

# Manual Maestro de Linux: Guía Completa de Terminal y Scripting

Documentación Técnica para Administradores de Sistemas

Febrero 2026

## Índice

<b>I</b>	<b>Linux comandos básicos</b>	<b>2</b>
I.1	ls - Listar archivos . . . . .	2
I.2	cd - Cambiar de directorio . . . . .	2
I.3	pwd - Directorio actual . . . . .	2
I.4	mkdir - Crear carpetas . . . . .	2
I.5	touch - Crear archivos vacíos . . . . .	2
I.6	cp - Copiar archivos y carpetas . . . . .	2
I.7	rm - Eliminar . . . . .	2
I.8	cat - Visualizar contenido . . . . .	3
<b>II</b>	<b>Linux comandos avanzados</b>	<b>3</b>
II.1	chmod - Gestión de permisos . . . . .	3
II.2	chown - Cambio de propiedad . . . . .	3
II.3	sudo - Ejecución con privilegios . . . . .	3
II.4	Monitoreo de Sistema (top/htop y ps) . . . . .	3
II.5	kill - Terminación de procesos . . . . .	3
II.6	tar - Compresión y empaquetado . . . . .	3
<b>III</b>	<b>Programación Shell (Bash Scripting)</b>	<b>4</b>
III.1	Estructura de un script (Shebang) . . . . .	4
III.2	Uso de variables y paso de argumentos . . . . .	4
III.3	Estructuras de control: if-else, for, while . . . . .	4
III.4	Scripts Funcionales Completos . . . . .	5
<b>IV</b>	<b>Guía de Diagnóstico: Errores Comunes</b>	<b>5</b>
IV.1	Error 1: Permission Denied . . . . .	5
IV.2	Error 2: Command Not Found . . . . .	5
IV.3	Error 3: No space left on device . . . . .	5
IV.4	Error 4: Directory not empty . . . . .	5
IV.5	Error 5: Could not get lock /var/lib/dpkg/lock . . . . .	5
<b>V</b>	<b>Tuberías y Redireccionamientos</b>	<b>5</b>
V.1	Conceptos de stdin, stdout y stderr . . . . .	6
V.2	Operadores de Redirección y Pipes . . . . .	6
V.3	Ejemplos Prácticos de Flujo . . . . .	6

## Sección I: Linux comandos básicos

---

### I.1. ls - Listar archivos

Muestra el contenido del directorio actual. Es el comando más utilizado para verificar qué archivos existen.

- **Ejemplo 1:** `ls -l` (Muestra una lista detallada con permisos, dueños, tamaño y fecha de modificación).
- **Ejemplo 2:** `ls -a` (Muestra todos los archivos, incluyendo aquellos que están ocultos y empiezan con un punto).

### I.2. cd - Cambiar de directorio

Permite navegar a través de la jerarquía de carpetas del sistema de archivos de Linux.

- **Ejemplo 1:** `cd Documentos` (Entra directamente a la carpeta llamada Documentos).
- **Ejemplo 2:** `cd ..` (Retrocede un nivel en la jerarquía, regresando a la carpeta superior).

### I.3. pwd - Directorio actual

Muestra la ruta absoluta desde la raíz (/) hasta la posición donde te encuentras actualmente.

- **Ejemplo 1:** `pwd` (Muestra la ruta estándar del usuario, ej. `/home/usuario`).
- **Ejemplo 2:** `pwd -P` (Muestra la ruta física real, resolviendo cualquier enlace simbólico).

### I.4. mkdir - Crear carpetas

Crea directorios nuevos en la ubicación indicada.

- **Ejemplo 1:** `mkdir proyectos` (Crea una sola carpeta llamada proyectos).
- **Ejemplo 2:** `mkdir -p alfa/beta/gamma` (Crea toda la ruta de carpetas anidadas de una sola vez).

### I.5. touch - Crear archivos vacíos

Crea un archivo nuevo sin contenido o actualiza la fecha de modificación de uno existente.

- **Ejemplo 1:** `touch notas.txt` (Crea un archivo de texto vacío).
- **Ejemplo 2:** `touch index.html script.js` (Crea múltiples archivos de diferentes extensiones simultáneamente).

### I.6. cp - Copiar archivos y carpetas

Duplica información de un origen a un destino.

- **Ejemplo 1:** `cp archivo.txt copia.txt` (Realiza una copia simple de un archivo).
- **Ejemplo 2:** `cp -r carpeta_original carpeta_nueva` (Copia una carpeta completa con todo su contenido de forma recursiva).

### I.7. rm - Eliminar

Borra datos del sistema. **Nota crítica:** En Linux, esta acción es generalmente irreversible desde la terminal.

- **Ejemplo 1:** `rm basura.txt` (Elimina un archivo específico).
- **Ejemplo 2:** `rm -rf carpeta_vieja` (Borra una carpeta y todo lo que contiene de forma forzada).

## I.8. cat - Visualizar contenido

Muestra el texto contenido en un archivo directamente en la terminal.

- **Ejemplo 1:** `cat lista.txt` (Lee el contenido completo del archivo).
- **Ejemplo 2:** `cat f1.txt f2.txt > union.txt` (Concatena dos archivos y guarda el resultado en uno nuevo).

## Sección II: Linux comandos avanzados

---

### II.1. chmod - Gestión de permisos

Controla quién puede leer (**r**), escribir (**w**) o ejecutar (**x**) un archivo.

- **Ejemplo 1:** `chmod +x script.sh` (Otorga permisos de ejecución al archivo).
- **Ejemplo 2:** `chmod 644 imagen.png` (Establece permisos de lectura/escritura para el dueño y solo lectura para los demás).

### II.2. chown - Cambio de propiedad

Define qué usuario y qué grupo son dueños de un recurso. Requiere privilegios de administrador.

- **Ejemplo 1:** `sudo chown usuario archivo.txt` (Cambia el dueño a un usuario específico).
- **Ejemplo 2:** `sudo chown -R www-data:www-data /var/www` (Cambia dueño y grupo a toda una estructura web).

### II.3. sudo - Ejecución con privilegios

Permite ejecutar comandos con el poder del superusuario (root).

- **Ejemplo 1:** `sudo apt update` (Actualiza los índices de los repositorios del sistema).
- **Ejemplo 2:** `sudo su` (Abre una sesión de terminal directamente como usuario root).

### II.4. Monitoreo de Sistema (top/htop y ps)

Herramientas para supervisar el rendimiento y los procesos activos.

- **Ejemplo 1:** `top` (Muestra los procesos en tiempo real ordenados por consumo de CPU).
- **Ejemplo 2:** `ps aux | grep chrome` (Busca todos los procesos relacionados con el navegador Chrome).

### II.5. kill - Terminación de procesos

Detiene la ejecución de programas mediante su ID de proceso (PID).

- **Ejemplo 1:** `kill 1234` (Envía una señal de término suave).
- **Ejemplo 2:** `kill -9 5678` (Fuerza el cierre inmediato y radical del proceso).

### II.6. tar - Compresión y empaquetado

Maneja archivos comprimidos, fundamental para respaldos.

- **Ejemplo 1:** `tar -cvf respaldo.tar /documentos` (Crea un paquete sin comprimir).
- **Ejemplo 2:** `tar -xzvf datos.tar.gz` (Extrae el contenido de un archivo comprimido con gzip).

## Sección III: Programación Shell (Bash Scripting)

---

### III.1. Estructura de un script (Shebang)

El Shebang indica al kernel qué binario debe interpretar el archivo de texto.

- **Línea obligatoria:** `#!/bin/bash`
- **Ejemplo 1 (Hola Mundo completo):**

```
#!/bin/bash
# Este es un comentario descriptivo
echo "Iniciando el sistema de automatizacion..."
echo "Usuario actual: $USER"
```

- **Ejemplo 2 (Ejecución):** Para correr un script se requiere: `chmod +x mi_script.sh && ./mi_script.sh`

### III.2. Uso de variables y paso de argumentos

Las variables dinámicas permiten que los scripts sean reutilizables.

- **Ejemplo 1 (Variables de entorno y locales):**

```
HORA=$(date +%H:%M)
echo "La hora de ejecucion es: $HORA"
```

- **Ejemplo 2 (Argumentos de línea de comandos):** Captura datos externos con `$1`, `$2`, etc.

```
echo "Procesando el archivo de origen: $1"
echo "Enviando a la carpeta de destino: $2"
```

### III.3. Estructuras de control: if-else, for, while

Permiten la toma de decisiones y ciclos repetitivos.

- **Ejemplo 1 (Condicional de existencia):**

```
if [ -f "archivo.txt" ]; then
    echo "El archivo existe."
else
    echo "Archivo no encontrado."
fi
```

- **Ejemplo 2 (Bucle For para renombrado):**

```
for file in *.txt; do
    mv "$file" "old_$file"
done
```

### III.4. Scripts Funcionales Completos

Creación de scripts para tareas como backups o instaladores.

- **Ejemplo 1 (Backup Automatizado con Fecha):**

```
#!/bin/bash
DESTINO="/home/usuario/backups"
FECHA=$(date +%Y%m%d_%H%M)
tar -czf $DESTINO/respaldo_$FECHA.tar.gz /home/usuario/datos
echo "Respaldo creado con éxito."
```

- **Ejemplo 2 (Instalador Automático):**

```
#!/bin/bash
echo "Instalando dependencias de desarrollo..."
sudo apt update && sudo apt install -y git build-essential curl
echo "Instalacion finalizada."
```

## Sección IV: Guía de Diagnóstico: Errores Comunes

---

### IV.1. Error 1: Permission Denied

Este error indica que el usuario actual no tiene los bits de permisos necesarios.

- **Causa:** Intentar ejecutar un archivo sin permisos o escribir en zonas protegidas.
- **Solución:** `chmod +x` archivo o anteponer `sudo`.

### IV.2. Error 2: Command Not Found

El shell no localiza el comando en las rutas definidas.

- **Solución:** Instalar con `apt` o verificar la ortografía del comando.

### IV.3. Error 3: No space left on device

El almacenamiento físico ha llegado a su límite.

- **Solución:** Ejecutar `df -h` y limpiar archivos innecesarios.

### IV.4. Error 4: Directory not empty

El comando de borrado falla si hay archivos dentro.

- **Solución:** Usar el parámetro recursivo: `rm -rf nombre_directorio`.

### IV.5. Error 5: Could not get lock /var/lib/dpkg/lock

Conflicto entre procesos de gestión de paquetes.

- **Solución:** Esperar a que termine la actualización automática o cerrar procesos de `apt`.

## Sección V: Tuberías y Redireccionamientos

---

En Linux "todo es un flujo de datos".

## V.1. Conceptos de stdin, stdout y stderr

Linux trata las entradas y salidas como flujos direccionables.

- **Standard Input (stdin):** Entrada de datos.
- **Standard Output (stdout):** Salida exitosa.
- **Standard Error (stderr):** Salida de errores.

## V.2. Operadores de Redirección y Pipes

Documentación del uso de operadores clave.

- **|** (pipe): Conecta comandos.
- **>** (sobrescribir) y **>>** (añadir).
- **2>** (redirigir errores).

## V.3. Ejemplos Prácticos de Flujo

Ejemplos obligatorios para demostrar comprensión.

1. **Filtro Combinado:** `ls /bin | grep "sh" —`
2. **Conteo y Ordenamiento:** `cat usuarios.txt | sort — | wc -l —`
3. **Registro de Errores:** `ls /directorio/inexistente 2> errores.log`