MATEOPS

LINUX

DESDE LA TERMINAL HASTA LA OPERACIÓN REAL

Fernando Loitey García

Aprendé Linux para operar infraestructura real

Página legal

Este libro tiene fines educativos. Los comandos, configuraciones y procedimientos fueron preparados para entornos de práctica controlados. Antes de aplicarlos sobre un sistema real, el lector debe comprender su alcance, adaptarlos a su distribución y contexto, conservar la información necesaria para recuperar el estado anterior y respetar las políticas de la organización.
Aunque el contenido fue revisado con cuidado, los sistemas cambian y una misma orden puede producir resultados diferentes según la distribución, versión, permisos, paquetes instalados y configuración local. La ausencia de un mensaje de error no garantiza que el resultado sea el esperado: observar y validar forman parte de cada tarea.
Linux, Red Hat, Rocky Linux, Ubuntu, Debian, Oracle Linux, Windows, Zabbix y los demás nombres de productos o proyectos mencionados pertenecen a sus respectivos titulares. Su uso en este libro es exclusivamente descriptivo y educativo.
Los nombres de personas, usuarios, servidores, aplicaciones y organizaciones utilizados en los ejemplos son ficticios o fueron adaptados. Las situaciones narrativas combinan experiencias habituales de infraestructura y no describen necesariamente un incidente, sistema o empresa concretos.

Dedicatoria

A quienes alguna vez abrieron una terminal sin saber por dónde empezar, pero decidieron quedarse un rato más, como me pasó a mí el primer día que enfrenté a un Linux, “revisar un Ubuntu Desktop 10.04 que tenía un GroundWork” y era el sistema de monitoreo en producción, también a mi pareja, Camila, quien me bancó en la locura de escribir este libro, perdiéndome horas de estar compartiendo momentos juntos, pero entendiendo y apoyándome en esta aventura hermosa.

Antes de comenzar

Trabajo hace más de quince años en tecnología, principalmente en infraestructura, Linux, Windows, automatización y plataformas empresariales. Durante ese tiempo administré servidores Linux y Windows, trabajé con tecnologías Red Hat y Debian, virtualización con VMware, nubes privadas basadas en OpenStack, Ansible, Docker, Kubernetes y OpenShift. También investigué incidentes y aprendí, muchas veces, frente a una terminal que no ofrecía respuestas evidentes.

La experiencia no llegó como una colección ordenada de comandos. Llegó en forma de solicitudes incompletas, alertas, servicios que parecían estar bien, discos que se llenaban, permisos que no explicaban por sí solos un acceso denegado y cambios pequeños capaces de afectar mucho más de lo esperado. También llegó trabajando con otras personas: preguntando, leyendo documentación, observando cómo alguien resolvía un problema y tratando de entender por qué su método funcionaba.

Con el tiempo comencé a compartir contenido técnico en redes, videos y encuentros de Café Infra. Ahí confirmé algo que ya había visto dentro de los equipos de trabajo: existe mucha gente con ganas de aprender Linux, pero no siempre encuentra un punto de entrada que resulte cercano. Algunos materiales comienzan suponiendo demasiadas cosas. Otros entregan listas enormes de comandos que parecen útiles hasta que uno se queda solo frente al cursor.

Ese cursor puede imponer respeto. La terminal no muestra botones, no anticipa qué deberíamos escribir y rara vez explica por sí sola qué tan grande será el alcance de una instrucción. Al comienzo parece que todo depende de memorizar. Después descubrimos que el trabajo real se apoya mucho más en saber ubicarse, formular una pregunta, leer una salida y comprobar el resultado.

Este libro nació para acompañar ese recorrido.

No quise escribir una enciclopedia de Linux ni una preparación disfrazada para una certificación. Tampoco una colección de anécdotas sobre infraestructura. La idea es más concreta: llevar a una persona desde sus primeras preguntas en una terminal hasta una base operativa que le permita observar un sistema, realizar tareas básicas, interpretar problemas frecuentes y comenzar a tomar decisiones con criterio.

La experiencia aparece en estas páginas como contexto. Hay aplicaciones internas, servidores pequeños, documentación incompleta, pedidos poco precisos y errores humanos. Son situaciones reconocibles para cualquiera que haya trabajado alrededor de sistemas reales. Pero la historia siempre está al servicio del aprendizaje: plantea una necesidad y nos obliga a descubrir qué concepto o herramienta puede ayudarnos.

MateOps también nace de esa idea. Aprender infraestructura no consiste solamente en conocer la respuesta correcta; requiere practicar, equivocarse sin romper un sistema importante, volver a observar y explicar qué evidencia sostiene una conclusión. Este libro es una de las puertas de entrada a ese enfoque.

Sobre este libro

MateOps — Linux desde la terminal hasta la operación real está dirigido principalmente a quien tiene poca experiencia con Linux y quiere comenzar a desenvolverse con seguridad desde la línea de comandos.
También puede ser útil para quien ya conoce algunos comandos, pero siente que los utiliza de memoria, o para profesionales de nivel intermedio que quieran ordenar fundamentos y recuperar mejores hábitos de diagnóstico.
No necesitás venir de una carrera universitaria, trabajar actualmente en infraestructura ni conocer una distribución específica. Sí necesitás curiosidad, tiempo para probar y disposición para leer lo que el sistema responde antes de avanzar.

Qué vas a encontrar

Comandos presentados desde una necesidad concreta, con sintaxis, opciones útiles y salidas posibles.
Explicaciones para interpretar esas salidas, no solamente instrucciones para copiarlas.
Prácticas pequeñas y reproducibles, acompañadas por validaciones.
Errores frecuentes y decisiones operativas que ayudan a evitar cambios impulsivos.
Cheatsheets por capítulo y un recorrido progresivo que vuelve a utilizar lo aprendido.
Prompts de práctica asistida para generar ejercicios adicionales sin recibir inmediatamente la solución.

Lo que no es..

No es una enciclopedia de comandos ni pretende cubrir todas sus opciones.
No sustituye los manuales, la documentación oficial ni los procedimientos de una organización.
No prepara específicamente para RHCSA u otra certificación, aunque varios fundamentos sean relevantes.
No transforma a una persona en especialista en redes, seguridad, almacenamiento o Bash dentro de un solo volumen.
No promete experiencia sin práctica. La confianza frente a una terminal se construye utilizándola.
El libro toma Rocky Linux y la familia Red Hat como referencia principal. Muchos conceptos y comandos son compartidos por Ubuntu, Debian, Oracle Linux y otras distribuciones; cuando una diferencia resulta importante, la señalamos. También aparecen Windows, NFS, SMB y otros componentes porque la infraestructura real suele ser híbrida.

Cómo utilizar este libro

Podés leerlo de principio a fin como un ebook. Las situaciones, explicaciones y salidas están escritas para que el recorrido tenga sentido incluso cuando no estés frente a una computadora.
Sin embargo, vas a aprovecharlo mucho más si tenés una terminal cerca.
La opción recomendada es utilizar una máquina virtual dedicada con Rocky Linux. No necesitás una infraestructura compleja: un entorno descartable permite crear archivos, modificar servicios de práctica y equivocarte sin comprometer información importante.
Algunos ejercicios proponen una segunda máquina o interfaz, pero el propio capítulo indica cuándo es necesaria.
Leé primero la situación y tratá de anticipar qué comprobarías. Después ejecutá los comandos de a uno. Compará tu salida con el ejemplo, pero no esperes que sea idéntica: el hostname, los usuarios, las interfaces, los procesos, los tiempos y las versiones cambian entre sistemas.
Cuando el capítulo proponga una validación, no la saltees. Ejecutar un comando sin recibir errores solamente demuestra que la shell pudo iniciarlo. No confirma que haya afectado el objeto correcto ni que el sistema haya quedado en el estado esperado.
Usá man y --help incluso cuando el ejemplo parezca claro. El objetivo no es que dependas para siempre de este libro, sino que aprendas a pedirle ayuda al propio sistema.
Cada capítulo termina con un cheatsheet. Utilizalo para recordar una intención y recuperar la sintaxis; si no comprendés el alcance de una orden, regresá a la explicación antes de ejecutarla.

Práctica asistida con IA

Al final de cada capítulo encontrarás un prompt listo para copiar en un asistente de inteligencia artificial. Su función es generar cinco ejercicios progresivos, pedirte comandos, salidas e interpretación y permitirte reintentar antes de mostrar una respuesta.
La IA puede adaptar una práctica, plantear otra variante y ayudarte a revisar el razonamiento. No puede conocer automáticamente el estado real de tu equipo ni decidir qué es seguro en tu organización. No compartas contraseñas, claves, tokens, direcciones privadas ni información sensible. Si un ejercicio excede tu entorno de laboratorio, trabajá sobre una salida simulada.
Estos prompts no reemplazan los laboratorios diseñados y verificados de MateOps. Son una herramienta adicional para reforzar conceptos y, al mismo tiempo, aprender a pedirle a la IA algo más útil que una solución inmediata.

Convenciones utilizadas

Los bloques monoespaciados pueden representar comandos, salidas, rutas, archivos de configuración o fragmentos de scripts. El texto anterior y posterior indica qué estamos observando.

Comandos y salidas

Cuando mostramos solamente la instrucción que debe escribirse, el prompt de la terminal no forma parte del comando:
`pwd`
Si necesitamos conservar el contexto, podemos mostrar la sesión completa:
`operador@rocky-lab:~$ pwd`
`/home/operador`
En ese ejemplo, operador@rocky-lab:~$ es el prompt; pwd es el comando y /home/operador es la salida.

Valores que deben sustituirse

Las palabras en mayúsculas como INTERFAZ, USUARIO, SERVICIO, RUTA o DIRECCION/PREFIJO representan valores que deben adaptarse:
`ip address show dev INTERFAZ`
No escribas INTERFAZ literalmente. Primero observá el sistema y reemplazala por un nombre real, por ejemplo enp1s0.
Los nombres concretos utilizados en ejemplos —rocky-lab, operador, httpd o /srv/mateops— ayudan a leer una situación realista, pero tampoco deben copiarse sin comprobar el entorno.

Usuarios normales y privilegios

La mayoría de las prácticas comienza con un usuario normal. Cuando una acción requiere privilegios, el ejemplo utiliza sudo de forma explícita:
`sudo systemctl restart SERVICIO`
No conviertas sudo en una respuesta automática frente a Permission denied. Primero confirmá identidad, objeto, alcance y motivo. Algunos sistemas pueden mostrar # para una shell de root; este libro no lo utiliza como invitación a trabajar permanentemente con esa identidad.

Registros, logs y filesystems

Utilizamos «registro» como término general en español y «log» cuando forma parte de una ruta, un comando, una salida o una expresión habitual de infraestructura.
Utilizamos «sistema de archivos» al explicar el concepto en español y «filesystem» cuando necesitamos distinguir con precisión la unidad montada y administrada por Linux, especialmente en salidas, alertas y diagnósticos de espacio.

Distribuciones y resultados

Una salida posible no es una promesa exacta. Las versiones, paquetes, nombres de interfaces, servicios y decisiones de cada distribución pueden modificarla. Cuando un comando no existe, consultá el capítulo de paquetes y verificá la documentación de tu sistema.

Las rutas y configuraciones relacionadas con RHEL o Rocky Linux pueden tener equivalentes diferentes en Debian o Ubuntu. El criterio operativo se conserva aunque cambie la herramienta.

Una nota sobre esta colaboración

Este libro fue desarrollado por Fernando Loitey en colaboración con ChatGPT.
La inteligencia artificial participó en la organización editorial, la redacción inicial, la revisión técnica, el control de continuidad y el refinamiento del manuscrito. Las experiencias, el enfoque didáctico, el criterio operativo y las decisiones finales pertenecen al autor.
Cada capítulo fue revisado, corregido y aprobado durante un proceso de trabajo conjunto. Los ejemplos y procedimientos fueron adaptados para representar situaciones creíbles, mantener una progresión adecuada y reducir riesgos durante la práctica.
Reconocer esta colaboración no es solamente una cuestión de transparencia. También representa una posición presente en todo el proyecto: la inteligencia artificial puede ayudarnos a aprender, comparar, practicar y revisar, pero no reemplaza la comprensión de un sistema ni la responsabilidad sobre una decisión.
El objetivo nunca fue pedirle a una herramienta que produjera muchas páginas. Fue utilizarla como colaboradora para construir un libro que conserve experiencia, intención y voz humana.

Índice

Capítulo 1

La terminal: entrar, orientarse y no trabajar a ciegas

El mensaje no decía demasiado:
“¿Podés revisar el servidor? Dejaron unos archivos para nosotros, pero no sabemos exactamente dónde.”
No había una ruta, un nombre de archivo ni una captura que ayudara a reducir la búsqueda. Apenas un usuario de acceso, el nombre de un equipo y una terminal abierta con el cursor esperando.
`operador@rocky-lab:~$`
Es una escena bastante común. Las solicitudes no siempre llegan con toda la información necesaria y la documentación, cuando existe, puede estar incompleta. Frente a eso es tentador empezar a escribir comandos de inmediato: buscar archivos, recorrer directorios o probar cualquier cosa que parezca acercarnos a una respuesta.
Pero todavía faltaba algo más básico. Antes de buscar esos archivos había que saber con qué usuario se había iniciado la sesión, en qué equipo estaba abierta la terminal y desde qué lugar del sistema comenzaría la búsqueda.
En Linux, trabajar sin responder esas preguntas es una buena manera de hacer lo correcto en el lugar equivocado.

El cursor está esperando una instrucción

La terminal es una forma de conversar con el sistema mediante texto. Escribimos una instrucción, presionamos Enter y el sistema responde. A veces devuelve información; otras veces no muestra nada. Aprender a distinguir la instrucción de la respuesta es el primer paso para no leer una sesión de terminal como si fuera un bloque indescifrable.
La línea que estaba en pantalla se conoce como prompt:
`operador@rocky-lab:~$`
Su apariencia cambia entre distribuciones y configuraciones. En este ejemplo ya ofrece varias pistas: operador es el usuario, rocky-lab es el nombre del equipo y el símbolo ~ representa el directorio personal del usuario. El signo $ marca el lugar desde el que se puede escribir.
No hay que copiar el prompt cuando un libro o una documentación muestran un comando. Si vemos lo siguiente:
`operador@rocky-lab:~$ whoami`
lo que escribimos es solamente:
`whoami`
Luego presionamos Enter. El resto pertenece al entorno o a la salida del comando.

Primera pregunta: ¿con qué usuario entré?

El acceso recibido podía pertenecer a una persona, a un equipo o a una cuenta creada para una tarea particular. No convenía asumirlo por el nombre que aparecía en el prompt, porque el prompt se puede personalizar. La comprobación directa se hace con whoami:

```
operador@rocky-lab:~$ whoami
operador
```

La primera línea contiene el comando. La segunda es su salida: el sistema confirma que la sesión está trabajando como el usuario operador.

whoami es un comando sin argumentos en este uso básico. Su propósito es responder una sola pregunta: cuál es el nombre del usuario efectivo con el que se ejecutan las instrucciones de esta sesión.

Puede parecer una verificación demasiado sencilla, pero cambia el contexto de todo lo que viene después. Dos personas ubicadas en el mismo directorio pueden ver resultados diferentes o recibir permisos distintos. Más adelante trabajaremos usuarios, grupos y privilegios; por ahora alcanza con instalar el hábito de comprobar la identidad antes de actuar.

Segunda pregunta: ¿en qué equipo estoy?

En una laptop personal suele ser evidente dónde se abrió la terminal. En una infraestructura con varios servidores deja de serlo muy rápido. Dos ventanas pueden verse idénticas y estar conectadas a equipos distintos.

El comando hostname muestra el nombre que identifica al sistema:

```
operador@rocky-lab:~$ hostname
rocky-lab
```

La salida confirma que la sesión está en rocky-lab. Si la solicitud hubiese mencionado otro servidor, este sería el momento de detenerse. Continuar confiando en la memoria o en el título de una ventana convertiría un dato verificable en una apuesta innecesaria.

En tareas reales conviene leer juntos los dos primeros resultados:

```
operador@rocky-lab:~$ whoami
operador
operador@rocky-lab:~$ hostname
rocky-lab
```

Ahora sabemos quién ejecuta los comandos y en qué sistema lo hace. Todavía falta saber desde dónde.

Tercera pregunta: ¿en qué directorio estoy?

Linux organiza archivos y directorios mediante rutas. Cada sesión de terminal mantiene un directorio de trabajo actual: el lugar desde el que se interpretarán muchos nombres y operaciones.

pwd significa print working directory. Podríamos traducirlo como “mostrar el directorio de trabajo actual”:

```
operador@rocky-lab:~$ pwd
/home/operador
```

La salida /home/operador es una ruta. Empieza con /, por lo tanto indica una ubicación completa desde la raíz del sistema. A este tipo de ubicación se la llama ruta absoluta.
El directorio personal de un usuario suele ser el lugar seguro para sus documentos y prácticas. En este ejemplo, el usuario operador comenzó la sesión dentro de /home/operador. El ~ que aparecía en el prompt era una forma abreviada de representar esa ubicación.
pwd no modifica nada. Observa el estado actual y lo muestra. Esta diferencia entre observar y modificar será importante durante todo el libro: antes de cambiar un sistema necesitamos herramientas que nos permitan entenderlo.

Cuarta pregunta: ¿qué hay a mi alrededor?

Ya conocemos la identidad, el equipo y el directorio. El siguiente paso es mirar qué contiene la ubicación actual. Para eso usamos ls:

```
operador@rocky-lab:~$ ls
backups  documentos  scripts
```

La salida presenta tres nombres propios de este entorno de ejemplo. En tu equipo pueden aparecer otros nombres, una cantidad diferente o ninguno; no necesitás crear backups, documentos ni scripts para continuar. En esta vista todavía no sabemos mucho sobre los elementos listados; ni siquiera distinguimos con certeza cuáles son archivos y cuáles directorios. Ese nivel de detalle llegará en el capítulo siguiente. Por ahora ls cumple una función concreta: mostrar los elementos visibles de la ubicación indicada.
Si el directorio estuviera vacío, ls podría volver al prompt sin imprimir nada:

```
operador@rocky-lab:/tmp/carpeta-vacia$ ls
operador@rocky-lab:/tmp/carpeta-vacia$
```

La ausencia de salida no significa necesariamente que el comando falló. En este caso significa que no había elementos visibles para listar. Linux no siempre confirma con una frase que una operación salió bien; hay que interpretar qué respuesta era razonable para la instrucción ejecutada.

Cómo está formado un comando

Los primeros ejemplos permiten reconocer una estructura que se repetirá durante el libro:

```
comando [opciones] [argumentos]
```

El comando indica la herramienta que queremos ejecutar. Las opciones ajustan su comportamiento. Los argumentos indican sobre qué elemento debe trabajar: una ruta, un archivo, un usuario u otro valor, según la herramienta.
Los corchetes de esta representación no se escriben literalmente. Indican que una parte es opcional. Tampoco todos los comandos necesitan opciones o argumentos. whoami, hostname y pwd ya fueron útiles sin agregar nada.
ls también puede recibir una ruta como argumento. Si estamos en el directorio personal pero queremos observar /tmp, no es obligatorio movernos hasta allí:

```
operador@rocky-lab:~$ ls /tmp
systemd-private-a81c...  tmp.A4k9pQ
```

En esa instrucción, ls es el comando y /tmp es el argumento. La salida de /tmp cambia entre sistemas y momentos; los nombres mostrados son solamente un ejemplo realista, no algo que el lector deba reproducir exactamente.
Las opciones se incorporarán cuando exista una necesidad concreta. Memorizar una docena antes de saber qué problema resuelven haría más difícil lo que en realidad es simple.

Moverse sin perder la referencia

El comando cd cambia el directorio de trabajo. Su nombre viene de change directory. A diferencia de pwd y ls, no se limita a observar: modifica el lugar desde el que continuará la sesión.
Para entrar en el directorio documentos que apareció en el listado podemos usar su nombre:

```
operador@rocky-lab:~$ cd documentos
operador@rocky-lab:~/documentos$
```

No hubo una salida textual. El cambio se percibe en el prompt, pero conviene validarlo con pwd:

```
operador@rocky-lab:~/documentos$ pwd
/home/operador/documentos
```

documentos era una ruta relativa. No comenzaba con / y se interpretó desde el directorio actual, que era /home/operador. El sistema combinó ambas referencias y terminó en /home/operador/documentos.
También podríamos llegar utilizando la ruta absoluta:

```
operador@rocky-lab:~$ cd /home/operador/documentos
```

Las dos formas pueden conducir al mismo sitio, pero expresan puntos de partida diferentes. Una ruta absoluta comienza en / y conserva su significado sin importar dónde estemos. Una ruta relativa depende del directorio de trabajo actual.
Para subir al directorio inmediatamente anterior se utiliza ..:

```
operador@rocky-lab:~/documentos$ cd ..
operador@rocky-lab:~$ pwd
/home/operador
```

Los dos puntos representan el directorio superior. No significan “volver al último lugar visitado”; significan subir un nivel dentro de la ruta.
Cuando queremos regresar al directorio personal, cd sin argumentos ofrece una forma sencilla:

```
operador@rocky-lab:/tmp$ cd
operador@rocky-lab:~$ pwd
/home/operador
```

Esta es una de esas instrucciones breves que conviene comprender y no solo recordar. cd cambia de ubicación; sin un destino explícito, utiliza el directorio personal del usuario.

Pedir ayuda al propio sistema

Nadie trabaja durante años con Linux recordando cada opción y cada detalle de sintaxis. Parte del oficio consiste en saber consultar la información disponible en el propio sistema. Una consulta rápida suele comenzar con --help:

```
operador@rocky-lab:~$ pwd --help
pwd: pwd [-LP]
    Print the name of the current working directory.
    Options:
      -L    print the value of $PWD if it names the current working directory
      -P    print the physical directory, without any symbolic links
```

La ayuda puede aparecer en inglés aunque el sistema esté configurado en español. No es necesario comprender cada línea. En este punto alcanza con reconocer el nombre de la herramienta, una descripción breve y la forma de sus opciones.
La expresión [-LP] indica que las opciones son opcionales. No necesitamos usarlas todavía. El objetivo de abrir la ayuda no es consumir toda la documentación de una vez, sino aprender dónde buscar cuando aparezca una duda concreta.
Para una explicación más desarrollada están las páginas de manual:

```
man pwd
```

El manual se abre normalmente dentro de un visor de texto. Podemos avanzar con la barra espaciadora o las flechas, buscar una palabra escribiendo / seguida del término, pasar a la coincidencia siguiente con n y salir con q.
Una práctica útil es abrir man pwd, buscar DESCRIPTION y volver a la terminal. No hace falta leer la página completa. La meta es aprender a entrar, localizar una sección y salir sin sentir que el manual es un lugar reservado para especialistas.
En algunas instalaciones mínimas, man puede no estar instalado. Si la terminal responde man: command not found, la consulta con --help sigue siendo válida. Más adelante veremos cómo identificar e instalar paquetes; por ahora no hace falta modificar el sistema para completar este capítulo.

Cuando algo no sale como esperábamos

Los primeros errores en una terminal suelen ser pequeños y muy informativos. Si escribimos mal el nombre de un comando:

```
operador@rocky-lab:~$ hostnme
bash: hostnme: command not found
```

La shell informa que no encontró una instrucción llamada hostnme. No es un fallo del servidor: falta la letra a. Leer el mensaje antes de volver a intentar suele resolver más que ejecutar variantes al azar.
Las rutas también exigen atención. Desde /home/operador, estas dos instrucciones no significan lo mismo:

```
cd documentos
cd /documentos
```

La primera busca documentos dentro de la ubicación actual. La segunda intenta encontrar un directorio llamado documentos directamente bajo /. Si ese lugar no existe, cd lo dirá:

```
operador@rocky-lab:~$ cd /documentos
bash: cd: /documentos: No such file or directory
```

Otro error común es escribir también el signo $ que aparece en ejemplos de terminal. Ese símbolo representa el prompt; no forma parte del comando.
Finalmente, cd puede completarse sin mostrar ninguna salida y aun así dejarnos en un lugar distinto. Por eso pwd funciona como validación natural. Comando ejecutado y resultado confirmado son dos momentos diferentes.

Una rutina breve antes de trabajar

La solicitud inicial sigue pendiente, pero ahora la sesión dejó de ser una ventana anónima. Cuatro comandos permitieron construir contexto sin modificar el sistema:

```
operador@rocky-lab:~$ whoami
operador
operador@rocky-lab:~$ hostname
rocky-lab
operador@rocky-lab:~$ pwd
/home/operador
operador@rocky-lab:~$ ls
backups  documentos  scripts
```

Todavía no encontramos los archivos mencionados en el mensaje. Eso no es un problema: el capítulo no buscaba resolver una investigación completa, sino evitar que comenzara desde una suposición incorrecta.

En infraestructura, unos segundos de orientación pueden ahorrar una corrección mucho más costosa. No se trata de ejecutar siempre una ceremonia rígida de cuatro comandos. Se trata de desarrollar la costumbre de conocer usuario, equipo y ubicación cuando el contexto no sea evidente.

Práctica guiada: ubicarse y regresar

Esta práctica no crea, modifica ni elimina archivos. Puede realizarse en una instalación Linux local o en una máquina virtual con un usuario normal.

Objetivo

Confirmar la identidad y el punto de partida, desplazarse a una ubicación conocida y regresar al directorio personal validando cada cambio.

Recorrido

Primero ejecutá whoami y anotá mentalmente el usuario que aparece. Luego usá hostname para identificar el equipo. Ejecutá pwd y guardá la ruta inicial en tus notas, porque todavía no trabajamos con variables de shell.

Listá el contenido de la ubicación actual con ls. No es necesario que aparezcan los mismos nombres que en los ejemplos.

Ahora cambiá a /tmp mediante cd /tmp. Confirmá el cambio con pwd y observá el contenido con ls. No modifiques nada allí.

Regresá a tu directorio personal ejecutando cd sin argumentos. Volvé a consultar pwd.

Por último, abrí la ayuda de pwd con pwd --help. Si man está disponible, consultá man pwd, buscá la palabra DESCRIPTION y salí con q.

Validación

La práctica queda completa si podés responder con evidencia estas cuatro preguntas:

`¿Qué usuario está ejecutando los comandos?`

`¿En qué equipo se abrió la terminal?`
`¿Qué ruta muestra pwd después de cd /tmp?`
`¿Qué ruta muestra pwd después de ejecutar cd sin argumentos?`
La última ruta debería coincidir con el directorio personal del usuario. No tiene por qué ser idéntica a /home/operador: ese nombre pertenece al entorno de ejemplo.

Desafío breve

Sin ejecutar todavía, intentá anticipar el resultado de esta secuencia:
`pwd`
`cd /tmp`
`pwd`
`cd`
`pwd`
Después ejecutala y compará tu predicción con las tres salidas de pwd. El objetivo no es acertar por intuición, sino comprobar que entendés qué ubicación utiliza cada comando.

Criterio operativo

Antes de modificar un sistema, ubicate. Confirmá la identidad con la que trabajás, el equipo que recibe las instrucciones y el directorio desde el que se interpretarán las rutas. Observá primero y validá cada desplazamiento importante.
No hace falta desconfiar de todo; hace falta convertir en evidencia aquello que puede comprobarse con un comando simple.

Cheatsheet del capítulo

`whoami`
Muestra el usuario efectivo de la sesión.
`hostname`
Muestra el nombre del equipo.
`pwd`
Muestra la ruta absoluta del directorio de trabajo actual.
`ls`
Lista los elementos visibles del directorio actual.
`ls /ruta`
Lista los elementos visibles de la ruta indicada sin cambiar de directorio.
`cd directorio`
Cambia a un directorio indicado mediante una ruta relativa.
`cd /ruta/completa`
Cambia a un directorio mediante una ruta absoluta.
`cd ..`
Sube al directorio inmediatamente superior.
`cd`
Regresa al directorio personal del usuario.
`comando --help`

Muestra la ayuda breve disponible para un comando.
`man comando`
Abre la página de manual de un comando. Dentro del visor: / busca, n avanza a la coincidencia siguiente y q sale.

Conceptos para recordar

El prompt indica que la terminal está lista para recibir una instrucción, pero no forma parte del comando. Una ruta absoluta comienza en /; una ruta relativa se interpreta desde el directorio actual. Observar, modificar y validar son acciones distintas. Antes de ejecutar, conviene saber quién, dónde y desde qué lugar está trabajando.

Seguí practicando con IA

Si querés reforzar este capítulo, copiá el siguiente prompt en tu asistente de IA. La práctica se adapta a tu entorno y avanza de a un ejercicio por vez.

```
Actuá como instructor y evaluador de Linux para reforzar el capítulo 1 — la terminal: entrar, orientarse y no trabajar a ciegas.
Prepará exactamente cinco ejercicios progresivos sobre orientación inicial con whoami, hostname, pwd, ls, cd, rutas absolutas y relativas, man y --help.
Presentalos de a uno: no muestres el siguiente hasta que yo responda al actual.
Antes del primer ejercicio, preguntame qué distribución utilizo, si estoy en una máquina local o VM de laboratorio y si tengo sudo. Adaptá los nombres de interfaces, servicios, paquetes y rutas a mi respuesta.
En cada ejercicio:
- entregá una situación breve, un objetivo y el estado inicial;
- no reveles comandos, pistas ni solución salvo que yo los solicite;
- pedime el comando utilizado, la salida relevante, su interpretación y una validación;
- corregí mi respuesta con precisión y permitime reintentar antes de mostrar una solución;
- aumentá gradualmente la dificultad sin introducir temas posteriores al capítulo;
- al finalizar los cinco, resumí qué comprendí y qué debería practicar otra vez.
No modifiques archivos del sistema. Si necesitás crear algo, hacelo únicamente dentro de ~/practica-terminal.
No me pidas contraseñas, claves, tokens, direcciones privadas ni información real de una organización. Si mi entorno no permite una práctica segura, convertí ese ejercicio en análisis de una salida simulada y aclaralo.
```

Puente al próximo capítulo

Ya sabemos responder las primeras preguntas frente a una terminal: quiénes somos, en qué equipo estamos, desde qué directorio comenzamos y qué tenemos alrededor.

El siguiente paso es entender ese lugar dentro de una estructura mayor. En Linux no existen unidades separadas como C: o D: organizando la vista del usuario. Todo comienza en un único árbol, y la ruta de un archivo suele decir bastante sobre su función.
Antes de crear o modificar algo, necesitamos aprender a leer ese árbol.