TRABAJO FINAL

TECNICATURA SUPERIOR EN DESARROLLO DE SOFTWARE
SISTEMAS OPERATIVOS

Automatización de Tareas del Sistema Operativo

Autor/es: -Emmanuel Espinosa
- Lorenzo Colombo
- Yamila Etchepareborda
- Iñaki Velo

Docente/s: Lucas Martín Treser

Lugar y Fecha: Mar del Plata, Octubre de 2024

Introducción

En el ámbito de los sistemas operativos, la automatización de tareas de mantenimiento y gestión es fundamental para optimizar el rendimiento y garantizar la estabilidad del sistema. Estas tareas incluyen desde la creación de usuarios con permisos adecuados hasta la instalación de actualizaciones y la liberación de espacio en disco mediante la eliminación de archivos temporales. Con el objetivo de aplicar y consolidar los conocimientos adquiridos en la asignatura de Sistemas Operativos, se desarrolló un script en Bash para automatizar tres de estas tareas críticas.

Objetivo del Proyecto

El proyecto busca simplificar y agilizar procesos de administración de sistemas, reduciendo la intervención manual necesaria para tareas de mantenimiento. Para lograrlo, se eligieron las siguientes tareas de entre una lista de opciones propuestas:

Creación y gestión de usuarios en el sistema: Esta tarea permite agregar nuevos usuarios al sistema y gestionar los permisos y directorios personales de cada usuario. De esta forma, se facilita la administración de usuarios, asegurando que cada cuenta esté configurada con los permisos adecuados para preservar la seguridad y la privacidad de los datos.

Actualización automática del sistema: Mantener un sistema operativo actualizado es esencial para garantizar la seguridad y funcionalidad de sus componentes. Esta tarea permite verificar y ejecutar actualizaciones del sistema de manera automática, con un registro de cada cambio realizado. Esta funcionalidad garantiza que el sistema permanezca seguro y optimizado sin la necesidad de supervisión constante.

Mostrar el uso de memoria y crear un reporte: El objetivo principal de este script es monitorear el uso de memoria y espacio en disco del sistema, proporcionando un reporte detallado que permita evaluar el estado actual de estos recursos. Al registrar y presentar esta información en un archivo de reporte, el script ayuda a los administradores a identificar posibles problemas de rendimiento y a realizar un seguimiento continuo de los recursos. Con esto, se busca facilitar la toma de decisiones para el mantenimiento y la optimización del sistema, asegurando su estabilidad y eficiencia operativa.

Resultados Esperados

Al implementar este script, se espera mejorar la eficiencia y seguridad del sistema, reduciendo la necesidad de intervención manual y brindando al administrador una herramienta práctica para el mantenimiento periódico. La documentación técnica

que acompaña este proyecto detalla el código, los procedimientos de prueba y los resultados obtenidos, proporcionando una guía completa sobre la implementación y uso de cada funcionalidad desarrollada.

Requisitos Técnicos

Para ejecutar y aprovechar las funcionalidades de los scripts de este proyecto, es necesario cumplir con los siguientes requisitos técnicos:

Sistema Operativo:

El proyecto está diseñado para sistemas operativos basados en Linux, preferentemente distribuciones como Ubuntu o Debian.

Permisos de Usuario:

Los scripts requieren permisos de superusuario (root) o privilegios de sudo para ejecutar tareas administrativas como la actualización del sistema, creación de usuarios y eliminación de archivos en directorios protegidos. Esto garantiza que las acciones de administración puedan realizarse sin restricciones y con los niveles de seguridad adecuados.

Shell de Bash:

Es necesario contar con Bash instalado como intérprete de comandos, ya que los scripts están escritos en Bash. La mayoría de las distribuciones Linux vienen con Bash preinstalado, pero se puede verificar su instalación ejecutando bash --version.

Herramientas y Comandos de Sistema:

El script de limpieza depende de varios comandos de sistema para recolectar información sobre el rendimiento, como:

- date para obtener la fecha y hora actual.
- free para ver el uso de memoria.
- top o ps para monitorear el uso de CPU.
- df para conocer el espacio en disco.

Estos comandos vienen preinstalados en la mayoría de las distribuciones Linux, pero es recomendable verificar su disponibilidad para garantizar el correcto funcionamiento del script.

Permisos de Escritura en Directorios Específicos:

Los scripts generan archivos de registro, como registro.log y reporte.log, para registrar las acciones y los datos recopilados. Por lo tanto, es necesario que el usuario tenga permisos de escritura en el directorio en el que se ejecutan los scripts o en el directorio de destino de los archivos de log.

Cumplir con estos requisitos técnicos asegura que los scripts se ejecuten de manera correcta y que el sistema esté preparado para soportar las automatizaciones de administración planteadas en este proyecto.

Desarrollo

El proyecto consistió en la creación de scripts en Bash para automatizar tareas de administración del sistema, abarcando la gestión de usuarios, la actualización automática del sistema y la limpieza de archivos temporales. Para la gestión de usuarios, desarrollamos un script que permite crear usuarios, asignar permisos y establecer directorios personales, asegurando un entorno seguro para cada usuario. En la actualización del sistema, el script verifica permisos y aplica actualizaciones de manera automática, registrando cada cambio en un archivo de log para facilitar el seguimiento. Finalmente, el script de limpieza elimina archivos temporales y genera un reporte sobre el uso de recursos del sistema, contribuyendo a optimizar el rendimiento y liberar espacio en disco. Todos los scripts fueron diseñados con estructuras de control y manejo de errores, logrando una solución robusta y eficiente para el mantenimiento del sistema.

Menú

Este script presenta un menú interactivo que facilita la ejecución de diferentes tareas de administración del sistema. A continuación, se detallan sus funcionalidades:

-Definición de colores: Para mejorar la experiencia visual, se definen variables de color (CYAN, YELLOW, RED, NC) que se utilizan en los mensajes mostrados al usuario. Esto ayuda a destacar las opciones y mensajes importantes.

-Estructura del menú: El script crea un bucle while true que presenta continuamente el menú hasta que el usuario elija salir. Las opciones disponibles son:

- Crear un nuevo usuario (ingresarUsuario.sh).
- Verificar e instalar actualizaciones (verifActualiz.sh).
- Mostrar el uso de memoria y crear un reporte (mostrarUso.sh).

- Salir del menú.

-Ejecución de comandos según la opción seleccionada: Utiliza una estructura case para determinar la acción a realizar según la opción ingresada por el usuario.

- Cada opción se asocia a la ejecución de un script específico, permitiendo automatizar tareas comunes sin necesidad de repetir comandos manualmente.
- En caso de una entrada inválida, se muestra un mensaje en rojo que indica el error y permite al usuario intentar de nuevo.

Este menú interactivo ofrece una forma sencilla y clara de gestionar diferentes scripts administrativos, mejorando la eficiencia y la organización del sistema.

```bash
#!/bin/bash

# Colores para mejorar la experiencia del usuario
CYAN='\033[0;36m'
YELLOW='\033[1;33m'
RED='\033[0;31m'
NC='\033[0m'  # Sin color

while true; do
    echo -e "${CYAN}=== MENÚ INTERACTIVO ===${NC}"
    echo -e "${YELLOW}1) Crear un nuevo usuario${NC}"
    echo -e "${YELLOW}2) Verificar e instalar actualizaciones${NC}"
    echo -e "${YELLOW}3) Mostrar el uso de memoria y crear un reporte${NC}"
    echo -e "${YELLOW}4) Salir${NC}"

    read -p "Selecciona una opción: " opcion

    case $opcion in
        1) ./ingresarUsuario.sh ;;  # Ejecuta el script para crear un usuario
        2) ./verifActualiz.sh ;;  # Ejecuta el script de actualización
        3) ./mostrarUso.sh ;;  # Mostrar el uso de memoria y crear un reporte
        4) echo "Saliendo..."; exit 0 ;;
        *) echo -e "${RED}Opción inválida. Intenta de nuevo.${NC}" ;;
    esac
done
```

Creación y gestión de usuarios

Este script tiene como objetivo crear nuevos usuarios o modificar usuarios existentes en el sistema. A continuación, se explican sus componentes y funcionamiento:

-Solicitud de nombre de usuario: El script inicia pidiendo al usuario que ingrese el nombre del usuario que desea crear o modificar. Esto se logra mediante el comando read, que almacena la entrada en la variable $usuario.

-Verificación de existencia del usuario: Utiliza el comando id para comprobar si el usuario ya existe en el sistema. Redirige la salida a /dev/null para evitar que se muestren mensajes adicionales en la consola. Si el usuario existe, se informa al usuario con un mensaje adecuado.

-Opción para modificar el usuario existente: Si el usuario ya está registrado, el script pregunta al usuario si desea modificar la cuenta existente.

- Si la respuesta es s o S, se ejecutan comandos que permiten modificar al usuario (como cambiar la contraseña mediante sudo passwd $usuario).
- Si la respuesta es negativa, el script termina sin realizar modificaciones.

-Creación de un nuevo usuario: Si el usuario no existe en el sistema, se procede a crearlo utilizando el comando adduser. Una vez completado el proceso, se muestra un mensaje indicando que la cuenta se ha creado exitosamente.

Este diseño asegura que los usuarios puedan gestionar cuentas de manera eficiente, evitando duplicados y permitiendo modificaciones sencillas.

```bash
#!/bin/bash

echo "Ingrese el nombre del nuevo usuario: "
read usuario

# Verificar si el usuario ya existe
if id "$usuario" &>/dev/null; then
    echo "El usuario '$usuario' ya existe."
    echo "¿Deseas modificarlo? (s/n)"
    read respuesta

    if [ "$respuesta" == "s" ] || [ "$respuesta" == "S" ]; then
        echo "Modificando el usuario '$usuario'."
        # Aquí puedes agregar los comandos para modificar el usuario
        # Por ejemplo, cambiar la contraseña o los permisos
        sudo passwd "$usuario"
        echo "Usuario '$usuario' modificado con éxito."
    else
        echo "No se realizaron cambios."
    fi
else
    sudo adduser "$usuario"
    echo "Usuario '$usuario' creado con éxito."
fi
```

Actualización del sistema

Este script tiene la función de verificar e instalar actualizaciones en el sistema de forma automática, registrando cada acción en un archivo de log. En la primera línea, se define la función actualizar_sistema(), que contiene la lógica de actualización.

-Verificación de permisos: Primero, el script comprueba si el usuario tiene permisos de superusuario (UID=0) o pertenece al grupo sudo. Si el usuario cumple con alguna de estas condiciones, el script procede; de lo contrario, muestra un mensaje de error y termina la ejecución.

-Actualización del sistema: Dependiendo de los permisos, el script ejecuta los comandos apt update y apt upgrade -y, que verifican e instalan las actualizaciones disponibles. Este proceso garantiza que el sistema se mantenga actualizado con los últimos parches de seguridad y mejoras.

-Registro de cambios: Una vez realizada la actualización, el script captura la fecha y hora actuales mediante el comando date y guarda esta información junto con el mensaje de actualización en el archivo registro.log. Esto permite llevar un historial de las actualizaciones aplicadas en el sistema para futuras auditorías.

Finalmente, la función actualizar_sistema se llama en la última línea del script, ejecutando todo el proceso descrito cuando el script se inicia.

```bash
#!/bin/bash

actualizar_sistema() {
    # Verificamos que el usuario es root, es necesario para actualizar el sistema
    if [ "$UID" -eq 0 ]; then
        echo "Actualizando como root"
        apt update && apt upgrade -y

        # Registramos los cambios en un archivo log con fecha y hora
        fecha_hora=$(date +"%Y-%m-%d_%H:%M:%S")
        echo "Se realizaron actualizaciones del sistema a las $fecha_hora" >> registro.log

    elif groups "$USER" | grep -q "\<sudo\>"; then
        echo "Actualizando como sudo"
        sudo apt update && sudo apt upgrade -y

        # Registramos los cambios en un archivo log con fecha y hora
        fecha_hora=$(date +"%Y-%m-%d_%H:%M:%S")
        echo "Se realizaron actualizaciones del sistema a las $fecha_hora" >> registro.log

    else
        echo "Este script debe ser ejecutado como root o con sudo."
        exit 1
    fi
}

# Llama a la función
actualizar_sistema
```

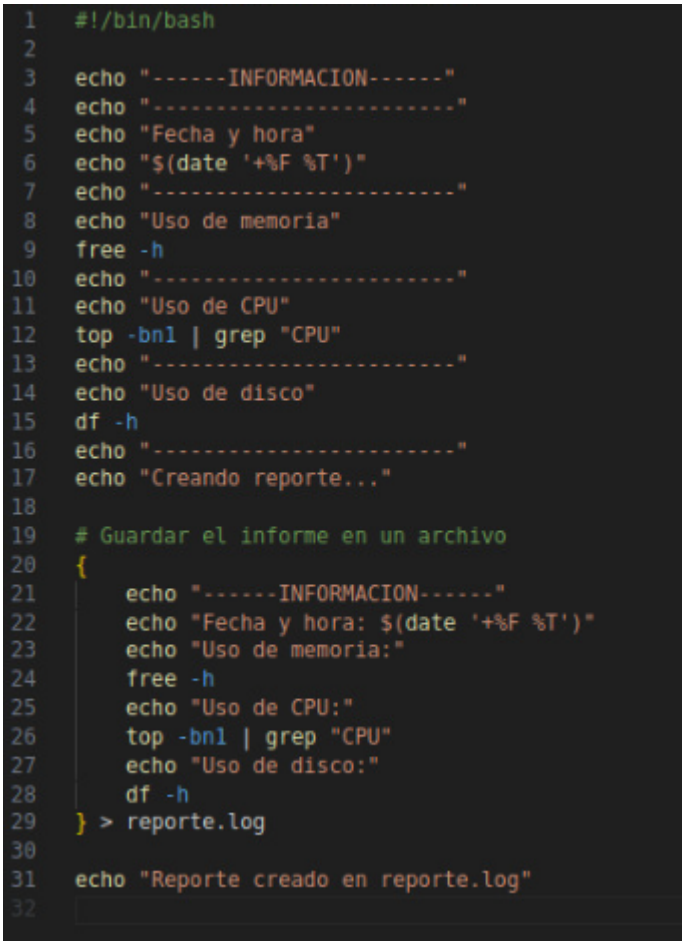
Mostrar el uso de memoria y crear un reporte

El tercer script tiene como objetivo mostrar el uso de memoria y crear un reporte detallado del mismo.

Información del sistema: En las primeras líneas, el script muestra información relevante como la fecha y hora, el uso de memoria, el uso de CPU y el espacio en disco. Para cada métrica, se utilizan comandos como date, free -h, top, y df -h, que permiten obtener detalles sobre el rendimiento actual del sistema.

Creación de reporte: Tras mostrar la información en pantalla, el script guarda esta misma información en un archivo llamado reporte.log. Esto se realiza mediante una función que utiliza echo para escribir cada dato en el archivo, creando un reporte detallado que se puede revisar más tarde.

Mensaje de finalización: Finalmente, el script muestra un mensaje de confirmación que indica que el reporte se ha creado exitosamente en reporte.log.



```bash
#!/bin/bash

echo "------INFORMACION------"
echo "------------------------"
echo "Fecha y hora"
echo "$(date '+%F %T')"
echo "------------------------"
echo "Uso de memoria"
free -h
echo "------------------------"
echo "Uso de CPU"
top -bn1 | grep "CPU"
echo "------------------------"
echo "Uso de disco"
df -h
echo "------------------------"
echo "Creando reporte..."

# Guardar el informe en un archivo
{
    echo "------INFORMACION------"
    echo "Fecha y hora: $(date '+%F %T')"
    echo "Uso de memoria:"
    free -h
    echo "Uso de CPU:"
    top -bn1 | grep "CPU"
    echo "Uso de disco:"
    df -h
} > reporte.log

echo "Reporte creado en reporte.log"
```

Pruebas y Validación

Para asegurar el correcto funcionamiento de los scripts desarrollados, realizamos una serie de pruebas exhaustivas que abarcan distintos aspectos de su operación. En primer lugar, verificamos que cada script ejecutara las tareas para las cuales fue diseñado, probando casos comunes y escenarios límite.

Pruebas de creación y gestión de usuarios: Probamos el script de creación de usuarios en múltiples escenarios, incluyendo la creación de usuarios nuevos, asignación de permisos y configuración de directorios personales. Verificamos que cada usuario creado tuviera acceso adecuado y que los permisos otorgados fueran los correctos. Además, evaluamos la respuesta del script en casos de datos incompletos o permisos insuficientes.

Pruebas de actualización automática: Para validar el script de actualizaciones, ejecutamos pruebas tanto en entornos controlados como en configuraciones de sistema más amplias. Observamos que el script verifica correctamente los permisos de usuario antes de ejecutar las actualizaciones

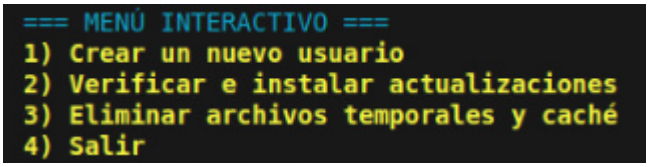
y que registra cada actualización realizada con precisión en el archivo de log, facilitando así el seguimiento de los cambios realizados.

Pruebas del Script de Monitoreo del Sistema:Se realizaron pruebas para validar que el script ejecutara correctamente la eliminación de archivos temporales y generara un reporte detallado del estado del sistema. Las pruebas incluyeron la verificación de la precisión en la información sobre el uso de memoria y espacio en disco. Los resultados confirmaron que el sistema se optimizaba y liberaba espacio tras la ejecución, cumpliendo así con los objetivos de monitoreo y mantenimiento del sistema.

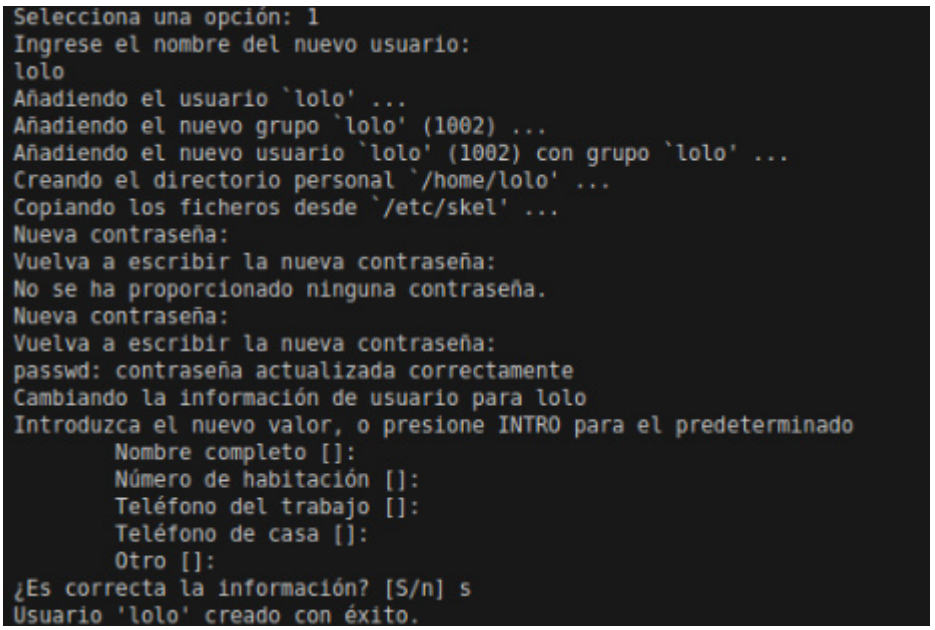
En cada una de estas pruebas, también se realizaron validaciones para asegurar que el sistema continuara funcionando normalmente tras la ejecución de los scripts y que no hubiera efectos secundarios no deseados. Como resultado, pudimos confirmar que las herramientas desarrolladas cumplen con sus objetivos y pueden integrarse en el sistema sin generar riesgos adicionales. Las pruebas nos permitieron afinar detalles y mejorar la confiabilidad de los scripts, garantizando su utilidad en un entorno real de administración del sistema.

Devoluciones recibidas

Menú:



Creación y gestión de usuarios en el sistema:



Actualización automática del sistema:

```
Selecciona una opción: 2
Actualizando como sudo
Obj:1 https://packages.microsoft.com/repos/code stable InRelease
Ign:2 http://packages.linuxmint.com virginia InRelease
Obj:3 http://archive.ubuntu.com/ubuntu jammy InRelease
Des:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Obj:5 http://packages.linuxmint.com virginia Release
Des:6 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
Des:8 http://security.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11 Metadata [43,2 kB]
Des:9 http://archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]
Des:10 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 DEP-11 Metadata [208 B]
Des:11 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 DEP-11 Metadata [126 kB]
Des:12 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 DEP-11 Metadata [208 B]
Des:13 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 DEP-11 Metadata [103 kB]
Des:14 http://archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 DEP-11 Metadata [212 B]
Des:15 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 DEP-11 Metadata [356 kB]
Des:16 http://archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 DEP-11 Metadata [940 B]
Des:17 http://archive.ubuntu.com/ubuntu jammy-backports/main amd64 DEP-11 Metadata [5.316 B]
Des:18 http://archive.ubuntu.com/ubuntu jammy-backports/restricted amd64 DEP-11 Metadata [212 B]
Des:19 http://archive.ubuntu.com/ubuntu jammy-backports/universe amd64 DEP-11 Metadata [23,1 kB]
Des:20 http://archive.ubuntu.com/ubuntu jammy-backports/multiverse amd64 DEP-11 Metadata [212 B]
Descargados 1.043 kB en 3s (306 kB/s)
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Todos los paquetes están actualizados.
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
```

Mostrar el uso de memoria y crear un reporte:

```
Selecciona una opción: 3
------INFORMACION------
------------------------
Fecha y hora
2024-10-28 07:38:55
------------------------
Uso de memoria
               total       usado       libre  compartido   búf/caché  disponible
Mem:           1,9Gi       1,3Gi        75Mi        60Mi       540Mi       406Mi
Inter:         1,1Gi       916Mi       246Mi
------------------------
Uso de CPU
%Cpu(s): 25,7 us,  5,7 sy,  0,0 ni, 65,7 id,  0,0 wa,  0,0 hi,  2,9 si,  0,0 st
------------------------
Uso de disco
S.ficheros     Tamaño Usados  Disp Uso% Montado en
tmpfs            197M   1,2M  196M   1% /run
/dev/sda3         24G    12G   11G  53% /
tmpfs            982M    14M  969M   2% /dev/shm
tmpfs            5,0M   4,0K  5,0M   1% /run/lock
/dev/sda2        512M   6,1M  506M   2% /boot/efi
tmpfs            197M   100K  197M   1% /run/user/1000
------------------------
Creando reporte...
Reporte creado en reporte.log
```

Conclusión

En conclusión, este proyecto nos permitió desarrollar una solución automatizada para facilitar tareas de administración en un sistema informático, simplificando procesos que de otro modo requerirían intervención manual. A lo largo de su implementación, logramos configurar y ejecutar scripts que permiten gestionar usuarios, mantener el sistema actualizado y optimizar el uso del espacio de almacenamiento. Gracias a este enfoque, no solo se mejora la eficiencia en la administración, sino que también se asegura que el sistema se mantenga ordenado y seguro, con menos posibilidades de errores o descuidos humanos. Este trabajo también nos brindó la oportunidad de adquirir habilidades prácticas en el uso de herramientas de automatización, lo cual es fundamental en el ámbito de la administración de sistemas. En resumen, el proyecto aporta una solución que hace la gestión del sistema más rápida, precisa y confiable, beneficiando tanto a los usuarios como a los administradores en su experiencia y manejo diario del sistema.