasks to be run by cron.

# Cada tarea debe definirse en una sola linea con el formato:
# m h dom mon dow command
# Donde:
#   - m = minuto (0 - 59)
#   - h = hora (0 - 23)
#   - dom = dia del mes (1 - 31)
#   - mon = mes (1 - 12)
#   - dow = dia de la semana (0 - 7, donde 0 y 7 son domingo)
#   - command = comando a ejecutar
#
# -----#
# MONITOREO AUTOMÁTICO DEL SISTEMA
# -----#
# Ejecutar el script cada 5 minutos y guardar el log
*/5 * * * * /home/lu_usuario/Escritorio/system_monitor.sh >> /home/lu_usuario/Escritorio/logs_monitor.txt 2>&1

# Hacer una copia de seguridad de /home cada dia a las 3 AM
0 3 * * * tar -czf /home/lu_usuario/backups/home_backup_$(date +\%F).tar.gz /home/lu_usuario

# Limpiar archivos temporales cada domingo a la medianoche
0 0 * * 0 rm -rf /home/lu_usuario/.cache/*

# Enviar un mensaje de prueba cada dia a las 8 AM
0 8 * * * echo "El sistema sigue funcionando correctamente" | mail -s "Reporte Diario" luisamoreno@uam.mx

lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ sudo systemctl start cron
sudo: system: orden no encontrada
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ sudo systemctl start cron
sudo: system: orden no encontrada

```

Fig. 10. Enter Caption

Para iniciar el servicio `cron`, usa este comando:

`sudo systemctl start cron`

Siquieres asegurarte de que se inicie automáticamente con el sistema, ejecuta:

`sudo systemctl enable cron`

Para verificar que `cron` está activo, usa:

```

lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ sudo systemctl status cron
sudo: system: orden no encontrada
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ sudo systemctl status cron
sudo: system: orden no encontrada
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ systemctl status cron
● cron.service - Regular background program processing daemon
   Loaded: loaded (/lib/systemd/system/cron.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2025-03-12 03:30:51 -05; 4 days ago
     Docs: man:cron(8)
       Main PID: 698 (cron)
          Tasks: 1 (limit: 9014)
            CPU: 2.4H
           Memory: 2.4M
          CGroup: /system.slice/cron.service
                  └─ 698 /usr/sbin/cron -f

mar 16 16:45:01 lu-Latitude-E5470 CRON[133046]: pam_unix(cron:session): session closed for user lu
mar 16 16:50:01 lu-Latitude-E5470 CRON[133146]: pam_unix(cron:session): session opened for user lu by (uid=0)
mar 16 16:50:01 lu-Latitude-E5470 CRON[133147]: (lu) CMD (/home/lu_usuario/Escritorio/system_monitor.sh >> /home/lu_usuario/Escritorio/logs)
mar 16 16:50:01 lu-Latitude-E5470 CRON[133146]: (CRON) info (No MTA installed, discarding output)
mar 16 16:50:01 lu-Latitude-E5470 CRON[133146]: pam_unix(cron:session): session closed for user lu
mar 16 16:55:01 lu-Latitude-E5470 cron[698]: (lu) RELOAD (crontabs/lu)
mar 16 16:55:01 lu-Latitude-E5470 CRON[133328]: pam_unix(cron:session): session opened for user lu by (uid=0)
mar 16 16:55:01 lu-Latitude-E5470 CRON[133329]: (lu) CMD (/home/lu_usuario/Escritorio/system_monitor.sh >> /home/lu_usuario/Escritorio/logs)
mar 16 16:55:01 lu-Latitude-E5470 CRON[133328]: (CRON) info (No MTA installed, discarding output)
mar 16 16:55:01 lu-Latitude-E5470 CRON[133328]: pam_unix(cron:session): session closed for user lu

```

Fig. 11. Enter Caption

Esto significa que cron está intentando enviar correos con la salida de los comandos, pero no hay un servidor de correo (MTA) instalado.

(CRON) info (No MTA installed, discarding output) Esto significa que cron intentó enviar un correo con la salida del comando, pero no hay un servidor de correo (MTA) instalado.

```
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ realpath ~/Escritorio/Primer_Proyecto/system_monitor.sh
/home/lu/Escritorio/Primer_Proyecto/system_monitor.sh
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ git init
Reinicilizado el repositorio Git existente en /home/lu/Escritorio/Primer_Proyecto/.git/
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ git remote add origin https://github.com/LuMor14/Primer_Proyecto.git
remote: fatal: remote origin ya existe.
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ git clone https://github.com/LuMor14/Primer_Proyecto.git
Clonando en 'Primer_Proyecto'...
remote: Enumerando objetos: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Desempaquetando objetos: 100% (3/3), 887 bytes | 443.00 KB/s, listo.
```

Fig. 12. Enter Caption

git status Verifica el estado de los archivos git add . Agrega todos los archivos al commit git commit -m "Primer commit" Crea un commit con un mensaje git push origin main Sube los archivos al repositorio

rm -rf PrimerProyecto Borra la carpeta clonada (¡Asegúrate de no perder archivos importantes!)

```
git clone https://github.com/LuMor14/PrimerProyecto.git
cd PrimerProyecto
```

```
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ git add system_monitor.sh
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ git commit -m "subiendo script"
[main f612992] subiendo script
 1 file changed, 38 insertions(+)
 create mode 100755 system_monitor.sh
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ git push origin main
Username for 'https://github.com': LuMor14
Password for 'https://LuMor14@github.com':
remote: Support for password authentication was removed on August 13, 2021.
remote: Please see https://docs.github.com/get-started/getting-started-with-git/about-remote-repositories#cloning-with-https-urls for information on currently recommended modes of authentication.
fatal: Autenticación falló para 'https://github.com/LuMor14/Primer_Proyecto.git'
```

Fig. 13. Enter Caption

Genera un Token de Acceso Personal en GitHub:

- Ve a <https://github.com/settings/tokensGitHub> → Configuración de Tokens
- Crea un nuevo token con los siguientes permisos: repo (acceso a repositorios)
- Copia el token generado (se mostrará solo una vez).

Usa el Token en lugar de tu contraseña

Cuando ejecutes: bashCopiarEditargit push origin main En vez de la contraseña, pega tu token.

Solución permanente con autenticación SSH (recomendada)

Si no quieres ingresar el token cada vez, puedes configurar SSH para autenticarte automáticamente:

Genera una clave SSH (si no tienes una)

```
bashCopiarEditarssh-keygen -t rsa -b 4096 -C "tuemail@example.com" Pre-
siona Enter varias veces (sin contraseña opcional).
```

Copia la clave pública

```
bashCopiarEditarcat ~/.ssh/id_rsa.pub Copia el con-
tenido.
```

Agrega la clave a GitHub

```
lu@lu-Latitude-E5470:~/Escritorio/Primer_Proyecto$ git push --set-upstream origin main
Username for 'https://github.com': LuMor14
Password for 'https://LuMor14@github.com':
Enumerando objetos: 4, listo.
Contando objetos: 100% (4/4), listo.
Comprimiendo objetos: 100% (3/3), listo.
Escribiendo objetos: 100% (3/3), 816 bytes | 816.00 KiB/s, listo.
Total 3 (delta 0), reusado 0 (delta 0)
To https://github.com/LuMor14/Primer_Proyecto.git
 40a9fe7f612992 main -> main, origin/main, origin/HEAD
Author: LuMor14 <luisamorenoa@usantotomas.edu.co>
Date:  Sun Mar 16 18:21:38 2025 -0500

    subiendo scrip

commit 40a9fe773b6c4e1b4f82fc8e152d4759ac69d3ec
Author: LuMor14 <luisamorenoa@usantotomas.edu.co>
Date:  Sun Mar 16 12:44:16 2025 -0500
```

Fig. 14. Enter Caption

- Ve a <https://github.com/settings/keysGitHub> → Configuración SSH
- Agrega una nueva clave SSH y pega el contenido.

Prueba la conexión

```
bashCopiarEditarssh -T git@github.com Si ves Hi <tu_usuario> ¡ya está listo!
```

Cambia la URL remota de tu repo a SSH

```
bashCopiarEditargit remote set-url origin git@github.com:
```

Para comprobar que la automatización funcionaba, se revisaron los logs del sistema (journalctl -u cron -since "5 minutes ago") y se verificó que el script se ejecutara correctamente.

Ahora el código ya está subido a GitHub correctamente. La rama **main** está sincronizada con **origin/main**, y el historial de commits se muestra correctamente con **git log**