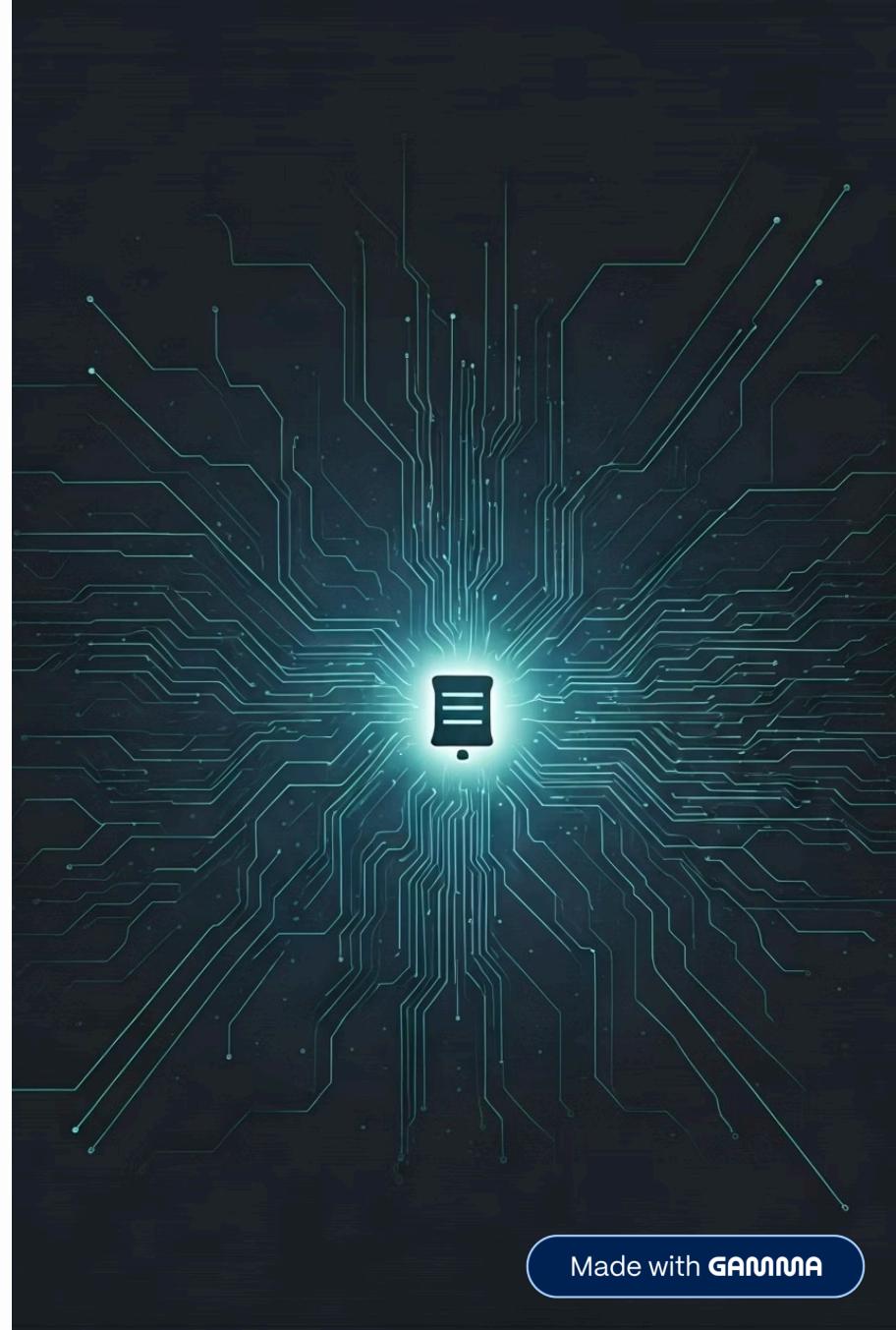
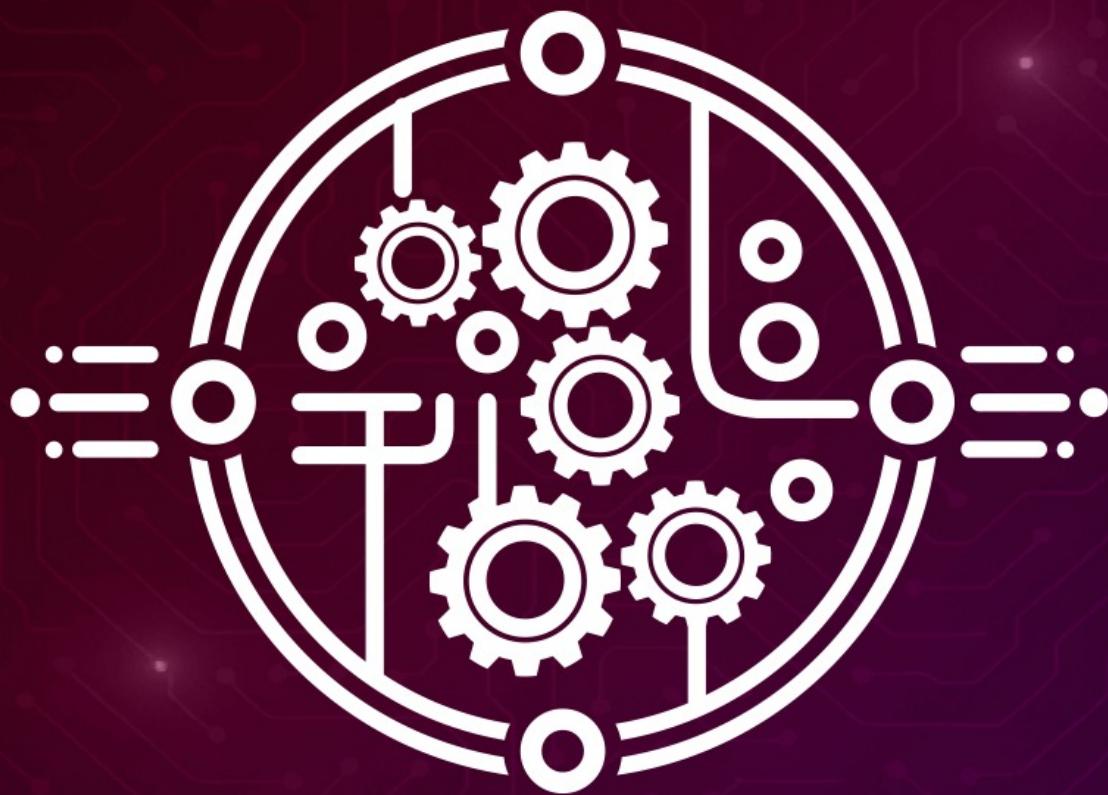


Introducción a la Ciberseguridad

Clase 2: Herramientas de Análisis y
Privilegios





∴ ENIGMA ∴

CONGRESO ESTUDIANTIL DE INFORMÁTICA



SÁBADO
20 DE SEPTIEMBRE



50 Y 120
FACULTAD DE INFORMÁTICA UNLP



UNIVERSIDAD
NACIONAL
DE LA PLATA



Facultad de
INFORMÁTICA
UNIVERSIDAD NACIONAL DE LA PLATA

FUA
FEDERACIÓN UNIVERSITARIA ARGENTINA

CEFI {
Centro de Estudiantes de la Facultad de Informática

Agenda del Día

¿Qué veremos hoy?



01

Repaso Rápido

Comandos clave de la Clase 1: una pincelada para refrescar la memoria.

02

Privilegios

El poder de `sudo` y el rol fundamental del superusuario.

03

Procesos

¿Qué está haciendo tu sistema ahora mismo? Monitoreo y análisis.

04

Búsqueda

Encontrando agujas en un pajar digital con `grep` y `find`.

05

Redes

Los primeros pasos para entender la comunicación de tu máquina con el mundo exterior.

El Superusuario: root

El "Administrador" de Linux

En el universo Linux, el usuario root es el equivalente al "Administrador" en otros sistemas operativos. Posee **poder absoluto** sobre el sistema. Esto significa que puede realizar cualquier operación: instalar o desinstalar programas, modificar archivos de sistema críticos, acceder a cualquier directorio, e incluso, inadvertidamente, dañar el sistema de forma irreparable.

Concepto Clave de Seguridad

Utilizar el usuario root para tareas cotidianas es una práctica extremadamente peligrosa y desaconsejada. Un simple error al ejecutar un comando como root puede tener consecuencias catastróficas, desde la eliminación accidental de archivos importantes hasta la corrupción total del sistema operativo. Por eso, la regla de oro es: **nunca operes como root a menos que sea estrictamente necesario.**



La Meta en Ciberseguridad

Desde la perspectiva de un atacante o analista de seguridad, el objetivo final al comprometer un sistema es, en muchos casos, **obtener acceso root**. Este proceso se conoce como "**escalada de privilegios**". Una vez que se logra el acceso root, el atacante tiene control total sobre la máquina, pudiendo instalar software malicioso, robar datos, o utilizar el sistema como plataforma para atacar otras redes.



sudo: Super User Do

La solución elegante y segura para realizar tareas administrativas en Linux sin comprometer la integridad del sistema es el comando `sudo`. Su nombre, que significa "**Super User Do**", encapsula perfectamente su función: permite ejecutar **UNA SOLA ORDEN** con los privilegios del superusuario (root).

Cuando utilizas `sudo`, el sistema te pedirá tu **propia contraseña** de usuario, no la contraseña de root. Esto es crucial por varias razones:

- **Responsabilidad:** Las acciones realizadas quedan registradas bajo tu usuario, facilitando la auditoría.
- **Seguridad:** No necesitas conocer ni almacenar la contraseña de root.
- **Minimización de Riesgos:** Si cometes un error, su impacto está limitado a la acción específica que ejecutaste con `sudo`, no a toda tu sesión.

`sudo` es la forma **correcta y segura** de realizar tareas que requieren privilegios elevados, como instalar software, actualizar el sistema, o modificar archivos de configuración vitales. Es la herramienta indispensable para todo administrador de sistemas y estudiante de ciberseguridad.

La Necesidad de sudo

Permiso Denegado

Paso 1: Intenta sin sudo

Abre una terminal y ejecuta el siguiente comando:

```
apt update
```

Este comando intenta actualizar la lista de paquetes disponibles en los repositorios de tu sistema. Es una tarea que requiere permisos elevados porque modifica archivos críticos del sistema.

Resultado esperado:

Verás un mensaje similar a: E: No se pudo abrir el archivo de bloqueo /var/lib/apt/lists/lock - open (13: Permiso denegado) o E: No se pudo bloquear el directorio /var/lib/apt/lists/.

Esto es el sistema protegiéndote. Como usuario normal, no tienes permiso para escribir en esos directorios o archivos.

Paso 2: Usa sudo

Ahora, intenta el mismo comando, pero anteponiendo sudo:

```
sudo apt update
```

El sistema te pedirá tu contraseña de usuario (la que usas para iniciar sesión). Una vez que la ingreses correctamente, el comando se ejecutará con éxito.

Resultado esperado:

Verás que el sistema comienza a descargar y actualizar las listas de paquetes. ¡Funciona! Has utilizado sudo para pedir prestado el poder de root de forma segura y controlada.

Esta es la forma estándar de interactuar con el sistema para actualizaciones o instalaciones de software.

Comandos de Identidad

En el mundo de la ciberseguridad y la administración de sistemas, es fundamental saber quién eres dentro del sistema y a qué grupos perteneces. Esto te permite entender tus propios permisos y, en un contexto de penetration testing o análisis forense, identificar posibles vías para la escalada de privilegios o para entender el rol de un usuario comprometido.



whoami

Este comando es tan sencillo como directo: te dice el **nombre de tu usuario actual**. Es útil para una verificación rápida cuando trabajas con múltiples cuentas o en diferentes contextos de shell.

Ejemplo:

```
whoami
```

```
estudiante_unlp
```



id

El comando **id** te ofrece una visión **mucho más detallada** de tu identidad dentro del sistema. Muestra:

- **Tu ID de usuario (UID)**: un número único asignado a tu cuenta.
- **Tu grupo principal (GID)**: el grupo por defecto al que perteneces.
- **Todos los grupos a los que perteneces**: fundamental para entender los permisos de acceso a archivos y recursos compartidos.

Ejemplo:

```
id
```

```
uid=1000(estudiante_unlp) gid=1000(estudiante_unlp)
grupos=1000(estudiante_unlp),4(adm),24(cdrom),27(sudo),3
0(dip),46(plugdev),116(lpadmin)
```

Entender tu **UID** y los grupos a los que perteneces es esencial. Por ejemplo, si un usuario pertenece al grupo **sudo**, significa que puede ejecutar comandos con privilegios elevados, lo que lo convierte en un objetivo de interés para un atacante.

¿Qué son los Procesos?

Para comprender cómo funciona un sistema operativo y, más aún, cómo se puede proteger o analizar su comportamiento, es fundamental entender el concepto de **proceso**.

El Sistema Está Vivo

Imagina tu computadora como una ciudad en constante actividad. Cada vez que abres una aplicación, escuchas música, navegas por internet o incluso cuando el sistema realiza tareas en segundo plano, algo se está ejecutando. Ese "algo" es un proceso.

Un **proceso** es, en esencia, **cualquier programa que se está ejecutando** en tu máquina. Desde tu navegador web con múltiples pestañas (cada pestaña podría ser un proceso o hilo dentro de un proceso), hasta el más pequeño script del sistema que comprueba la hora o gestiona la conexión a internet, **todo es un proceso**.

Características Clave de un Proceso:

- **Recursos:** Cada proceso consume recursos del sistema, principalmente CPU (tiempo de procesamiento), memoria RAM, y a veces espacio en disco o recursos de red.
- **ID Único (PID):** El sistema operativo asigna un identificador único a cada proceso, llamado Process ID (PID). Esto permite al sistema gestionar y rastrear cada ejecución de forma individual.
- **Estado:** Los procesos pasan por diferentes estados (ejecutándose, esperando, suspendido, terminado, etc.) a lo largo de su ciclo de vida.



La Importancia en Ciberseguridad:

En el ámbito de la ciberseguridad, el análisis de procesos es una herramienta invaluable:

- **Detección de Actividad Sospechosa:** Un proceso desconocido que consume mucha CPU o red, o que se ejecuta desde un directorio inusual, podría ser indicio de malware o una intrusión.
- **Análisis de Malware:** Los analistas de malware estudian cómo los programas maliciosos se ejecutan y qué procesos crean o modifican.
- **Monitoreo:** Permite identificar aplicaciones no autorizadas o servicios que están drenando recursos del sistema sin justificación.

htop

Mientras que comandos como ps te permiten listar procesos, htop es una herramienta de monitoreo de procesos **interactiva y visual** para la terminal, que te ofrece una sala de control en tiempo real de lo que ocurre en tu sistema.

¿Por qué htop?

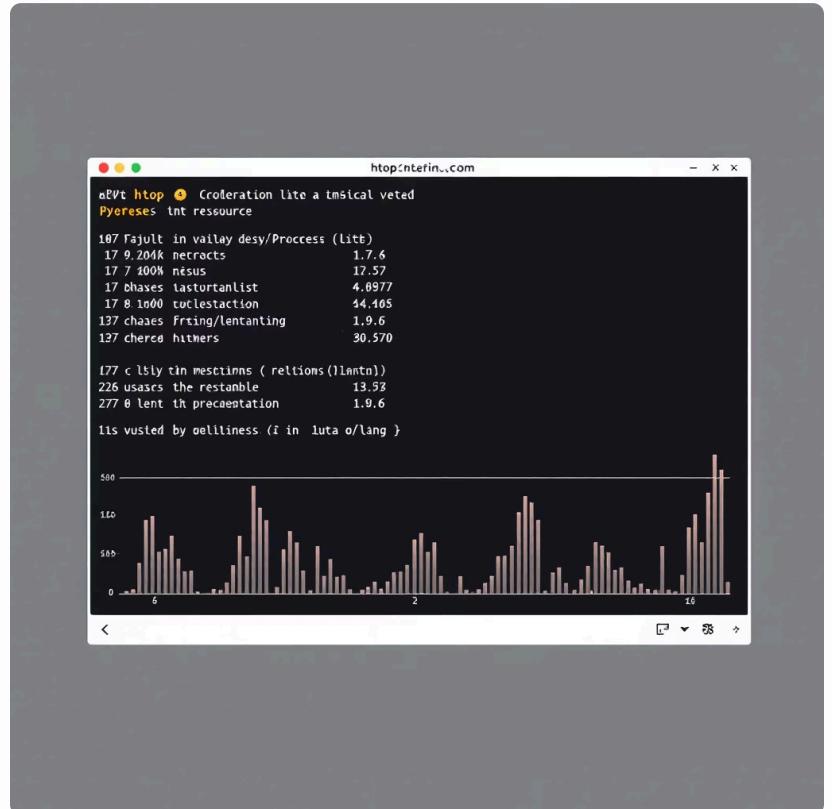
- **Interfaz Amigable:** A diferencia de top, htop es más intuitivo, muestra colores, y permite la navegación con el teclado y el ratón.
- **Información en Tiempo Real:** Actualiza constantemente los datos de uso de CPU, memoria y la lista de procesos, permitiéndote ver cambios instantáneamente.
- **Análisis Rápido:** Identifica rápidamente qué procesos están consumiendo más recursos, lo cual es crucial para diagnosticar problemas de rendimiento o detectar actividad anómala.

Instalación

Si htop no está preinstalado en tu distribución de Linux, puedes instalarlo fácilmente usando apt:

```
sudo apt install htop
```

Una vez instalado, simplemente escribe htop en tu terminal y presiona Enter para iniciar esta poderosa herramienta.



Entendiendo htop

Al abrir htop, te encontrarás con una gran cantidad de información. Aquí te desglosamos las columnas más importantes para un análisis inicial:

PID	Process ID: Es el número de identificación único asignado por el sistema operativo a cada proceso. Es como el DNI de cada programa en ejecución. Fundamental para referenciar un proceso específico.
USER	Usuario: Indica qué usuario inició el proceso. Esto es vital en ciberseguridad para identificar procesos ejecutados por usuarios no esperados o por usuarios de sistema que podrían haber sido comprometidos.
PRI / NI	Prioridad / Nice: Muestran la prioridad de ejecución de un proceso. Un valor "nice" más bajo (o negativo) indica mayor prioridad. Los atacantes pueden intentar reducir la prioridad de sus procesos para pasar desapercibidos.
VIRT / RES / SHR	Memoria Virtual / Residente / Compartida: Estos valores muestran el uso de memoria por parte del proceso. RES (Resident Set Size) es la cantidad de memoria RAM física que el proceso está utilizando actualmente y que no puede ser intercambiada a disco. Valores altos pueden indicar fugas de memoria o uso excesivo.
CPU%	Porcentaje de CPU: Indica el porcentaje de tiempo de procesador que el proceso está consumiendo. Un proceso malicioso, como un minador de criptomonedas o un keylogger activo, a menudo mostrará un uso inusualmente alto de CPU.
MEM%	Porcentaje de Memoria: Muestra la cantidad de memoria RAM que el proceso está utilizando en relación con la memoria total disponible. Un uso excesivo puede ser un indicio de un proceso mal configurado o una amenaza.
TIME+	Tiempo de CPU acumulado: El tiempo total que el proceso ha estado usando la CPU desde que se inició. Un valor muy alto para un proceso que no esperas que sea intensivo en CPU es una señal de alerta.
COMMAND	Comando: El nombre del programa o el comando completo que inició el proceso. Fundamental para identificar de qué se trata el proceso.

Interactuando con htop

htop no es solo un monitor; es una herramienta interactiva que te permite gestionar los procesos directamente desde la terminal. Esto es extremadamente útil para diagnosticar problemas, liberar recursos o incluso terminar procesos sospechosos.



Moverse y Ordenar

Utiliza las **flechas** del teclado (arriba/abajo) para navegar por la lista de procesos. Para ordenar la lista por una columna específica (por ejemplo, CPU% o MEM%), simplemente haz clic en el encabezado de esa columna o usa las teclas **F6** y las flechas para seleccionar el criterio de ordenación.



Buscar Procesos

Si la lista de procesos es muy larga, puedes buscar uno específico presionando **F3**. Esto abrirá un cuadro de búsqueda en la parte inferior. Escribe el nombre del proceso o parte de él y presiona Enter. htop te llevará al primer proceso que coincide.



"Matar" un Proceso (Terminar)

Para detener un proceso, selecciónalo con las flechas y presiona **F9**. Esto enviará una señal al proceso. Generalmente, la señal por defecto (**SIGTERM**, señal 15) es suficiente para que el proceso termine limpiamente. Si no responde, puedes probar con **SIGKILL** (señal 9), que lo fuerza a terminar inmediatamente.

¡Práctica Guiada!

Paso 1: Abre una **segunda terminal** (mantén htop abierto en la primera).

Paso 2: En la segunda terminal, ejecuta un proceso que no haga nada por un tiempo largo, como:

```
sleep 1000
```

Este comando hará que el sistema "duerma" durante 1000 segundos, creando un proceso visible.

Paso 3: Vuelve a tu primera terminal donde está htop. Busca el proceso sleep (usa **F3** si es necesario).

Paso 4: Una vez localizado el proceso sleep, selecciónalo y presiona **F9**. Confirma la acción con Enter.

Resultado: El proceso sleep debería desaparecer de la lista de htop, indicando que lo has terminado con éxito. ¡Felicitaciones, has tomado el control de un proceso!

El comando Pipe (|)

En el vasto universo de la línea de comandos de Linux, pocos operadores son tan fundamentales y versátiles como el "pipe" o tubería, representado por el carácter vertical |. Este simple símbolo es la clave para desatar un nivel de eficiencia y automatización que transformará la forma en que interactúan con sus sistemas.

¿Qué es el Pipe?

Imaginen el pipe como un conector invisible pero poderoso. Su función principal es tomar la **salida estándar (stdout)** de un comando (el texto que normalmente verían en la terminal) y redirigirla para que sea la **entrada estándar (stdin)** de otro comando. Es decir, el resultado de la primera operación alimenta directamente a la segunda.

```
"comando1 | comando2"
```

Donde comando1 produce una salida que es directamente procesada por comando2, sin necesidad de archivos intermedios o interacción manual.

Una Analogía para Entenderlo Mejor

Piensen en el pipe como una **cinta transportadora de información**. Cada comando es una estación de trabajo:

- La primera estación (comando1) toma la materia prima y realiza un procesamiento inicial.
- En lugar de dejar el producto final en el suelo, lo coloca en la cinta transportadora (el pipe).
- La cinta lo lleva directamente a la siguiente estación (comando2), que lo toma como su propia materia prima para un procesamiento adicional.

Esta capacidad de encadenar comandos de manera fluida es lo que confiere al pipe su "magia", permitiendo construir flujos de trabajo complejos y potentes con gran facilidad. Es una herramienta indispensable para la administración de sistemas y el análisis de datos.

grep: El Filtro Universal de Texto

Dentro del arsenal de herramientas de línea de comandos de Linux, grep (del inglés "global regular expression print") se destaca como el especialista en la búsqueda y filtrado de texto. Es la navaja suiza que les permitirá encontrar agujas en pajes de información de manera rápida y eficiente.

¿Qué hace grep?

La función principal de grep es buscar líneas que contengan una palabra, una frase o un patrón específico dentro de uno o varios archivos. Es increíblemente potente porque puede trabajar con **expresiones regulares**, lo que le permite definir patrones de búsqueda muy complejos.

Uso Básico:

```
grep "palabra_a_buscar" nombre_del_archivo.txt
```

En este ejemplo, grep escaneará el archivo nombre_del_archivo.txt y mostrará en la terminal todas las líneas que contengan la "palabra_a_buscar". Por defecto, grep distingue entre mayúsculas y minúsculas, pero existen opciones para modificar este comportamiento.

Opciones Comunes y sus Superpoderes:

- -i: Ignorar mayúsculas y minúsculas (case-insensitive).
- -v: Invertir la búsqueda (mostrar líneas que NO contengan el patrón).
- -n: Mostrar el número de línea donde se encontró la coincidencia.
- -c: Contar cuántas líneas contienen el patrón.
- -r: Buscar recursivamente en directorios y subdirectorios.
- -l: Mostrar solo el nombre de los archivos que contienen el patrón.

Dominar grep es fundamental para cualquier tarea de administración de sistemas, desde la revisión de logs de seguridad hasta la depuración de scripts o la búsqueda de configuraciones específicas. Es un comando que usarán a diario una vez que descubran su verdadero potencial.

grep + Pipe = ¡Superpoder Analítico!

Aquí es donde la verdadera magia de la línea de comandos se manifiesta. La combinación de grep con el operador pipe | es una técnica fundamental que les permitirá filtrar y refinar la salida de **CUALQUIER** comando en tiempo real. Es como tener un control de calidad automático para la información que fluye por su terminal.

Filtrando Resultados en Vivo

Cuando se utiliza grep después de un pipe, ya no está buscando en un archivo estático, sino que está filtrando el flujo de datos que proviene del comando anterior. Esto es increíblemente potente para:

- **Reducir el Ruido:**
- **Análisis Rápido:**
- **Auditoría y Seguridad:**

Ejemplo 1: Explorando tu Historial

```
history | grep "sudo"
```

Este comando toma el historial completo de los comandos que has ejecutado (history) y lo pasa a grep. grep, a su vez, filtra esas líneas para mostrarte **sólo aquellos comandos que has ejecutado utilizando sudo**. Es excelente para revisar qué acciones con privilegios has realizado recientemente.

La combinación de grep y el pipe es un patrón de diseño fundamental en la línea de comandos. Una vez que dominen esta técnica, descubrirán un sinfín de posibilidades para manipular y analizar datos en sus sistemas.

Ejemplo 2: Buscando Configuraciones

```
ls -la /etc | grep ".conf"
```

Aquí, ls -la /etc lista en detalle todos los archivos y directorios dentro de /etc (donde se guardan muchas configuraciones del sistema). La salida de ls se canaliza a grep, que se encarga de mostrarte **únicamente las líneas que contengan la extensión .conf**. Esto te ayuda a encontrar rápidamente archivos de configuración sin tener que leer toda la lista.

Encontrando Secretos en /etc/services

Ahora que hemos comprendido el poder del pipe y la versatilidad de grep, es momento de poner manos a la obra con un ejercicio práctico que les permitirá aplicar estos conocimientos en un escenario real.

El Archivo /etc/services:

El archivo /etc/services es una pieza clave en cualquier sistema Linux. Contiene una lista de servicios de red bien conocidos y los números de puerto asociados a ellos. Es como una "agenda telefónica" de los puertos de red. Aunque no todos los servicios que corren en un sistema están listados aquí (muchos usan puertos dinámicos o personalizados), es un excelente punto de partida para entender la relación entre servicios y puertos.

Que haremos:

Utilizando la combinación de cat (para ver el contenido de un archivo) y grep (para filtrar), vamos a encontrar en qué puerto corren los servicios más comunes para la web, la conexión remota y la transferencia de archivos: http, ssh y ftp.

Paso 1: Visualizar el Contenido (aunque sea mucho):

Primero, simplemente usen cat para ver el contenido completo del archivo /etc/services. Verán que es un archivo bastante extenso.

```
cat /etc/services
```

Paso 2: ¡A Filtrar con grep!

Ahora, canalicen la salida de cat a grep para buscar específicamente cada servicio.

Buscar SSH:

```
cat /etc/services | grep "ssh"
```

Buscar HTTP:

```
cat /etc/services | grep "http"
```

Buscar FTP:

```
cat /etc/services | grep "ftp"
```

Al ejecutar estos comandos, observen la salida. Verán que grep les devuelve las líneas que contienen la palabra clave, y en esa misma línea, generalmente encontrarán el número de puerto asociado al servicio. Esta habilidad de "inspeccionar" archivos de configuración de forma eficiente es vital para la resolución de problemas y la administración de sistemas. ¡Felicitaciones, acaban de realizar su primera auditoría de puertos básica!

find: El Buscador de Archivos Proactivo

Mientras que grep se especializa en buscar contenido dentro de archivos, find es la herramienta definitiva para localizar archivos y directorios basándose en sus propiedades. Es como tener un bibliotecario incansable que puede rastrear cualquier documento en su vastísima biblioteca del sistema de archivos, sin importar cuán profundo esté oculto.

¿Para Qué Sirve find?

find es indispensable para:

- **Localizar archivos específicos:**
- **Organización:**
- **Administración de Espacio:**
- **Seguridad:**

Sintaxis Básica:

```
find [ruta] [criterios de búsqueda] [acción]
```

Los elementos clave son:

- **dónde_buscar (Path):** find comience su búsqueda. Puede ser . (directorio actual), / (todo el sistema), /home/usuario, etc.
- **criterio_de_búsqueda (Expression):** find debe buscar. Aquí es donde entra la magia:
 - -name "": Busca por nombre exacto o patrón de nombre.
 - -type : Busca por tipo (f para archivo, d para directorio, l para enlace simbólico).
 - -size : Busca por tamaño (c: bytes, w: palabras, b: bloques, k: KB, M: MB, G: GB).
 - -perm : Busca por permisos (ej. 777).
- **acción (Action - Opcional):** -print para imprimir, -delete para borrar, -exec comando {} \; para ejecutar un comando sobre cada resultado). Si no se especifica una acción, por defecto es -print (mostrar los resultados).

find es un comando extremadamente potente y puede ser peligroso si se utiliza con opciones de modificación (como -delete o -exec) sin la debida precaución. ¡Siempre prueben sus búsquedas antes de aplicar acciones irreversibles!

find en Acción: Ejemplos Prácticos y Comodines

Para comprender mejor la potencia de `find`, veamos algunos ejemplos concretos que ilustran cómo pueden utilizarse sus criterios de búsqueda para localizar archivos y directorios de manera efectiva.

Ejemplos de Uso:

Buscar archivos Python en el directorio actual y subdirectorios:

```
find . -name "*.py"
```

Aquí, `.` indica que la búsqueda comenzará en el directorio actual. La opción `-name "*.py"` busca cualquier archivo cuyo nombre termine con la extensión `.py`. El asterisco `*` es un **comodín** que significa "cualquier secuencia de caracteres". Es indispensable poner el patrón entre comillas para que el shell no intente expandir el comodín antes de pasarlo a `find`.

Buscar el archivo de configuración de SSHD en /etc:

```
find /etc -name "sshd_config"
```

Este comando buscará específicamente un archivo llamado `sshd_config` dentro del directorio `/etc` y todos sus subdirectorios. Este es un uso común para localizar archivos de configuración importantes.

Buscar todos los directorios que se llamen "tmp" en /var:

```
find /var -type d -name "tmp"
```

En este caso, `-type d` le dice a `find` que solo busque **directorios** (`d` por `directory`). Es útil para diferenciar entre un archivo y un directorio con el mismo nombre.

Buscar archivos mayores a 100MB en /home/usuario:

```
find /home/usuario -size +100M
```

El `+` antes de `100M` significa "más grande que". Si usáramos `-100M` sería "más pequeño que", y `100M` sería "exactamente 100MB". La `M` indica Megabytes.

Notas Importantes sobre Comodines:

Los comodines son caracteres especiales que se utilizan para representar patrones en los nombres de archivos o cadenas de texto. Los más comunes son:

- *** (Asterisco):** `*.txt` (todos los archivos `.txt`), `archivo*` (archivos que empiezan con "archivo").
- **? (Signo de Pregunta):** `doc?.txt` (`doc1.txt`, `docA.txt`, pero no `doc12.txt`).

La combinación de `find` con comodines y sus diversas opciones les brinda un control granular sobre la búsqueda de archivos, una habilidad crítica para la administración y la seguridad de sistemas.

Servicios en Linux

Detrás de la interfaz que ven y los comandos que ejecutan, un ecosistema de programas opera incansablemente para mantener su sistema Linux funcionando de manera óptima y ofrecer diversas funcionalidades. Estos son los **servicios**, a menudo referidos como "demonios" (daemons) en la jerga de Unix/Linux.

¿Qué es un Servicio (o Demonio)?

Un **servicio** es un tipo de programa que se ejecuta de forma continua en segundo plano, sin una interfaz de usuario directa. Su propósito es esperar y responder a solicitudes o realizar tareas predefinidas de manera autónoma. Son los cimientos invisibles que permiten que un servidor web sirva páginas, un servidor SSH permita conexiones remotas o una base de datos almacene y recupere información.

Características Clave de los Servicios:

- **Ejecución en Segundo Plano:**
- **Propósito Específico:**
- **Gestión Centralizada:**
- **Recursos:**

Ejemplos Comunes de Servicios:

- **Servidor Web (ej. Apache2, Nginx):**
- **Servicio SSH (Secure Shell):**
- **Servidor de Base de Datos (ej. MySQL, PostgreSQL):**
- **Servidor DNS (Domain Name System):**
- **Servicio de Impresión (ej. CUPS):**

Comprender qué servicios se están ejecutando en un sistema es crítico para la administración de sistemas, la resolución de problemas y, fundamentalmente, para la **ciberseguridad**. Un servicio mal configurado o no monitoreado puede convertirse en un vector de ataque, mientras que un servicio esencial detenido puede inhabilitar funciones vitales.

Chequeando el Servicio SSH

Para concluir nuestra exploración de hoy, vamos a aplicar los conocimientos sobre servicios y `systemctl` para una tarea muy común e importante: verificar el estado del servicio SSH (Secure Shell) en su máquina. El servicio SSH es fundamental para la administración remota segura de servidores y sistemas Linux.

Servicio SSH: Puerta de Conexión Remota Segura

El servicio SSH, a menudo implementado por el paquete `openssh-server`, permite establecer conexiones cifradas entre dos máquinas. Esto significa que pueden controlar su servidor Linux desde cualquier parte del mundo de forma segura, ejecutar comandos, transferir archivos, y realizar tareas de administración sin estar físicamente frente a él.

Nuestra Misión:

Vamos a comprobar si el servicio SSH está corriendo en su sistema. Si no lo está, lo iniciaremos.

Paso 1: Verificar el Estado del Servicio SSH

Ejecuten el siguiente comando en su terminal:

```
sudo systemctl status ssh
```

- Observen cuidadosamente la salida.
- Busquen la línea que dice `Active:`.
- Si dice `active (running)` y está en color verde, ¡excelente! El servicio SSH está funcionando correctamente.
- Si dice `inactive (dead)` o `failed`, significa que el servicio no está corriendo o hay un problema.

Paso 2: Iniciar el Servicio SSH (Si No Está Activo)

Si en el paso anterior el servicio no estaba activo, pueden iniciararlo con el siguiente comando:

```
sudo systemctl start ssh
```

No verán mucha salida si el comando se ejecuta correctamente. Después de iniciararlo, **vuelvan a ejecutar el comando** `sudo systemctl status ssh` para confirmar que ahora está `active (running)`.

Paso 3: Asegurar que Inicie con el Sistema (Opcional, pero Recomendado en Servidores)

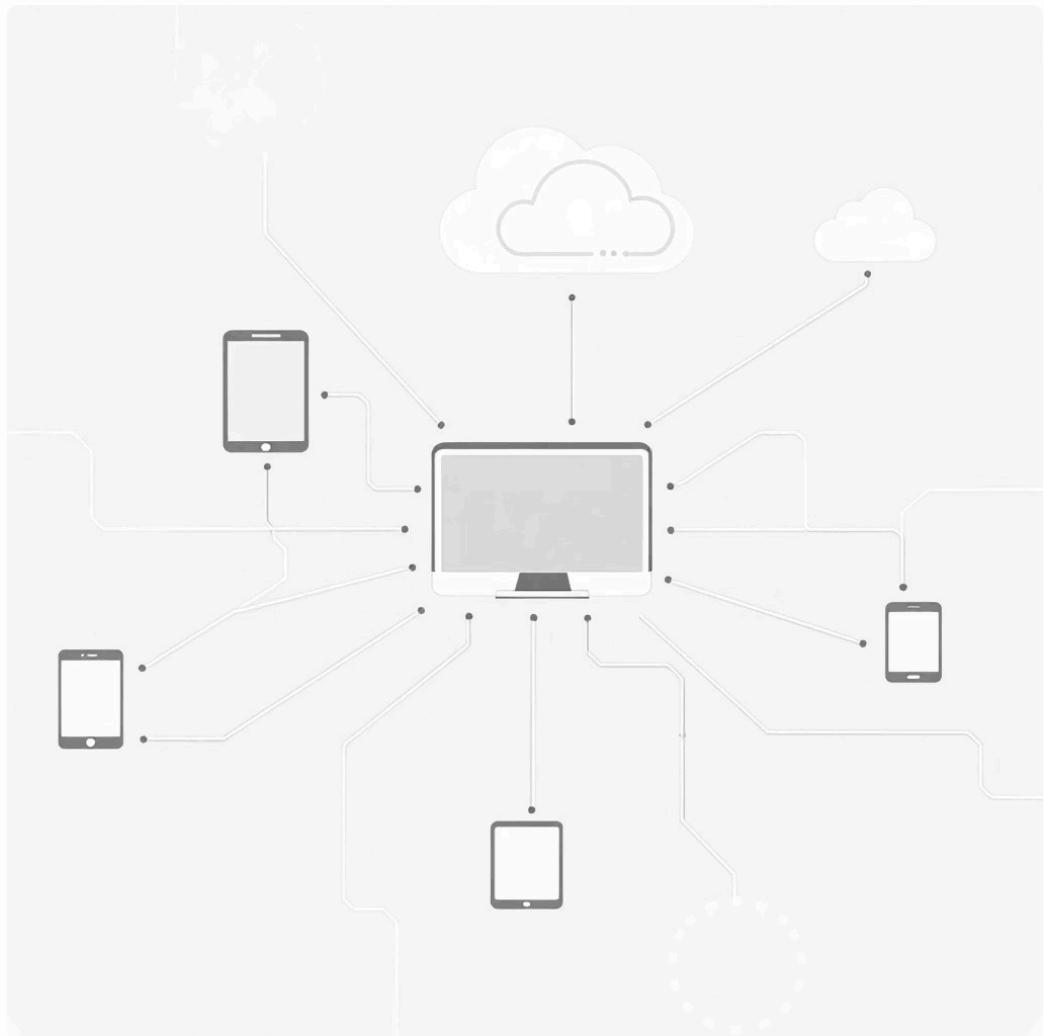
Si quieren que el servicio SSH se inicie automáticamente cada vez que enciendan su máquina (lo cual es crucial para un servidor), utilicen:

```
sudo systemctl enable ssh
```

Tu Máquina en el Mundo

En la era digital, la conectividad es la norma. Ningún sistema opera en completo aislamiento. Desde tu computadora personal hasta los servidores más robustos, todos se comunican constantemente con otros dispositivos en la red. Comprender cómo tu máquina interactúa con el mundo exterior es una habilidad fundamental, especialmente en el campo de la ciberseguridad.

Cuando hablamos de ciberseguridad, ver la comunicación de tu máquina no es solo una conveniencia, es una necesidad. Es el primer paso para identificar amenazas, diagnosticar problemas y asegurar tus sistemas.



Los comandos que exploraremos a continuación son la piedra angular para el mapeo de redes, la identificación de dispositivos y el análisis de vulnerabilidades. Te permitirán "ver" la infraestructura de red desde la perspectiva de tu propio sistema, un conocimiento invaluable para cualquier analista de seguridad.

ip addr

Cada dispositivo conectado a una red posee una identidad única, un "documento nacional de identidad" digital que lo distingue del resto. En el ámbito de las redes IP, esta identidad se materializa en la **dirección IP**. Es el equivalente a tu domicilio postal en el mundo real, permitiendo que los paquetes de datos lleguen a su destino correcto.

El comando `ip addr` (o su predecesor, `ifconfig`, todavía presente en algunos sistemas) es tu puerta de entrada para visualizar estas identidades. Al ejecutarlo en tu terminal, obtendrás una lista detallada de todas las interfaces de red presentes en tu sistema, junto con las direcciones IP asociadas a cada una.



Esta información es crucial para entender cómo tu máquina se conecta, qué redes tiene disponibles y cuál es su dirección para el resto del mundo digital. Es el punto de partida para cualquier tarea de configuración, diagnóstico o análisis de seguridad en red.

Desglosando la Información de ip addr

La salida del comando `ip addr` puede parecer abrumadora al principio, pero cada sección tiene un propósito claro. Aquí te ayudamos a interpretar la información más relevante:



lo: La Interfaz de Loopback

Esta es una interfaz virtual especial, que representa a tu propia máquina. Siempre tendrá la dirección `127.0.0.1` (o `::1` para IPv6). Se utiliza para la comunicación interna entre procesos en el mismo host, sin necesidad de salir a la red física. Si un programa intenta conectarse a `127.0.0.1`, está "hablándose a sí mismo".



eth0 o ens...: Tu Ethernet

Estas nomenclaturas suelen referirse a tu interfaz de red cableada, es decir, tu conexión a Internet a través de un cable Ethernet. Los nombres pueden variar (`eth0`, `eth1`, `ens33`, etc.) dependiendo de tu sistema operativo y hardware. Es donde se configura tu dirección IP principal para la red local.



wlan0: Tu Wi-Fi

Si tu máquina tiene conectividad inalámbrica, verás una interfaz como `wlan0` (o similar, como `wlp...`). Esta interfaz maneja la conexión a redes Wi-Fi. Las configuraciones de IP para esta interfaz serán las que obtenga de tu router inalámbrico.



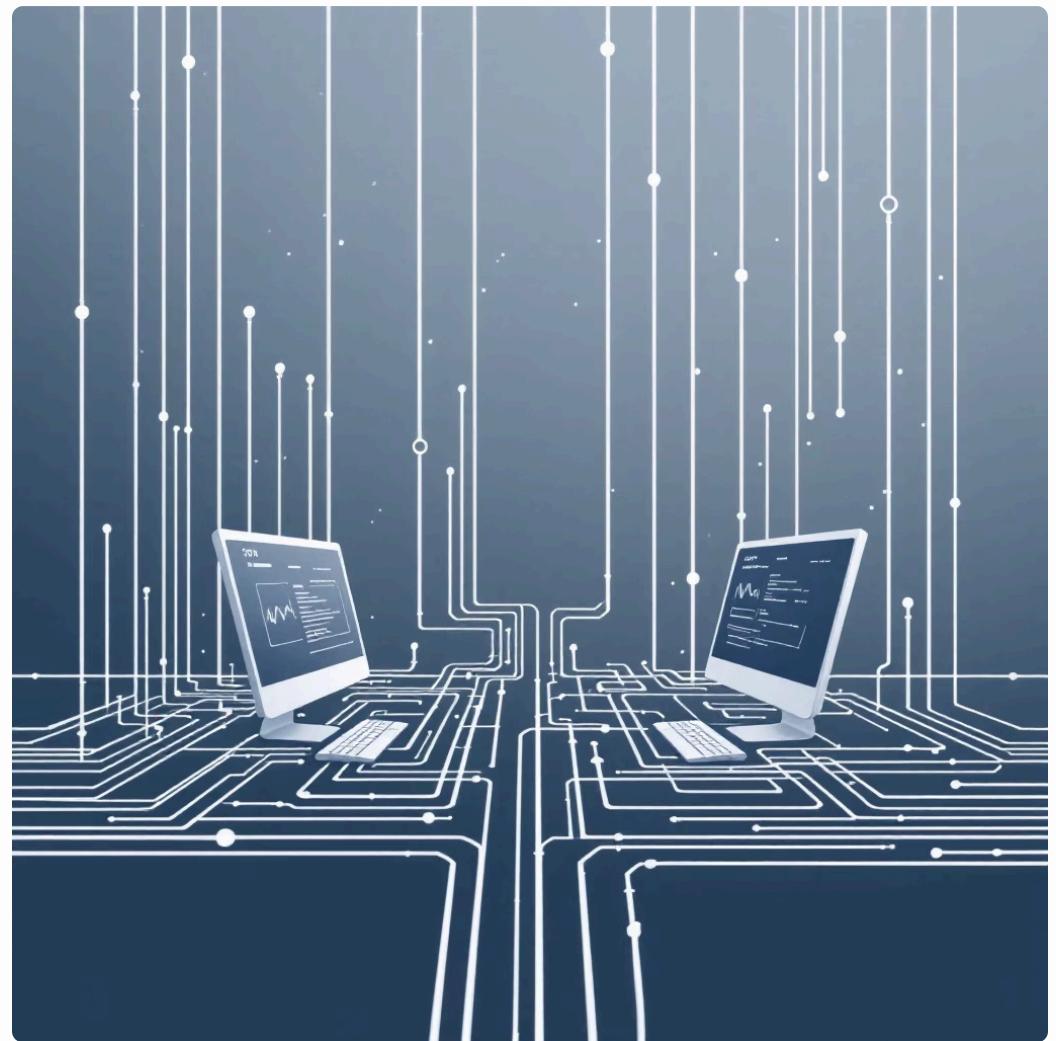
inet: Tu Dirección IPv4

Dentro de cada interfaz, busca la línea que comienza con `inet`. Este es el indicador de tu dirección IPv4, el formato de dirección IP más comúnmente utilizado. La dirección que aparece aquí es tu identificador único en la red a la que estás conectado. Si hay una línea `inet6`, esa es tu dirección IPv6.

ping:

El comando ping es una herramienta fundamental para el diagnóstico de redes, simple pero increíblemente potente. Su propósito es verificar la conectividad con un host remoto y medir el tiempo que tarda un paquete de datos en ir y volver. En esencia, le preguntas a otra máquina: "*¿Estás ahí? ¿Me escuchas?*", y esperas su respuesta.

Cuando ejecutas ping, tu sistema envía un pequeño paquete ICMP (Internet Control Message Protocol) conocido como "echo request" al destino especificado. Si el destino está "vivo" y es accesible, devolverá un "echo reply". ping te mostrará el tiempo de ida y vuelta (latencia) y si hubo pérdida de paquetes.



Práctica Sugerida:

Una forma común de probar tu conectividad a Internet es hacer ping a un servidor DNS público y muy conocido, como el de Google: ping 8.8.8.8

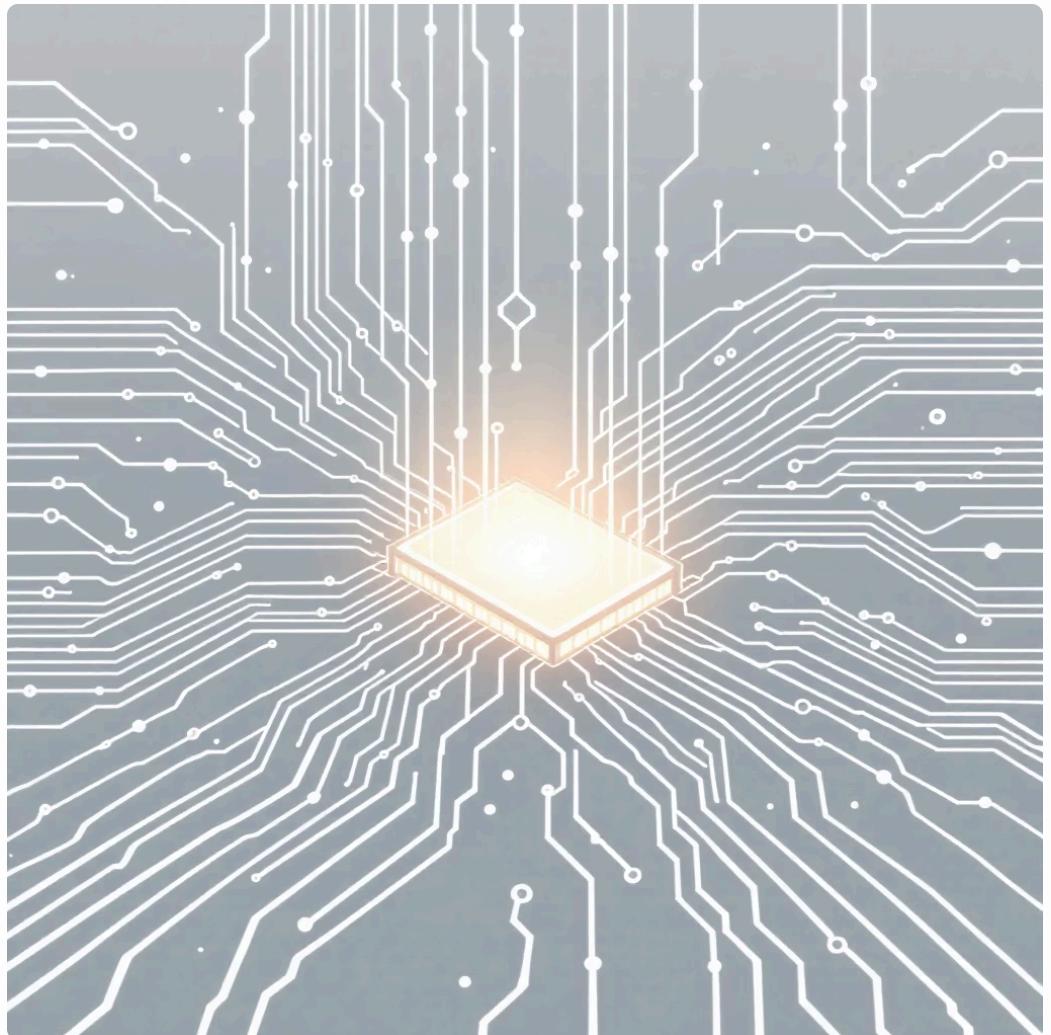
Observa cómo los paquetes son enviados y recibidos. Si ves respuestas, significa que tu máquina tiene conectividad con ese servidor. Si solo ves "Request timeout" o "Host unreachable", hay un problema de red.

Para detener el ping, simplemente presiona Ctrl+C en tu terminal.

netstat:

Si las direcciones IP son las direcciones de las casas en la red, los **puertos** son las puertas numeradas en esas casas. Cada aplicación o servicio en tu máquina utiliza uno o más puertos para comunicarse. Por ejemplo, un servidor web puede "escuchar" en el puerto 80 para el tráfico HTTP, o en el 443 para HTTPS.

El comando netstat (Network Statistics) es una herramienta esencial para la ciberseguridad y la administración de sistemas, ya que te permite "espiar" las conexiones de red activas, las tablas de enrutamiento y, lo que es más importante, los puertos que tu máquina tiene abiertos y "escuchando" conexiones entrantes.



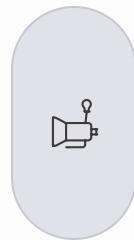
El comando clásico y más útil para ver qué puertos están abiertos y qué procesos los están utilizando es:

```
netstat -tulpn
```

Esta combinación de opciones te brinda una vista instantánea y completa de la actividad de red a nivel de puerto. Es indispensable para identificar servicios no autorizados, posibles puertas traseras o simplemente para verificar que tus aplicaciones estén funcionando correctamente.

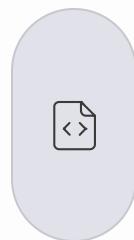
Desglosando netstat -tulpn:

Entender cada opción del comando `netstat -tulpn` es clave para aprovechar al máximo esta poderosa herramienta. Cada letra agrega una capa de información a la salida, permitiéndonos filtrar y visualizar exactamente lo que necesitamos:



`-t`: TCP

Muestra todas las conexiones y puertos **TCP** (Transmission Control Protocol). TCP es un protocolo orientado a la conexión, lo que significa que establece una sesión confiable entre dos puntos antes de enviar datos.



`-u`: UDP

Muestra todas las conexiones y puertos **UDP** (User Datagram Protocol). UDP es un protocolo sin conexión, más rápido pero menos confiable que TCP. Se usa para servicios donde la velocidad es más crítica que la integridad perfecta, como streaming de video o DNS.



`-l`: Listening

Filtre la salida para mostrar **solo los puertos que están "escuchando"** (listening). Esto significa que un servicio está esperando activamente conexiones entrantes en ese puerto. Es crucial para identificar servicios expuestos.



`-p`: Process

Muestra el **ID del proceso (PID)** y el **nombre del programa** asociado a cada conexión o puerto. Esta es una de las opciones más valiosas, ya que te permite saber qué aplicación específica está usando un puerto determinado.

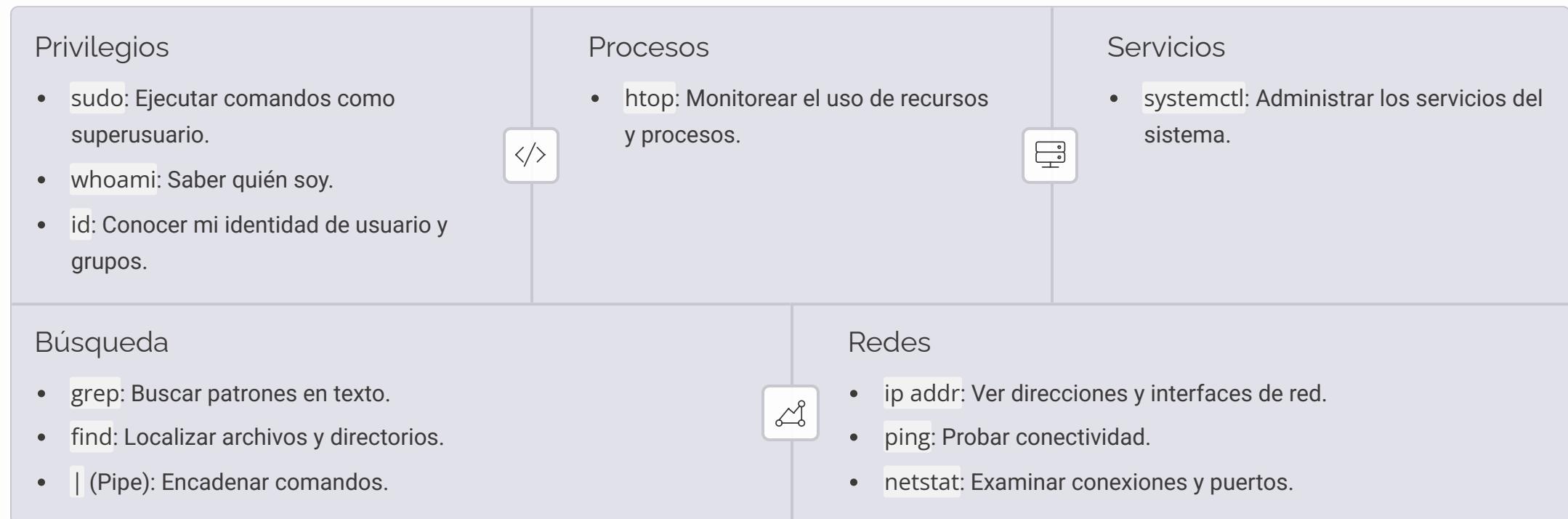


`-n`: Numeric

Muestra las direcciones IP y los **números de puerto en formato numérico**, en lugar de intentar resolverlos a nombres de host o nombres de servicio (ej. "80" en vez de "http"). Esto acelera la ejecución del comando y evita consultas DNS innecesarias.

Caja de Herramientas de la Clase 2

Hemos recorrido un camino fascinante, equipándote con una colección de comandos esenciales que forman la base de la administración de sistemas y la ciberseguridad en entornos Linux. Esta es tu caja de herramientas, lista para ser usada en cualquier escenario de diagnóstico, análisis o auditoría.



¡Reto Final de la Clase 2!



Ha llegado el momento de poner a prueba tus nuevas habilidades. Este reto integrará varios de los comandos que hemos aprendido, simulando una situación real que un analista de sistemas o ciberseguridad podría enfrentar.

La Misión del Analista

Contexto: Existe un servicio web secreto, no documentado, corriendo en esta máquina virtual o en el sistema que estás auditando. Tu tarea es descubrirlo y obtener información clave sobre su configuración.

Tu Misión (¡paso a paso!):

1. **Usa** `netstat`: Ejecuta el comando `netstat -tulpn` para identificar el puerto en el que está escuchando el servicio web. Busca un puerto inusual o un proceso que parezca un servidor web (como "apache2", "nginx" o "httpd"). Anota el número de puerto y el PID (ID del proceso) asociado.
2. **Usa** `find`: Sabiendo que los archivos de configuración suelen estar en el directorio `/etc`, utiliza `find /etc -name "*.conf*"` (o un patrón similar) y complementa con el nombre del proceso que identificaste en `netstat` para localizar su archivo de configuración principal. Por ejemplo, si el proceso es "nginx", podrías buscar `find /etc -name "nginx.conf"`.
3. **Usa** `grep`: Una vez que hayas encontrado el archivo de configuración, utiliza `grep` para buscar dentro de él la línea que define el `DocumentRoot`. El `DocumentRoot` es la carpeta principal donde se encuentran los archivos del sitio web servido. El comando sería algo como `grep -i "documentroot" /ruta/al/archivo/config`.

Este desafío integra la observación de redes con la búsqueda de archivos y el análisis de contenido, habilidades cruciales para la investigación forense digital y la auditoría de sistemas. ¡Mucho suerte!

¡Próximos Pasos!

i

- Para la próxima Clase:
 - En la próxima clase, profundizaremos en el mundo de las redes.
 - Exploraremos los modelos OSI/TCP, esenciales para entender la comunicación.
 - Mapearemos redes con Nmap, la herramienta de escaneo por excelencia.
 - Analizaremos tráfico con Wireshark, el microscopio de la red.
 - Y mucho más...

by Qb1t