

AWK

SISTEMAS OPERATIVOS

El comando `awk` en Linux se utiliza para procesar y manipular datos de texto en formato tabular. Proporciona una forma flexible de buscar, filtrar y transformar datos en archivos o en la entrada estándar. A continuación se presenta la estructura básica del comando `awk`:

```
`awk 'patrón {acción}' archivo`
```

Shell

- `patrón` especifica una condición o expresión regular que se debe cumplir para ejecutar la acción correspondiente.
- `acción` define lo que se debe hacer cuando el patrón coincide con una línea del archivo.
- `archivo` es el nombre del archivo de entrada. Si no se especifica, se usa la entrada estándar.

Algunas opciones y características clave de `awk` son las siguientes:

- `-F [carácter]` o `--field-separator [carácter]`: Establece el separador de campos en las líneas de entrada.
- `-v variable=valor`: Asigna un valor a una variable de `awk`.
- `BEGIN {acción}`: Define una acción que se ejecuta antes de procesar las líneas del archivo de entrada.
- `END {acción}`: Define una acción que se ejecuta después de procesar todas las líneas del archivo de entrada.
- Variables predefinidas: `NR` representa el número de registro actual, `NF` representa el número de campos en el registro actual, `$n` se refiere al campo `n`-ésimo del registro actual, donde `n` es un número

CAT

[[SISTEMAS OPERATIVOS]]

El comando `cat` se utiliza para mostrar el contenido de uno o más archivos en la salida estándar.

```
cat OPCIONES NOMBRE_DE_ARCHIVO
```

Shell

OPCIONES:

```
-b: Numerar las líneas que no están en blanco.  
-e: Mostrar el final de una línea (como $) y todos los caracteres no imprimibles.  
-n: Numerar todas las líneas de salida, comenzando por el 1.  
-s: Sustituir varias líneas en blanco por una sola línea en blanco.  
-t: Mostrar las tabulaciones como ^I.  
-v: Mostrar los caracteres no imprimibles.  
-A: Mostrar todos los caracteres, incluidos los no imprimibles.
```

WC

SISTEMAS OPERATIVOS

El comando `wc` se utiliza para contar palabras, líneas y caracteres en un archivo o en la entrada estándar. "wc" significa "word count" (conteo de palabras) en inglés.

```
wc OPCIONES NOMBRE_DE_ARCHIVO
```

Shell

OPCIONES:

```
-l: Muestra el número de líneas en el archivo.  
-w: Muestra el número de palabras en el archivo. Por defecto, se considera una palabra como una secuencia de caracteres delimitada por espacios en blanco.  
-c: Muestra el número de caracteres en el archivo.  
-nada: Si no se especifica ninguna opción, `wc` proporcionará un recuento completo que incluye líneas, palabras y caracteres en el archivo.
```

COMMAND

`command` en bash se utiliza para ejecutar un comando sin verse afectado por funciones del mismo nombre o alias que puedan existir en el entorno del script o del usuario. También se puede utilizar para comprobar la existencia de un comando.

Sintaxis básica:

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

Shell

Opciones importantes:

- `-v`: Imprime el nombre del comando antes de ejecutarlo.

Ejemplo:

```
if [ ! command -v dmidecode &> /dev/null ]; then
    echo "dmidecode no está instalado. Instalándolo..."
    apt-get update
    apt-get install dmidecode -y
fi
```

Shell

CRON

[[SISTEMAS OPERATIVOS]]

El comando `cron` es un programador de tareas en sistemas operativos Unix y Unix-like. Permite a los usuarios programar tareas para que se ejecuten automáticamente en momentos específicos, de forma recurrente o única.

• Estructura Básica:

```
cron [ OPCIONES ]
```

Shell

tiene los siguientes valores:

MINUTO HORA DAY(month) MONTH DAY(week)

Cualquier cosa anda a contab.guru

- La coma es para ponerle varios valores
- El guion es para un rango
- "/" es para un valor específico
- /10 (min) -> se va a ejecutar cada 10 minutos

Cuando utilizamos esto en el parcial, comentamos la línea para que no se ejecute en el momento que determinamos. No hace falta darle permisos de ejecución, por lo menos en el parcial.

• Uso Común:

- `cron -l`: Lista las tareas programadas para el usuario actual.
- `cron -r`: Elimina todas las tareas programadas del usuario actual.
- `cron -e`: Edita las tareas programadas del usuario actual.

DMESG

[[SISTEMAS OPERATIVOS]]

```
dmesg [opciones]
```

Shell

Descripción:

El comando `dmesg` muestra el buffer del kernel, proporcionando mensajes del sistema relacionados con el hardware, drivers y otros eventos del kernel.

Funciones principales:

- Muestra mensajes del kernel en tiempo real.
- Proporciona información de arranque, errores y eventos del kernel.

Opciones comunes:

- `-c`: Borra el buffer después de mostrar los mensajes.
- `-H`: Muestra mensajes en formato humano (más legible).
- `-T`: Muestra las marcas de tiempo en un formato más amigable.

DMIDECODE

[SISTEMAS OPERATIVOS]

`dmidecode` es una herramienta en sistemas operativos tipo Unix/Linux que proporciona información detallada sobre la configuración del hardware del sistema almacenada en la tabla DMI (Desktop Management Interface) de la BIOS. Aquí hay algunas de las banderas más utilizadas:

- 1. `-t, --type`
 - **Uso:** `dmidecode -t tipo`
 - Muestra información específica del tipo proporcionado. Por ejemplo, `dmidecode -t memory` para información sobre la memoria.
- 2. `-s, --string`
 - **Uso:** `dmidecode -s clave`
 - Muestra el valor de cadena asociado con la clave proporcionada. Por ejemplo, `dmidecode -s system-product-name` para obtener el nombre del producto del sistema.
- 3. `-t, --dump`
 - **Uso:** `dmidecode -t tipo`
 - Muestra una salida más detallada, incluyendo la información hexagonal para cada sección.
- 4. `-u, --unit-suffix`
 - **Uso:** `dmidecode -u`
 - Muestra las unidades de medida junto con los valores, por ejemplo, "MB", "GB", etc.
- 5. `--type`
 - **Uso:** `dmidecode --type tipo`
 - Similar a `-t, --type`, permite especificar el tipo de información deseada.
- 6. `-y, --type y`
 - **Uso:** `dmidecode -y tipo`
 - La bandera `-y` en `dmidecode` se utiliza para especificar el nivel de profundidad en la información que se muestra. Esta opción controla la cantidad de detalles que se incluyen en la salida, permitiendo elegir entre un formato más resumido o más detallado. Aquí tienes más detalles:
 - Muestra información específica del tipo proporcionado de manera más compacta.
 - Puede utilizarse para obtener una visión general rápida de la información del sistema sin la complejidad de la salida detallada.

Ejemplo de Uso:

```
sudo dmidecode -y memory # Muestra información resumida sobre la memoria
```

Shell

DU

[SISTEMAS OPERATIVOS]

El comando `du` en Linux (disk usage) se utiliza para obtener información sobre el uso de espacio en disco por parte de archivos y directorios. Proporciona una forma de mostrar el tamaño total de un directorio y sus subdirectorios, y puede ser útil para identificar qué archivos o directorios están ocupando más espacio en disco. A continuación se presenta la estructura básica del comando `du`:

```
du [opciones] [directorio]
```

Shell

- `opciones` son modificadores que controlan el comportamiento de `du`, como `-h` para mostrar los tamaños en un formato legible para humanos.
- `directorio` es el directorio que se desea analizar. Si no se especifica, se utiliza el directorio actual.

Algunas opciones y características clave de `du` son las siguientes:

- `-h` o `--human-readable`: Muestra los tamaños en un formato legible para humanos, utilizando unidades como "K" para kilobytes, "M" para megabytes, etc.
- `-s` o `--summarize`: Muestra solo el tamaño total del directorio, sin listar tamaños individuales de subdirectorios.
- `-c` o `--total`: Muestra el tamaño total acumulado de todos los directorios y archivos analizados.
- `-d [nivel]` o `--max-depth=[nivel]`: Limita el nivel de profundidad para mostrar información detallada de subdirectorios hasta el nivel especificado.
- `-a` o `--all`: Muestra tamaños de todos los archivos y directorios, incluyendo los ocultos (que comienzan con ".").
- `-x` o `--one-file-system`: Analiza solo el sistema de archivos donde se encuentra el directorio especificado y no cruza los límites de montaje.

EXECL

[[SISTEMAS OPERATIVOS]]

La función `exec1` en C se utiliza para ejecutar un programa en el espacio de memoria de un proceso existente, reemplazando el programa actual. Su sintaxis es:

```
int exec1(const char *path, const char *arg0, const char *arg1, ..., const char *argn, (char *)NULL);
```

C

- `path`: Ruta del programa que se va a ejecutar.
- `arg0`, `arg1`, ..., `argn`: Argumentos del programa. El primer argumento (`arg0`) generalmente es el nombre del programa.
- El último argumento debe ser `(char *)NULL`.

FDISK

[[SISTEMAS OPERATIVOS]]

`fdisk` sirve para realizar operaciones en discos y particiones. Permite visualizar, crear, modificar y eliminar particiones en dispositivos de almacenamiento como discos duros.

```
fdisk OPCIONES /dev/sdX
```

Shell

OPCIONES:

- Visualización de la tabla de particiones:

```
fdisk -l
```

Shell

Muestra información detallada de todas las particiones en todos los discos disponibles.

- Visualización de la tabla de particiones de un disco específico:

```
fdisk -l /dev/sdX
```

Shell

Muestra información detallada de las particiones en el disco especificado.

- Iniciar `fdisk` en un disco específico:

```
fdisk /dev/sdX
```

Shell

Abre `fdisk` para el disco especificado, permitiendo operaciones como crear, modificar o eliminar particiones.

- Operaciones dentro de `fdisk`:

- `p`: Muestra la tabla de particiones.
- `n`: Crea una nueva partición.
- `d`: Elimina una partición.
- `w`: Escribe los cambios en la tabla de particiones y sale.
- `q`: Sale sin escribir cambios.

- Tipos de particiones:

- `p`: Partición primaria.
- `e`: Partición extendida.
- `l`: Partición lógica.

- Formato de la partición:

- `t`: Cambia el tipo de partición.



FIND

SISTEMAS OPERATIVOS

El comando `find` se utiliza para buscar archivos que coincidan con un criterio específico en un directorio y sus subdirectorios.

`find` DIRECTORIO OPCIONES

Shell

(DIRECTORIO es la ruta del directorio en el que deseas buscar los archivos)

OPCIONES:

`-depth`: Procesa primero el directorio actual y luego sus subdirectorios.
`-maxdepth n`: Restringe la búsqueda a n niveles de directorios.
`-follow`: Procesa los directorios que se encuentran dentro de enlaces simbólicos.
`-name modelo`: Localiza los archivos cuyos nombres coinciden con el modelo propuesto.
`-ctime n`: Localiza los archivos creados hace n días.
`-user nombre_usuario`: Localiza los archivos pertenecientes al usuario especificado.
`-group nombre_grupo`: Localiza los archivos pertenecientes al grupo especificado.
`-path ruta`: Localiza los archivos cuya ruta coincide con el modelo propuesto.
`-perm modo`: Localiza los archivos con los permisos especificados.
`-size +nK`: Localiza los archivos cuyo tamaño (en kilobytes) es mayor al especificado.
`-print`: Imprime el nombre de los archivos encontrados.
`-exec comando [opciones] {} \;`: Ejecuta el comando especificado analizando el nombre del archivo encontrado.

FREE

SISTEMAS OPERATIVOS

El comando `free` se utiliza en sistemas Linux para mostrar información sobre el uso de la memoria del sistema. Aquí está un resumen del comando:

`free` [OPCIONES]

Shell

Opciones Comunes:

- `-h` o `--human`: Muestra las cifras en formato legible para humanos (por ejemplo, KB, MB, GB).
- `-t` o `--total`: Muestra una línea adicional con la suma total de la memoria.
- `-s` [SEGUNDOS] o `--seconds` [SEGUNDOS]: Actualiza la salida cada ciertos segundos.

Ejemplos:

- Mostrar el uso de memoria en formato legible:

`free -h`

Shell

- Actualizar la salida cada 3 segundos:

`free -s 3`

Shell

GCC

SISTEMAS OPERATIVOS

El comando `gcc` se utiliza para compilar programas escritos en el lenguaje de programación C.

`| gcc` OPCIONES ARCHIVO_DE_ENTRADA `-o` ARCHIVO_DE_SALIDA`

Shell

OPCIONES:

- `-c`: Compila el código fuente en un archivo objeto sin enlazar.
- `-o NOMBRE`: Especifica el nombre del archivo ejecutable de salida.
- `-Wall`: Activa mensajes de advertencia detallados.
- `-g`: Incluye información de depuración en el ejecutable.
- `-lm`: Enlaza la biblioteca matemática.

GREP

[[SISTEMAS OPERATIVOS]]

El comando `grep` se utiliza para buscar líneas que coincidan con un modelo de búsqueda en uno o más archivos. Es una herramienta poderosa para realizar búsquedas en texto.

```
grep OPCIONES MODELO ARCHIVO
```

Shell

OPCIONES:

- N: Muestra N líneas que contienen el modelo de búsqueda especificado.
- c: Muestra el número de líneas que contienen el modelo de búsqueda.
- f archivo: Lee las opciones de búsqueda desde el archivo especificado.
- F: Indica que se busca una cadena de texto exacta en lugar de un patrón.
- color: Resalta en color las cadenas de texto que coinciden con el modelo de búsqueda.
- n: Muestra el número de línea en el que se encuentra la palabra o cadena de texto buscada.
- i: Ignora mayúsculas y minúsculas al realizar la búsqueda.
- l: Muestra los nombres de los archivos que contienen el modelo de búsqueda.
- q: Devuelve el número de línea siguiente a aquellas en las que se encuentra el modelo de búsqueda.
- v: Muestra las líneas que no contienen el modelo de búsqueda.
- w: Busca líneas que contengan la palabra completa en lugar de partes de palabras.
- o: Muestra únicamente la cadena de texto que coincide con el modelo de búsqueda.

IFCONFIG

SISTEMAS OPERATIVOS

`ifconfig` es un comando que muestra y configura interfaces de red en sistemas Unix y Linux. Aquí está el resumen con el formato que me indicaste:

```
ifconfig [OPCIONES] [NOMBRE_INTERFAZ] [COMANDO]
```

Shell

OPCIONES:

- a o --all: Muestra información sobre todas las interfaces, incluso las inactivas.
- s o --short: Muestra una salida más corta y fácil de leer.

EJEMPLOS:

1. Mostrar información detallada de todas las interfaces:

```
ifconfig
```

Shell

2. Mostrar información detallada de una interfaz específica (por ejemplo, eth0):

```
ifconfig eth0
```

Shell

3. Mostrar información breve de todas las interfaces:

```
ifconfig -s
```

Shell

4. Mostrar información sobre todas las interfaces, incluyendo las inactivas:

```
ifconfig -a
```

Shell

LAST


[[SISTEMAS OPERATIVOS]]

El comando `last` muestra una lista de las sesiones de inicio y cierre de usuarios en el sistema. Proporciona información detallada sobre quién ha iniciado sesión, desde qué terminal o dirección IP, cuándo comenzó la sesión y cuándo finalizó.

```
last [OPCIONES] [NOMBRE_DE_USUARIO] [TERMINAL]
```

Shell

```
**OPCIONES:**  
- `-f ARCHIVO`: Especifica el archivo de registro a examinar (por defecto: `/var/log/wtmp`).  
- `-t FECHA`: Muestra las sesiones desde la fecha especificada.  
- `-s FECHA`: Muestra las sesiones hasta la fecha especificada.  
- `-x`: Muestra solo cierres de sesión.
```



LET

SISTEMAS OPERATIVOS

Se utiliza para realizar operaciones aritméticas dentro de un script de shell o en la línea de comandos. Se utiliza comúnmente para realizar cálculos y operaciones matemáticas simples en variables. `let` se utiliza para asignar y manipular valores numéricos en variables dentro de un script de shell.

```
let "variable = expresión"
```

Shell

- **Operaciones aritméticas:** Puedes realizar operaciones de suma, resta, multiplicación, división y otras operaciones aritméticas con `let`.

```
let "resultado = variable1 + variable2"  
let "resultado = variable1 - variable2"  
let "resultado = variable1 * variable2"  
let "resultado = variable1 / variable2"
```

Shell

- **Incremento y decremento:** Puedes incrementar o decrementar el valor de una variable.

```
let "variable++"  
let "variable--"
```

Shell

- **Expresiones complejas:** `let` permite el uso de expresiones más complejas para realizar cálculos.

```
let "resultado = (variable1 + variable2) * variable3"
```

Shell

- **Comparaciones numéricas:** Puedes utilizar `let` para realizar comparaciones numéricas y asignar el resultado a una variable.

```
let "comparacion = (variable1 > variable2)"
```

Shell

NETSTAT

[[SISTEMAS OPERATIVOS]]

```
netstat [opciones]
```

Shell

- `netstat` es una herramienta de línea de comandos que muestra información sobre la actividad de red en un sistema.

Uso Común:

- `netstat -a`: Muestra todas las conexiones y puertos, tanto TCP como UDP.
- `netstat -t`: Muestra las conexiones TCP.
- `netstat -u`: Muestra las conexiones UDP.
- `netstat -n`: Muestra las direcciones y números de puerto en formato numérico.
- `netstat -p`: Muestra los programas que utilizan las conexiones.

PGREP

[[SISTEMAS OPERATIVOS]]

Propósito: `pgrep` se utiliza para buscar procesos basados en sus nombres o atributos, y devuelve sus IDs de proceso (PIDs).

```
pgrep [opciones] nombre_proceso
```

Shell

Opciones comunes:

- `-u usuario`: Filtra por el nombre del usuario.
- `-x`: Coincide exactamente con el nombre del proceso.
- `-l`: Muestra el nombre del proceso junto con el PID.

▪ **Ejemplo de uso:**

```
pgrep -u $USER nombre_proceso
```

Shell

- Busca procesos del usuario actual cuyo nombre coincide con "nombre_proceso" y devuelve sus PIDs.

▪ **Funcionalidad adicional:**

- `pkill`: Se utiliza para enviar señales a procesos basados en sus nombres o atributos, similar a `kill` pero más conveniente para procesos seleccionados con `pgrep`.

PS

[[SISTEMAS OPERATIVOS]]

El comando `ps` en Unix/Linux se utiliza para mostrar información sobre los procesos en ejecución. Aquí hay una descripción básica y un ejemplo de su uso:

```
ps [opciones]
```

Shell

OPCIONES COMUNES:

- `-e`: Muestra información sobre todos los procesos.
- `-f`: Proporciona información detallada en formato completo.
- `-l`: Muestra una lista larga con información adicional.
- `-u usuario`: Muestra los procesos del usuario especificado.

Ejemplo de uso:

```
ps -ef
```

Shell

Este comando mostrará todos los procesos en ejecución con detalles completos.

COMANDO PS AUX

El comando `ps aux` es una variante extendida de `ps` que proporciona información más detallada sobre los procesos. Aquí está una descripción y un ejemplo de su uso:

```
ps aux
```

Shell

OPCIONES:

- `a`: Muestra todos los procesos.
- `u`: Muestra información del usuario.
- `x`: Muestra procesos que no están asociados con una terminal.

Ejemplo de uso:

```
ps aux
```

Shell

READ

[[SISTEMAS OPERATIVOS]]

El comando `read` se utiliza para leer la entrada del usuario o datos desde un archivo y asignarlos a una o más variables en Bash.

```
read OPCION VARIABLE
```

Shell

OPCIONES:

- r: Evita interpretar las secuencias de escape en la entrada. Se utiliza cuando se desea leer datos literalmente, sin procesar caracteres especiales.
- p prompt: Muestra un mensaje de solicitud al usuario antes de leer la entrada. El mensaje se especifica como argumento del prompt.
- a array: Lee la entrada en un array en lugar de en una variable individual. Los elementos se asignan a los índices consecutivos del array.
- n num: Limita la lectura a `num` caracteres en lugar de esperar una nueva línea completa. Esto es útil cuando se desea leer una entrada de longitud fija.
- d delim: Especifica un delimitador personalizado en lugar del carácter de nueva línea para terminar la lectura. El delimitador se utiliza para separar y terminar los campos de entrada.
- s: Hace que la entrada sea silenciosa, lo que significa que no se muestra en la pantalla mientras se lee. Esto es útil para leer contraseñas o información sensible.

SED

[[SISTEMAS OPERATIVOS]]

El comando `sed` en Linux (editor de secuencias) se utiliza para realizar transformaciones en el contenido de un archivo de texto o en la entrada estándar. Proporciona una forma flexible de buscar, reemplazar y editar texto utilizando expresiones regulares. A continuación se presenta la estructura básica del comando `sed`:

```
sed [opciones] 'comando' archivo
```

Shell

- opciones son modificadores que controlan el comportamiento de `sed`, como `-i` para realizar cambios en el archivo de entrada.
- comando especifica la operación que se realizará en las líneas del archivo. Puede ser un comando único o una secuencia de comandos separados por punto y coma (;).

Algunas opciones y características clave de `sed` son las siguientes:

- i [extensión] o --in-place[=extensión]: Modifica el archivo de entrada directamente. Opcionalmente, se puede proporcionar una extensión para crear una copia de seguridad del archivo original.
- e 'script' o --expression='script': Permite especificar un script con múltiples comandos separados por punto y coma (;).
- n o --quiet o --silent: Suprime la salida predeterminada de `sed` y solo muestra las líneas modificadas explícitamente.
- /patrón/acción o s/patrón/reemplazo/: Busca el patrón en las líneas y realiza la acción correspondiente, como reemplazar el patrón por un texto específico.
- p o print: Imprime las líneas coincidentes.

SEQ

[[SISTEMAS OPERATIVOS]]

Te devuelve numeros enteros, es como una secuencia.

seq donde empiezo + cada cuantos quiero q me devuelva + hasta que numero quiero que me tire

```
seq [valor_inicial] [valor_final] [incremento]
```

Shell

Este comando genera una secuencia de números desde `valor_inicial` hasta `valor_final` con un incremento especificado. Aquí hay algunas banderas comunes que se pueden usar con `seq`:

- s o --separator [separador]: Establece el separador entre los números de la secuencia. Por defecto, es un salto de línea.
- w o --equal-width: Asegura que todos los números en la secuencia tengan el mismo ancho, rellenando con ceros a la izquierda si es necesario.
- f o --format [formato]: Especifica un formato de salida para los números de la secuencia utilizando especificadores de formato de estilo printf.
- t o --terminator [terminador]: Establece el carácter de terminación después de la secuencia generada. Por defecto, es un salto de línea.
- h o --help: Muestra la ayuda y la información de uso del comando `seq`.

SPRINTF

SISTEMAS OPERATIVOS

La función `sprintf()` en C se utiliza para formatear una cadena y almacenarla en un búfer en lugar de imprimirla en la consola. Su sintaxis es similar a la función `printf()`.

Sintaxis:

```
#include <stdio.h>

int sprintf(char *buffer, const char *format, ...);
```

C

- `buffer`: Un puntero a un búfer donde se almacenará la cadena formateada.
- `format`: Una cadena de formato que especifica cómo se formatearán los datos.

Uso Básico:

```
#include <stdio.h>

int main() {
    char buffer[50];
    int numero = 42;

    sprintf(buffer, "El número es: %d", numero);

    // En este punto, buffer contiene "El número es: 42"

    return 0;
}
```

C

STAT

SISTEMAS OPERATIVOS

El comando "stat" se utiliza para mostrar información detallada sobre archivos y directorios en sistemas basados en Unix. Proporciona información como los atributos de acceso, modificación y cambio de un archivo, así como su tamaño y tipo.

```
stat [opciones] archivo
```

Shell

Opciones y banderas más comunes:

- `-c` `formato`: Especifica un formato de salida personalizado utilizando secuencias de formato. Puedes usar diversas secuencias para mostrar los atributos específicos que desees. Por ejemplo, `%s` muestra el tamaño del archivo, `%A` muestra los permisos en formato de texto, `%n` muestra el nombre del archivo, entre otros.
- `-f`: Muestra información de un sistema de archivos en lugar de un archivo individual. Es útil para obtener información sobre el sistema de archivos que contiene un archivo o directorio.
- `-t`: Muestra la marca de tiempo en un formato específico. Puedes utilizar esta opción junto con secuencias de formato para personalizar la salida de la marca de tiempo. Por ejemplo, `%y` muestra la marca de tiempo de la última modificación en formato legible para humanos.
- `-L`: Sigue enlaces simbólicos y muestra información sobre el archivo o directorio al que apuntan.
- `-r`: Muestra información en formato de solo lectura. Esto evita que se muestren caracteres especiales o secuencias de escape.
- `-s`: Muestra información resumida o estadísticas resumidas del archivo o directorio, como el número de bloques asignados, el número de enlaces y el tamaño en bytes.
- `-t`: Muestra información sobre la fecha y hora en que se modificaron ciertos atributos del archivo, como los permisos o la propiedad.

SYSTEM

La función `system()` en C se utiliza para ejecutar un comando del sistema desde dentro de un programa. `system()` ejecuta el comando especificado por la cadena `command` utilizando el intérprete de comandos del sistema operativo.

Su sintaxis es simple:

```
#include <stdlib.h>

int system(const char *command);
```

C

Valor de Retorno:

- Devuelve `-1` si no se puede ejecutar el comando o si hay un error en la invocación del intérprete de comandos. De lo contrario, devuelve el resultado del comando ejecutado.

SWAP

`swapoff` y `swapon` son comandos en sistemas basados en Unix/Linux que se utilizan para administrar el espacio de intercambio (swap) del sistema. Aquí tienes un resumen de cada uno:

1. `swapoff`:

- **Uso:**
`swapoff [opciones] [archivo/dispositivo_swap]`
- **Descripción:** `swapoff` se utiliza para desactivar el espacio de intercambio en un sistema. Puede desactivar toda la swap o dispositivos swap específicos.
- **Ejemplo:** `swapoff /dev/sda2` desactivaría el espacio de intercambio en el dispositivo `/dev/sda2`.
- **Nota:** Desactivar el espacio de intercambio puede llevar tiempo, ya que el sistema operativo debe mover todas las páginas en uso en el espacio de intercambio de vuelta a la memoria principal.

2. `swapon`:

- **Uso:**
`swapon [opciones] [archivo/dispositivo_swap]`
- **Descripción:** `swapon` se utiliza para activar el espacio de intercambio en un sistema. Puede activar toda la swap o dispositivos swap específicos.
- **Ejemplo:** `swapon /dev/sda2` activaría el espacio de intercambio en el dispositivo `/dev/sda2`.
- **Nota:** Activar el espacio de intercambio permite al sistema utilizar la memoria de intercambio para almacenar páginas de memoria que no se utilizan activamente.

`swapoff`:

- `-a o --all`: Desactiva todos los dispositivos swap especificados en `/etc/fstab`.
- `-v o --verbose`: Muestra información detallada durante la desactivación del espacio de intercambio.

`swapon`:

- `-a o --all`: Activa todos los dispositivos swap especificados en `/etc/fstab`.
- `-v o --verbose`: Muestra información detallada durante la activación del espacio de intercambio.
- `-p N o --priority=N`: Establece la prioridad del espacio de intercambio. Cuanto más alto sea el valor de N, mayor será la prioridad. El valor predeterminado es -1.
- `--show`: Muestra la información detallada sobre todos los espacios de intercambio activos.

TAR

[[SISTEMAS OPERATIVOS]]

Con el comando `tar`, puedes comprimir y descomprimir archivos.

```
tar OPCIONES **ARCHIVO NUEVO** **ARCHIVOS ANTIGUOS**
```

Shell

OPCIONES:

- c: Crea un nuevo archivo tar.
- z: Comprime el archivo tar usando Gzip, lo que reduce el tamaño total del archivo.tar.
- f: Especifica el nombre del archivo tar. Se utiliza junto con la opción
- c para forzar la creación del archivo.tar.
- x: Extrae y descomprime los directorios y archivos que se encuentran dentro del archivo.tar específico.
- v: Muestra en pantalla los pasos que se están realizando. Si no se incluye esta opción, se realizarán las mismas operaciones pero sin mostrar detalles por pantalla.
- j: Crea un archivo tar.bz2.

UNAME

****[[SISTEMAS OPERATIVOS]]****

El comando `uname` proporciona información del sistema operativo. Aquí tienes un resumen siguiendo el formato que me indicaste:

```
uname [OPCIONES]
```

Shell

OPCIONES:

- `-a` o `--all`: Muestra toda la información disponible.
- `-s` o `--kernel-name`: Muestra el nombre del kernel.
- `-n` o `--nodename`: Muestra el nombre del nodo de red.
- `-r` o `--kernel-release`: Muestra la versión del kernel.
- `-v` o `--kernel-version`: Muestra la versión del kernel de forma más detallada.
- `-m` o `--machine`: Muestra el tipo de máquina (hardware).
- `-p` o `--processor`: Muestra el tipo de procesador.
- `-i` o `--hardware-platform`: Muestra la plataforma de hardware.
- `-o` o `--operating-system`: Muestra el nombre del sistema operativo.
- `-n` Para el nombre de la maquina

VMSTAT

SISTEMAS OPERATIVOS

El comando `vmstat` proporciona estadísticas sobre el uso de memoria, la actividad de la CPU y el rendimiento general del sistema en intervalos regulares. Aquí tienes un resumen del comando:

```
vmstat [INTERVALO] [REPETICIONES]
```

Shell

Parámetros:

- **INTERVALO**: Especifica el intervalo de tiempo entre cada actualización de estadísticas.
 - **REPETICIONES**: Define la cantidad de actualizaciones que se mostrarán antes de que el comando finalice.
1. `vmstat -s`: Muestra estadísticas detalladas del sistema, incluyendo información sobre interrupciones, cambios de contexto, forks, entre otros.
 2. `vmstat -d`: Muestra estadísticas de actividad del disco, incluyendo la cantidad de operaciones de lectura y escritura realizadas.
 3. `vmstat -w`: Presenta las estadísticas en un formato más amplio y legible.

Ejemplos:

1. Mostrar estadísticas cada 2 segundos, 5 veces:

```
vmstat 2 5
```

Shell

Salida Típica:

```
procs -----memory----- --swap-- ----io---- -system-- -----cpu-----
 r  b   swpd   free   buff  cache   si   so    bi   bo    in   cs us sy id wa st
 0   0       0 1529212 246372 466092    0    0   30    9   186  375  3  2 94  1  0
```

Shell

Descripción de Columnas:

- procs: Información sobre procesos y ejecución de tareas.
- memory: Estadísticas de memoria (en kilobytes).
- swap: Uso de espacio de intercambio (swap).
- io: Estadísticas de operaciones de entrada/salida.
- system: Información del sistema.
- cpu: Utilización de la CPU.