



UT6.Configuración de sistemas operativos

Sistemas Informáticos

Profesorado:
Diego J. García
Rosa María Zapata Calle

Configuración de sistemas operativos.

Índice

1. Configuración de usuarios y grupos locales.
2. Usuarios y grupos predeterminados.
3. Seguridad de cuentas de usuario.
4. Seguridad de contraseñas.
5. Acceso a recursos. Permisos locales.
6. Servicios y procesos.
7. Comandos de sistemas libres y propietarios.
8. Herramientas de monitorización del sistema.



1. Configuración de usuarios y grupos

1. Gestión de grupos
2. Gestión de usuarios

1.Gestión de grupos

Tanto en Windows como en Linux, todos los usuarios del sistema pertenecen obligatoriamente a un grupo.

Este grupo “**obligatorio**” se denomina grupo **primario**.

Además puede pertenecer a otros muchos grupos llamados grupos **secundarios**.

La pertenencia a un grupo ayuda a mejorar la gestión de los recursos compartidos de forma que se pueda dar (o no) permisos de uso de los mismos a los miembros del grupo.

Una buena gestión de los grupos permite aumentar en gran medida la seguridad del sistema ya que permite restringir el uso de los diferentes recursos del mismo

1.Gestión de grupos

Para ver los grupos existentes en Linux debemos mostrar el fichero **/etc/group** con un comando de visualización de texto. Podemos usar **cat** para verlo entero por pantalla o también paginar la salida usando **more** o también **less**

En cada linea del fichero hay **un grupo** con información, en campos, separados por “:” siendo:

El primer campo el nombre del grupo

El segundo su contraseña (o una x indicando que la contraseña está en **/etc/gshadow**)

El tercero el **GID** o número identificador de grupo

El cuarto el listado separado por comas de los usuarios que pertenecen a este grupo.

Si queremos mostrar solo unos pocos campos usamos **cut**

Por ejemplo: **cut -d: -f1 /etc/group** para sacar solo los nombres de grupos. Las opciones significan: **-d** (delimitador) y **-f** (n.º de campo)

1.Gestión de grupos

Para crear grupos (como root) **groupadd <nuevo_grupo>**

Si ponemos varios grupos separados por comas podemos crear varios grupos al mismo tiempo. Un grupo siempre se debe crear antes de crear los usuarios pertenecientes a él.

Para comprobar si existe un grupo se puede usar un **filtro** o **tubería** (que envía la salida de un comando como entrada del siguiente) En este caso, podemos usar **grep** para buscar una cadena en un fichero:

cut -d: -f1 /etc/group | grep <nuevo_grupo>

Si nos muestra el nombre nuevo grupo es que se ha creado correctamente. Si no, puede que haya algún error en el comando anterior o en la creación del grupo.

1.Gestión de grupos

Si queremos ver (por comando), modificar o eliminar grupos debemos de operar con permisos de super usuario.

Para ver los grupos a los que pertenece un usuario usamos **groups [usuario]**

Para borrar grupos usamos **groupdel <grupo_a_borrar>**

Para modificar el GID de un grupo ejecutamos:

sudo groupmod -g <nuevoGID> <grupo>

Para modificar el nombre de un grupo utilizamos:

Sudo groupmod -n <grupo_nuevo> <grupo_actual>

Para modificar **temporalmente** el grupo primario de un usuario (lo modifica el usuario y debe pertenecer a él)

newgrp <nuevo_grupo_primario>

1.Gestión de grupos

Para gestionar los grupos en Windows, desde el menú inicio (con el botón izquierdo) accedemos a las **Herramientas administrativas de Windows**. Desde ahí entramos en la **Administración de equipos**. Al expandir el árbol de **Usuarios y grupos locales** podemos ver la carpeta de **Grupos** desde donde podremos ver los grupos existentes.

Pulsando con el botón derecho en la carpeta podemos crear un **Grupo nuevo...** y si pulsamos en cada uno de los grupos de la lista podremos **Agregar a grupo...** a un usuario, **Eliminar** un grupo, **Cambiar nombre** a un grupo o ver las **Propiedades** del grupo seleccionado.

1.Gestión de grupos

En la consola **PowerShell** de Windows los comandos son:

Crear un nuevo grupo local:

New-localgroup [-name] <nuevo_grupo> [-description <descripcion>]

Listar los grupos existentes:

Get-localgroup

Renombrar un grupo existente

Rename-localgroup -name <actual> -newname <nuevo>

Eliminar un grupo existente:

Remove-localgroup <nombredegrupo>

Listar los usuarios dentro de un grupo

Get-localgroupmember <nombredegrupo>

1.Gestión de usuarios

Actualmente, la gran mayoría de los sistemas operativos hacen uso de un sistema de usuarios que tendrán permiso para entrar al sistema y para realizar un listado más o menos amplio de tareas dentro del mismo.

Para poder entrar en un sistema operativo hay que estar registrado previamente en el mismo.

Una correcta y meticulosa gestión de los usuarios de un sistema incrementará en un alto grado su seguridad.

Igualmente, una mala gestión, ya sea por desconocimiento o por comodidad, puede generar accesos no autorizados llevando incluso a la pérdida de datos por parte de la empresa o institución a la que pertenezca el sistema

1.Gestión de usuarios

Para ver los usuarios en Linux debemos acceder al fichero **/etc/passwd** con un comando de lectura como puede ser **cat**, **more** o **less**.

Cada usuario está en una línea junto con información relativa al mismo separada por campos. Usando **cut**, podemos ver cada uno de los campos por separado.

Campo 1: Nombre del usuario

Campo 2: Contraseña o indicador de estar en **/etc/shadow**

Campo 3: UID o identificador de usuario

Campo 4: GID primario del usuario

Campo 5: Datos personales descriptivos del usuario

Campo 6: Directorio personal del usuario

Campo 7: Intérprete de comandos usado al iniciar sesión

1.Gestión de usuarios

Para crear nuevos usuarios en el sistema ejecutamos:

sudo useradd <nuevousuario>

El comando admite gran cantidad de opciones como pueden ser:

- D Muestra los valores por defecto que se usarán
- m crea el directorio home del usuario
- d <directorio_personal_del_nuevo_usuario>
- g <nombre_o_identificador_del_grupo_primario>
- G <lista_de_grupos_secundarios>
- u <Identificador_de_usuario_que_deseamos>
- s <shell_que_usará_el_usuario_al_conectarse>

1.Gestión de usuarios

Para modificar usuarios en el sistema ejecutamos:

sudo usermod <usuario>

El comando puede cambiar los parámetros indicados en la creación del mismo con las opciones anteriores.

Para bloquear / desbloquear un usuario sin eliminarlo:

sudo usermod -L <nombre_de_usuario>

sudo usermod -U <nombre_de_usuario>

Se puede cambiar el nombre de usuario utilizando:

sudo usermod -l <nuevo_nombre> <nombre_actual>

Para borrar un usuario utilizamos:

sudo userdel <usuario_del_sistema>

1.Gestión de usuarios

Para gestionar los usuarios con Windows accedemos desde el menú inicio (botón izquierdo) a “**Herramientas administrativas de Windows**”, “**Usuarios y grupos locales**” y a la carpeta “**Usuarios**”

Desde ahí aparece la lista de usuarios del sistema.

Al pulsar con el botón derecho sobre **Usuarios** podemos crear un **Usuario nuevo...** En el formulario que aparece podemos introducir sus datos personales y la contraseña.

Pulsando el botón derecho sobre un usuario se puede **Establecer contraseña...** nueva, **Eliminar** el usuario y consultar sus **Propiedades** como pueden ser su nombre, los grupos a los que pertenece, o el perfil del usuario

1.Gestión de usuarios

Los comandos de gestión de usuarios en PoewrShell son:

Crear un nuevo usuario:

New-localuser <nombre_de_usuario> (pedirá password)

Mostrar los usuarios del sistema:

Get-localuser

Renombrar un usuario local:

Rename-localuser -name <actual> -newname <nuevo>


Desactivar / Activar una cuenta de usuario:

Disable-localuser <usuario>

Enable-localuser <usuario>

Eliminar una cuenta de usuario:

Remove-localuser <usuario>



2. Usuarios y grupos predeterminados

1. Usuarios predeterminados
2. Grupos predeterminados

2. Usuarios predeterminados

- En Linux hay tres tipos de usuarios.
- El usuario **root**, también llamado superusuario o administrador tiene por **UID** el 0 y acceso a todos los archivos, directorios y privilegios del sistema de forma independiente de los permisos del sistema.
- Los usuarios **especiales**, también llamados cuentas del sistema, son cuentas que no inician sesión. Se crean en la instalación del sistema o de algunas aplicaciones. Dependiendo de la distribución se les asigna un **UID** entre 1 y 500 o entre 1 y 999
- Los usuarios **normales** se usan para usuarios del sistema. Solo tienen privilegios en su HOME. Su **UID** es superior al de los usuarios especiales

2. Usuarios predeterminados

- En Windows hay cinco tipos de usuarios por defecto.
- El usuario **administrador**, es el primero que se crea en la instalación de Windows. Tiene control total sobre los archivos, directorios, servicios y recursos del sistema. No se puede eliminar ni bloquear pero sí cambiar el nombre o deshabilitarla por seguridad. Puede ejecutar programas con permisos elevados sin necesidad de usar la opción **Ejecutar como administrador**
- El usuario **invitado** suele estar deshabilitado. Permite el inicio de sesión de forma puntual. Su contraseña está en blanco y permite acceso anónimo. Se aconseja dejar esta cuenta deshabilitada y con el mínimo nivel de permisos **salvo que sea estrictamente imprescindible.**

2. Usuarios predeterminados

- La cuenta **asistente de ayuda** se habilita al crear una sesión de asistencia remota. Se deshabilita si no hay solicitudes de asistencia pendientes.
- La cuenta **DefaultAccount**, también llamada **Cuenta administrada predeterminada del sistema (DSMA)** es un tipo de cuenta de usuario conocido. Es una cuenta neutra para el usuario que se puede usar para ejecutar programas independientes del usuario.
- La cuenta de **sistema** se usa por los servicios y procesos del sistema operativo que necesitan iniciar una sesión interna. No aparece en el administrador de usuarios y no se puede agregar a ningún grupo

2. Grupos predeterminados

- Igual que con los usuarios, en Linux existen tres tipos de grupos. El grupo **root** para poder obtener permisos elevados sin necesidad de ser root. Los grupos **especiales** referentes a ciertas operaciones que pueden realizar diferentes aplicaciones y/o servicios del sistema. Por último los grupos creados para los usuarios normales.
- En Windows, los principales grupos por defecto son:
- **Administradores**: sus miembros tienen control total
- **Operadores de copia de seguridad**: Sus miembros pueden crear y restaurar copias de seguridad
- **Operadores criptográficos**: Permite a sus miembros realizar operaciones criptográficas en el sistema

2. Grupos predeterminados

- **Operadores de configuración de red:** Sus miembros pueden configurar algunos aspectos de la red
- **Invitados:** De forma predeterminada (salvo la cuenta de invitado) tienen el mismo acceso que los miembros del grupo Usuarios
- **Usuarios:** Pueden ejecutar la mayoría de las aplicaciones No pueden hacer cambios accidentales o intencionados.
- **Usuarios avanzados:** Se incluyen para la compatibilidad con versiones anteriores. Poseen algunos derechos administrativos pero muy limitados.
- **Usuarios de administración remota:** sus miembros pueden acceder a los recursos de sistema utilizando protocolos de administración.



3.Seguridad de cuentas de usuario

1. Cuentas de usuarios Linux
2. Cuentas de usuario Windows

3. Cuentas de usuario Linux

- En Linux, el fichero de usuarios **/etc/passwd** se relaciona directamente con el fichero de passwords **/etc/shadow**
- Este segundo fichero contiene las contraseñas “**Hasheadas**” de los usuarios del sistema junto con datos relativos a ellas y al usuario al que pertenece.
- Este fichero pertenece al usuario **root** que puede leer y el único que puede editarlo. Además pertenece al grupo **shadow** aunque solo para leerlo.
- En caso de que, por algún error de seguridad, se pudiera leer las contraseñas están cifradas con una función **Hash** de una sola dirección que impide decodificarla de nuevo por lo que atacarlo necesitaría probar otras passwords

3. Cuentas de usuario Linux

- Aunque se han utilizado diferentes funciones Hash a lo largo de la historia de Linux, actualmente se suele usar la función resumen **SHA512** (se puede comprobar viendo que la cadena de la contraseña empieza por **\$6\$**) Si fuera un 1 sería MD5, un 2 blowfish y 5 SHA256. El siguiente valor entre dos símbolos de \$ se llama “sal”, es un conjunto de bytes aleatorios para complicar el cifrado
- El primer campo es el usuario al que pertenece la contraseña, el segundo la password cifrada, si tiene acceso, un asterisco si es un usuario especial sin poder iniciar sesión, o una exclamación si está deshabilitado
- El resto de campos son fechas referentes a la password en formato “**Epoch**” (días transcurridos desde 1-1-1970)

3. Cuentas de usuario Linux

- **Campo 3: Días desde que se cambió la contraseña.**
- **Campo 4: Mínimo de días para volver a cambiarla.**
- **Campo 5: Días durante los que la contraseña es válida.**
- **Campo 6: Días a los que el usuario será avisado de que la contraseña va a caducar y debe cambiarla.**
- **Campo 7: Días tras los que se va a deshabilitar la cuenta después de que caduque la contraseña.**
- **Campo 8: Días tras los que se deshabilita la cuenta.**
- **Estos campos se pueden modificar mediante el uso del comando `chage` cuya función es cambiar los periodos de uso de las cuentas y/o contraseñas. En muchos casos se definen por defecto mediante políticas**

3. Cuentas de usuario Windows

- **Windows utiliza el UAC**, Control de cuentas de usuario para evitar accesos no autorizados al sistema.
- Este sistema muestra el nombre del programa que va a realizar cambios en el sistema y que requiere de aprobación del administrador mostrando información del editor del programa y el origen del fichero ejecutable.
- Si la aplicación es confiable se puede autorizar su uso.
- Si el usuario que ejecuta el software no es administrador, pedirá una contraseña de un usuario con privilegios.
- Se reconoce el uso de este sistema de control **UAC** porque muestra un escudo azul y amarillo situado sobre el logotipo de la aplicación en un lugar visible

3. Cuentas de usuario Windows

- Para el almacenamiento de las contraseñas, Windows utiliza un sistema **SAM** Security Account Manager que contiene las cuentas de usuario, sus contraseñas cifradas, y su localización en el registro de Windows.
- También se pueden encontrar físicamente en el fichero **%SystemRoot%/system32/config/SAM**
- Dentro de este fichero, almacenado en binario, se almacenan los datos anteriormente indicados con las contraseñas cifradas mediante una función **Hash** que puede estar basada en el algoritmo **NTLM** (en versiones modernas) y/o **LM** (en versiones más modernas)
- El empleo de estos algoritmos se robusteció con SysKey aunque puede que no funcione desde julio de 2019



4. Seguridad de contraseñas

1. Contraseñas seguras

4. Contraseñas seguras

- A lo largo del tiempo se ha visto que cualquier sistema de cifrado es rompible de una forma u otra. En algunos casos porque existe alguna vulnerabilidad en el algoritmo que permita la reversibilidad del cifrado y en otros por fuerza bruta (probando cientos/miles/millones de contraseñas)
- La mejora del hardware (incremento de RAM y CPUs) ha hecho que se puedan probar muchas más contraseñas por segundo que hace unos años
- Además, existen listas de palabras (diccionarios) con las contraseñas más utilizadas a nivel mundial para que las combinaciones sean mucho menores a la hora de realizar un ataque por fuerza bruta.

4. Contraseñas seguras

- Para que nuestra contraseña sea segura debe cumplir un listado de requisitos. Cuantos más cumpla, más segura.
- **Originalidad:** Debe ser única. No debe haberse usado por otras personas y en cada aplicación debe ser distinta
- **Longitud:** Cuantos más caracteres tenga más seguridad
- **Aleatoriedad:** Cuantos más caracteres distintos tenga, mejor y si estos están desordenados será mucho más complicado de averiguar. A ser posible debe tener letras minúsculas, mayúsculas, números y caracteres especiales en diferentes partes de la clave
- **Caducidad:** La fortaleza del sistema de claves contra los ataques de fuerza bruta se consigue cambiando las contraseñas con una periodicidad lo más corta posible

4. Contraseñas seguras

- Recomendaciones para aumentar la seguridad:
- Nunca se debe utilizar la misma contraseña para más de una cuenta aunque sean diferentes aplicaciones
- Nunca se debe compartir una contraseña con otras personas
- Nunca se debe apuntar la contraseña
- Nunca comunicar la contraseña por teléfono, correo electrónico o mensajería instantánea
- Finalizar o desbloquear la sesión antes de dejar el equipo
- Cambiar la contraseña si se sospecha que ha sido comprometida. No es necesario que lo haya sido, la sola sospecha es motivo suficiente como para cambiarla



5. ACCESO A RECURSOS. PERMISOS LOCALES

1. Acceso a recursos
2. Permisos en Windows
3. Permisos en Linux

5. Acceso a recursos

- Para acceder a diferentes recursos del sistema operativo, como periféricos o ficheros y directorios, necesitamos tener permisos sobre ellos.
- Cada uno de los recursos tendrá unos permisos por defecto además de un usuario y/o un grupo al que pertenece.
- Estos permisos se pueden modificar dando o quitando permisos a más usuarios o grupos ya sea en el sistema local o a través de la compartición por la red.
- Una buena configuración de estos permisos aumentarán en gran medida la privacidad y por tanto la seguridad en el acceso a dicho recurso por los usuarios autorizados

5. Permisos en Windows

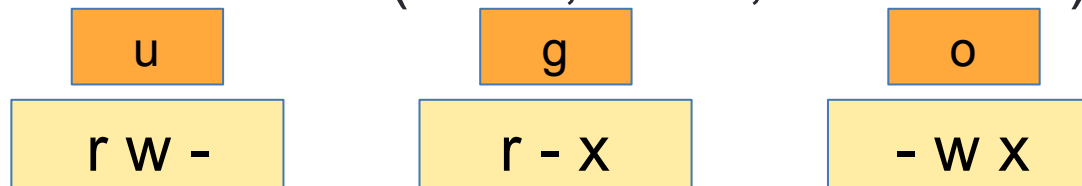
- En Windows cada usuario tiene permiso a sus ficheros y directorios. Para que otro usuario pueda acceder a dicho recurso debería compartirlo y darle permisos
- Los permisos que podemos darle a un fichero son **lectura, escritura, modificación, lectura y ejecución y permisos especiales** y, aparte de estos, a un directorio también se permite **mostrar el contenido de la carpeta**
- Para dar permisos debemos pulsar con el botón derecho en el recurso y elegir “**Propiedades**” En el formulario se debe elegir la pestaña **Compartir** para compartir la carpeta en la red y **seguridad** para indicar los usuarios y grupos específicos a los que se les quiere dar permiso. En el botón editar, se **Agregan** usuarios sus permisos

5. Permisos en Linux

- Los permisos en Linux se gestionan de forma diferente.
- Cada recurso tiene un usuario y un grupo propietario.
- Haciendo un listado extendido se puede ver que al principio de cada línea hay una serie de diez letras. La primera hace referencia al tipo de fichero y las otras nueve son tres grupos de tres permisos cada uno.
- Los tipos de ficheros son:
 - - → Un guión significa un fichero normal
 - d → directorio
 - l → Enlace simbólico (acceso directo)
 - b → Dispositivo de bloque
 - c → Dispositivo de carácter
 - p → Dispositivo de tubería (pipe)

5. Permisos en Linux

- En el caso de los nueve caracteres de los permisos se dividen en tres grupos (**U**usuario, **G**grupo, **O**tros) de tres permisos cada uno (**R**ead, **W**rite, **eX**ecution) de la forma:



- En cada uno de los tres grupos la letra se considera que se tienen permisos y un guión que no se tienen.
- Los permisos del **usuario** se refieren al usuario propietario del recurso (se determinan comparando su UID)
- Los permisos del **grupo** se refieren a los usuarios que pertenecen al grupo propietario del recurso excepto al usuario propietario (se determinan comparando su GID)

5. Permisos en Linux para ficheros

- El permiso de lectura indica que se puede ver/leer su contenido (con cat, more, head, tail, cut, grep, less, ...)
- El permiso de escritura indica que podemos modificar su contenido, añadiendo, cambiando o borrando parte del mismo con un editor. El hecho de tener permiso de “borrado” no quiere decir que podamos borrar el fichero sino solamente su contenido ya que el permiso borrar/crear/mover/renombrar ficheros se le debe dar al directorio que los va a contener
- El permiso de ejecución indica que podemos ejecutar el fichero como si fuera un programa. Solamente podremos ejecutar un programa si tenemos permisos de ejecución

5. Permisos en Linux para directorios

- El permiso de lectura indica que se puede ver/leer/listar su contenido (con ls)
- El permiso de escritura indica que podemos crear borrar, mover, renombrar ficheros y/o directorios contenidos en él. Se puede dar el caso de que se tenga permiso de escritura en un directorio pero no en los ficheros contenidos. Se daría la paradoja de que podríamos borrar el fichero pero no modificar su contenido
- El permiso de ejecución indica que podemos entrar dentro del directorio utilizando el comando cd

5. Permisos en Linux

- Para cambiar los permisos de un fichero o directorio en Linux debemos ser el propietario del mismo o ser root
- El comando **chmod <lista_de_permisos> <fichero>** permite cambiar estos permisos de dos formas diferentes:
- La forma simbólica divide el parámetro “**lista_permisos**” en tres partes. La primera es los usuarios a los que afecta siendo “**u**” – usuario, “**g**” – grupo, “**o**” – otros y “**a**” – todos. La segunda es si se dan “**+**” o se quitan “**-**” permisos. La tercera parte es la lista de permisos que se modifican siendo “**r**” – lectura, “**w**” – escritura y “**x**” – ejecución
- Por ejemplo: **chmod uo+w,g-x fichero** daría permiso de escritura al usuario y a otros y quitaría el permiso de ejecución al grupo

5. Permisos en Linux

- La otra forma de especificar la lista de permisos es la numérica. Presupone los permisos de forma binaria tal que, si hay permiso es un **1** y si no lo hay es un **0**
- Puesto que hay tres grupos de permisos se dividen cada grupo en números en base octal del **0** al **7**
- Por ejemplo: **rw--wxr-x** se traduce en **110011101** que separados en 3 grupos y pasados a octal vemos que se quedaría como **635**
- El comando **chmod 542 fichero** asignaría los permisos de lectura y ejecución al usuario, el de lectura al grupo y el de escritura al resto eliminando el resto de permisos

5. Permisos en Linux

- Por motivos de seguridad, el único que puede cambiar el propietario de un fichero es el usuario **root**.
- Para cambiar el propietario utilizamos el comando:

chown <nuevo_usuario> <fichero>

- Sí se permite cambiar el grupo por el usuario propietario siempre que sea a otro grupo del que sea miembro. Si no lo es, debe cambiarlo también el root. Para ello se usa:

chgrp <nuevo_grupo> <fichero>

- Es posible cambiar tanto el usuario como el grupo propietarios a la vez usando chown de la siguiente forma:

chown <nuevo_usuario>[:grupo] <fichero>



6. Servicios y procesos

1. Procesos.
2. Estados de los procesos.
3. Servicios y demonios.

6. Procesos

- Un **proceso** es un programa en ejecución.
- Para poder ejecutar un programa, necesitamos colocar el proceso por completo en memoria RAM. Si no hubiera memoria suficiente se usaría memoria virtual en el disco
- Además, para poder ejecutar este proceso se necesita que tenga acceso a los recursos que necesita (impresora, teclado, ratón, monitor,...) sobre todo la CPU. Debido a los algoritmos de gestión de procesos puede que nuestro proceso tarde en empezar y, en muchos casos, se detenga para dejar paso a otros (multitarea) mientras espera en una **cola de procesos**.
- Si hubiera varios procesos intentando usar un mismo recurso al mismo tiempo se produciría un **interbloqueo**

6. Estados de los procesos

- Cada proceso, además de las instrucciones y datos del programa del que proceden tienen un “**contexto**” que son una serie de variables que indican su estado, variables creadas, hilos,... Cuando un proceso pasa de la CPU a la cola o viceversa se produce un “**cambio de contexto**”
- Los estados que puede tener un proceso son:
 - **Preparado**: Se ha creado su contexto y está en cola para iniciarse en cuanto haya posibilidad
 - **R – Corriendo**: El proceso está ejecutándose en CPU
 - **D – Espera ininterrumpible**: El proceso está en la cola esperando una operación de entrada/salida desde/ hasta algún dispositivo

6. Estados de los procesos

- **S – Espera interrumpible:** El proceso está en cola esperando a que se produzca algún evento en el sistema por ejemplo, que el planificador de procesos indique que es su turno y puede entrar en la CPU
- **T – Parado:** El proceso se ha parado por alguna razón, por ejemplo porque se ha enviado una señal de parada
- **Z – Zombie o difunto:** El proceso ha terminado pero todavía no se ha eliminado de la tabla de procesos ya que la información que conforma su **contexto** todavía no se ha liberado y por tanto se puede detectar todavía
- Por norma general los procesos no necesitan de una gestión externa salvo que haya problemas en el sistema. Es ahí donde entraríamos nosotros a administrarlos

6. Estados de los procesos

- Para ver los procesos en Windows pulsamos la combinación de teclas **Ctrl + Alt + Supr**, nos aparecerá una pantalla azul con un menú donde debemos elegir el **Administrador de tareas**. Inicialmente solamente nos aparecen los procesos que hemos arrancado nosotros
- Podemos pulsar en **Más detalles** y ver que además de nuestros procesos hay muchos más “**en segundo plano**” que corresponden generalmente al propio sistema o a programas que se ejecutan sin interfaz gráfica
- Si hay algún proceso que decidimos que no debe seguir ejecutándose por alguna razón, lo **seleccionamos** y, a continuación, pulsamos el botón “**Finalizar tarea**”

6. Estados de los procesos

- Para ver los procesos en Linux, en la consola, ejecutamos el comando **ps** y nos aparecerán los procesos que hemos lanzado nosotros incluida la **bash** que se ejecuta al iniciar la sesión
- Si queremos ver una lista más completa podemos usar: **ps aux** que nos muestra un listado de todos los procesos del sistema indicando por columnas, entre otros datos:
 - El usuario que lo ha lanzado
 - El **PID**, un entero identificador único de proceso
 - El porcentaje de CPU y de RAM que utiliza
 - El estado del proceso
 - La hora de inicio y el tiempo de uso de CPU
 - El comando usado para lanzarlo

6. Estados de los procesos

- El comando **top** nos permite ver en tiempo “real” con una actualización cada 3 segundos el listado de procesos ordenados de más a menos uso de los recursos principales como son CPU y RAM. Además nos ofrece datos como el **PID** o una estadística sobre los procesos
- Si pulsamos la **q** salimos de la consola de top y con **k** podemos enviar una señal a un proceso sabiendo su PID
- Los procesos pueden cambiar de estado de forma manual si les **enviamos señales**. Una señal es un indicador del nuevo estado que queremos que tenga el proceso.
- **kill -<señal> <PID>** envía el nuevo estado a un proceso concreto pudiendo pararlo, terminarlo, matarlo,... se puede ver una lista de señales usando el comando **man**

6. Estados de los procesos

- Si estamos ejecutando un proceso activo y pulsamos la combinación de teclas **Ctrl-c** lo terminaremos, sin embargo si pulsamos la combinación **Ctrl-z** lo pararemos y se enviará al segundo plano.
- Usando el comando **jobs** podemos ver los procesos en segundo plano, su estado y un **identificador de trabajo**
- Si ejecutamos el comando **bg <id_trabajo>** reanudaremos el trabajo pero esta vez en segundo plano. Nosotros podemos seguir usando la consola y ejecutando comandos. Si muestra mensajes por pantalla los seguiremos viendo y pueden llegar a molestar. En algunos casos es interesante redireccionar la salida con **>** a **/dev/null**
- Con **fg <id_trabajo>** traemos un trabajo al primer plano

6. Servicios y demonios

- Los **servicios** (en Windows) y **demonios** (en Linux) son procesos especiales ya que no son interactivos.
- Se ejecutan en segundo plano y no son controlados por el usuario si no es el root para enviarles alguna señal
- Suelen ser **persistentes**, es decir, tratan de reiniciarse si por alguna razón se les ha matado de forma “brusca”
- No disponen de interfaz interactiva con el usuario.
- No usan la entrada y salida estándar (teclado y monitor) para comunicar errores sino que los almacenan en archivos de log o registros de eventos del sistema.
- Suelen ofrecer un “servicio” al cliente que lo solicita como puede ser un servidor http, ftp, email, ssh,...



7. Comandos de sistemas libres y propietarios

1. Otros comandos

7 Otros comandos

- Es muy común en cualquier sistema operativo, sobre todo cuando se usan comandos que no conocemos la sintaxis exacta o la lista de opciones y argumentos que podemos utilizar. Por ello se crearon las páginas **man** que nos muestran información detallada del comando, librería o formato de fichero estamos utilizando. El uso sería utilizando: **man <comando>** En caso de querer buscar información relativa a comandos usando una palabra clave se puede usar la opción **-k <palabra_clave>** y nos salen comandos relacionados con ella
- Tras abrir un fichero de texto (o la salida de un comando anterior) de gran tamaño es muy probable que queramos “encontrar” un término concreto, por ejemplo una palabra “clave”. El comando **grep <cadena_de_texto>**

7 Otros comandos

- Puede ser que tengamos que indicar la línea en la que se encuentra cierto texto buscado, por ejemplo un error, o una modificación que haya que hacer el fichero. Para facilitar la tarea podemos usar **nl <ruta_al_fichero>** que numera las líneas de texto secuencialmente.
- También es posible querer saber el número de líneas, palabras y/o caracteres que tiene un fichero de texto, para ello usamos el comando **wc <ruta_al_fichero>**
- En Linux, se utilizan en gran variedad de ocasiones los “enlaces” que son similares a lo que en Windows se denomina “acceso directo”. Hay dos tipos de enlaces. Los enlaces duros/fuertes/hard/no simbólicos y los enlaces blandos/soft/simbólicos

7 Otros comandos

- El enlace duro crea otro acceso al fichero de forma que si se borra el fichero original, los datos siguen existiendo y pudiendo acceder a ellos a través de sus enlaces duros. Se crean usando **ln <enlace_duro> <ruta_al_fichero>**
- El enlace blando es similar pero se parece más a los accesos directos de Windows ya que si se borra el fichero original, dejaría de funcionar el enlace también. Son más comunes que los enlaces fuertes. Los enlaces simbólicos se crean usando **ln -s <enlace_duro> <ruta_al_fichero>**
- En todas las distribuciones nos encontramos, al menos, con un editor de textos. Uno de lo más antiguos es **vi** que suele venir instalado por defecto en todas. Es un editor por comandos y que, en casos extremos puede ser la única solución para recuperar/reconfigurar el sistema.

7 Otros comandos

- Puede ser que, desde la consola de Linux queramos descargarnos un fichero (o conjuntos de ficheros) o incluso una lista de URLs completas. Aunque existen varios comandos que se pueden utilizar para ellos, uno de los más conocidos es **wget** que se puede usar **wget <protocolo><URL_a_la_que_se_quiere_acceder>**
- Otro comando que, en casos determinados nos puede ser útil es **sort** que nos ordena un fichero de texto alfabéticamente por el primer campo, aunque se puede indicar que lo ordene de forma numérica y por cualquiera de los demás campos

7 Otros comandos

- Algunos comandos que nos pueden ser útiles a la hora de encontrar ficheros y/o directorios en el sistema son los comandos de búsqueda `locate`, `whereis`, `which` y `find`.
- El comando **`locate <nombre_de_fichero>`** busca un documento que se haya creado en el sistema. Utiliza una base de datos que se actualiza diariamente. En caso de que sepamos que el fichero existe y no se encontrase podemos actualizarla con el comando **`sudo updatedb`**
- El comando **`whereis <fichero binario>`** nos permite buscar ficheros ejecutables y ficheros relacionados que se encuentren dentro de la ruta `$PATH`
- Usar **`which <fichero binario>`** produce similar resultado

7 Otros comandos

- El comando **find**, es más potente que los anteriores.
- Su sintaxis es **find <path_inicial> <tipo_de_busqueda> [acción_a_llevar_a_cabo_en_caso_de_éxito]**
- El path inicial nos indica en qué rama o ramas del sistema de ficheros debe empezar a buscar
- Los tipos de búsqueda pueden ser por **-name**, por **-user**, por **-group**, por **-mmin** (minutos que hace que se han modificado), por **-size**, entre otros modificadores. En los dos últimos casos podemos usar los símbolos "+" (mayor que) o "-" (menor que) o nada (valor exacto)
- Como acciones se pueden usar **-exec <comando>** para ejecutar un comando sobre los resultados obtenidos.

7 Otros comandos

- La salida estándar de los comandos es por la pantalla. Si queremos enviarla en su lugar a otra salida, normalmente un fichero usamos el símbolo: **> <fichero_de_salida>**
- La redirección anterior crea el fichero si no existe y, si existe, borra su contenido e inserta la salida del comando
- Si queremos **añadir** la salida del comando al fichero sin borrar su contenido anterior, se usa **>> <fichero>**
- La entrada estándar es el teclado. Si, en su lugar, queremos enviar el contenido de un fichero como entrada a un comando utilizamos el símbolo: **< <ficheroentrada>**
- Los errores se muestran por defecto en pantalla. Para redireccionar los errores usamos: **2> <fichero_de_error>**

7 Otros comandos

- En los casos anteriores la salida de los comandos se enviaba a un dispositivo diferente del estándar. Normalmente la salida se almacena en un fichero.
- Igualmente la entrada estándar (teclado) se puede sustituir por un fichero con datos en texto plano.
- En algunos casos se pretende utilizar la salida de un comando como entrada de otro, para ello se usan las tuberías o pipes utilizando el símbolo |
- La forma de utilizar una tubería es:
<comando1_que_genera_resultados> |
<comando2_que_procesa_la_salida_del_comando1>

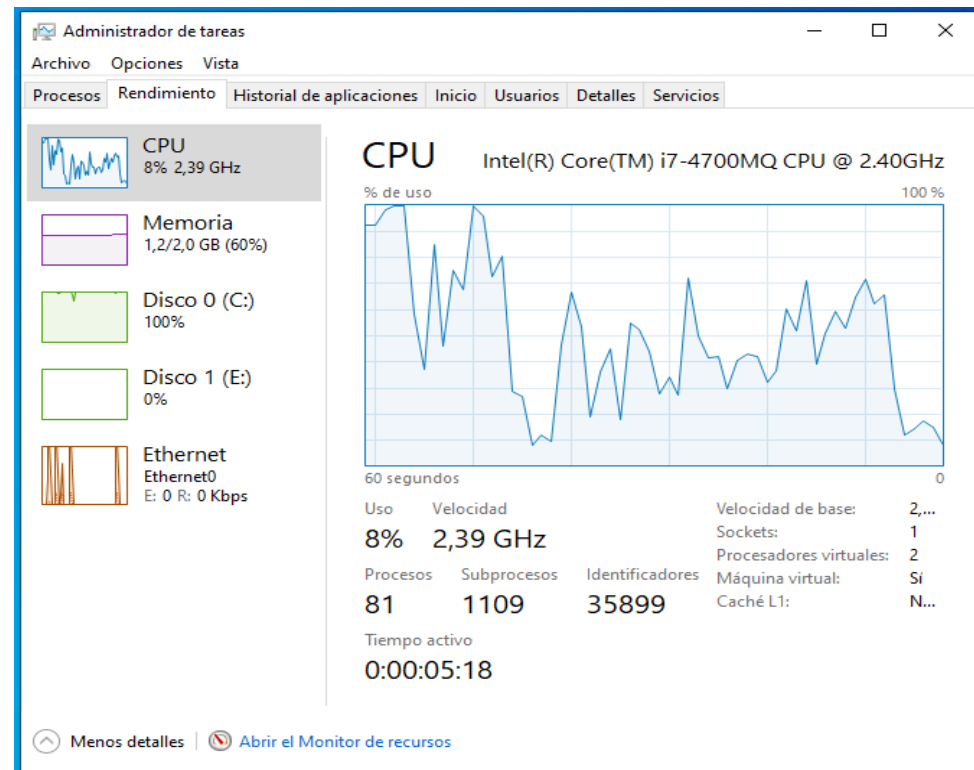


8. Herramientas de monitorización del sistema

1. Monitorización de la CPU
2. Monitorización de la RAM
3. Monitorización de la red
4. Monitorización del disco
5. Monitorización del sistema. Utilizando comandos

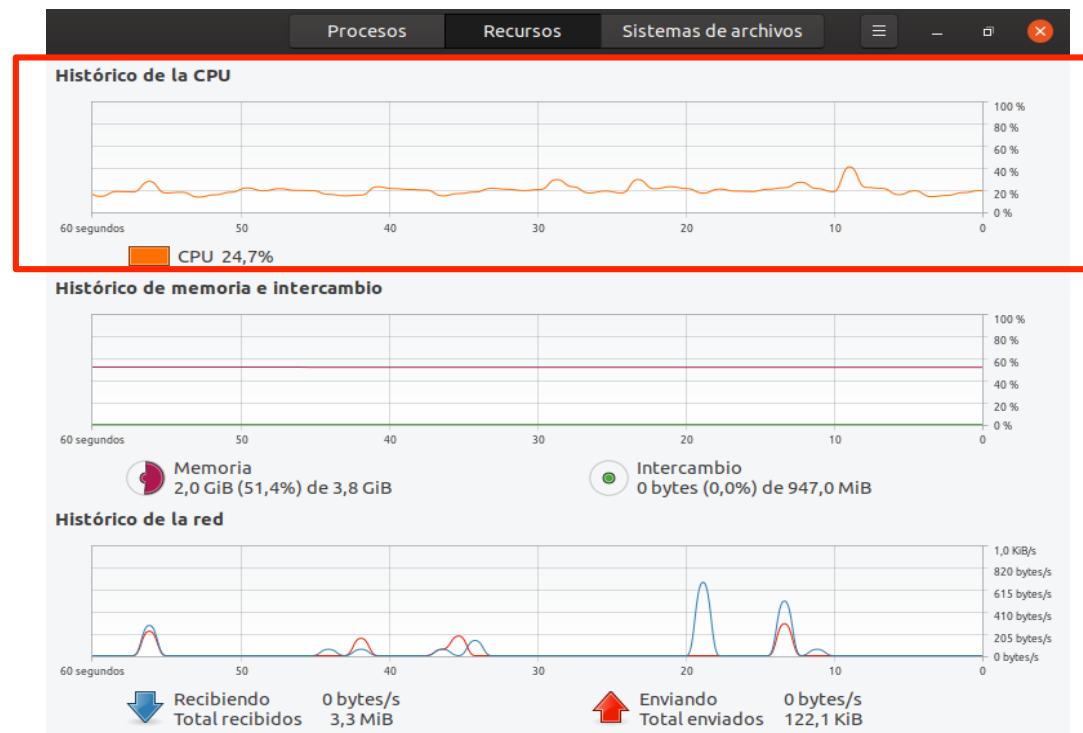
8. Monitorización de la CPU.

- Para monitorizar el uso de CPU por los procesos en Windows utilizamos el **Administrador de tareas** que vimos anteriormente. En este caso, es necesario pulsar sobre **Más detalles**
- Desde ahí debemos acceder a la pestaña **Rendimiento** y seleccionar en la izquierda el primer elemento **CPU** donde veremos datos estadísticos de uso



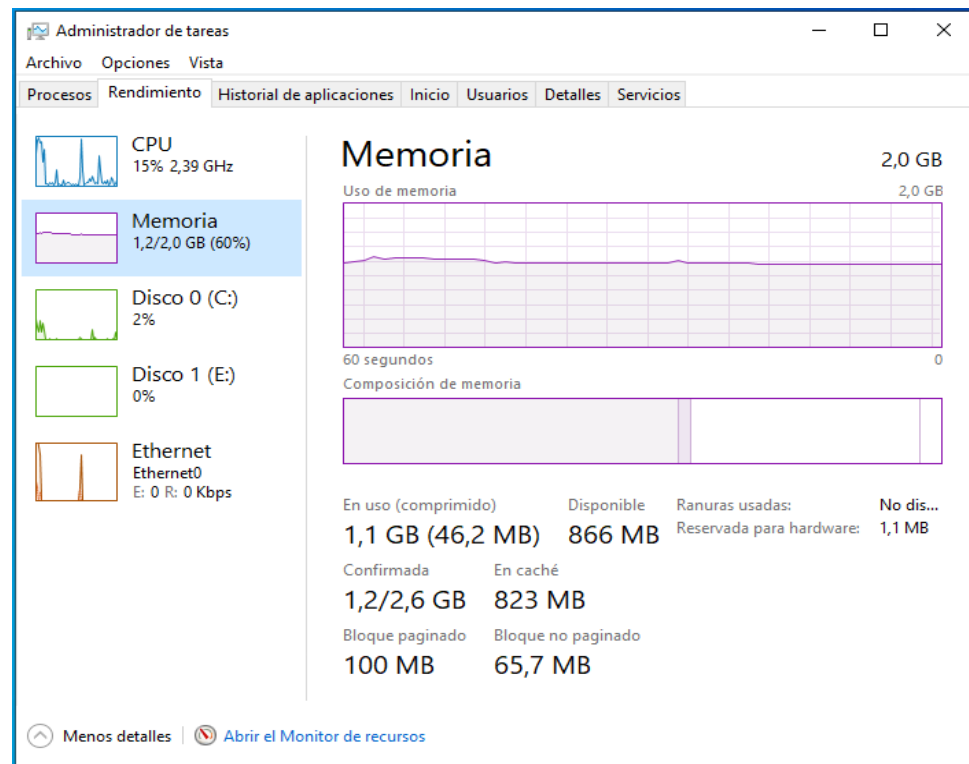
8. Monitorización de la CPU.

- Para monitorizar el uso de CPU por los procesos en Ubuntu utilizamos el **Monitor del sistema**. En este caso, es necesario pulsar sobre **Recursos**
- Desde ahí debemos acceder a la primera sección **Histórico de la CPU** donde se mostrará en tiempo real una estadística de uso de la CPU



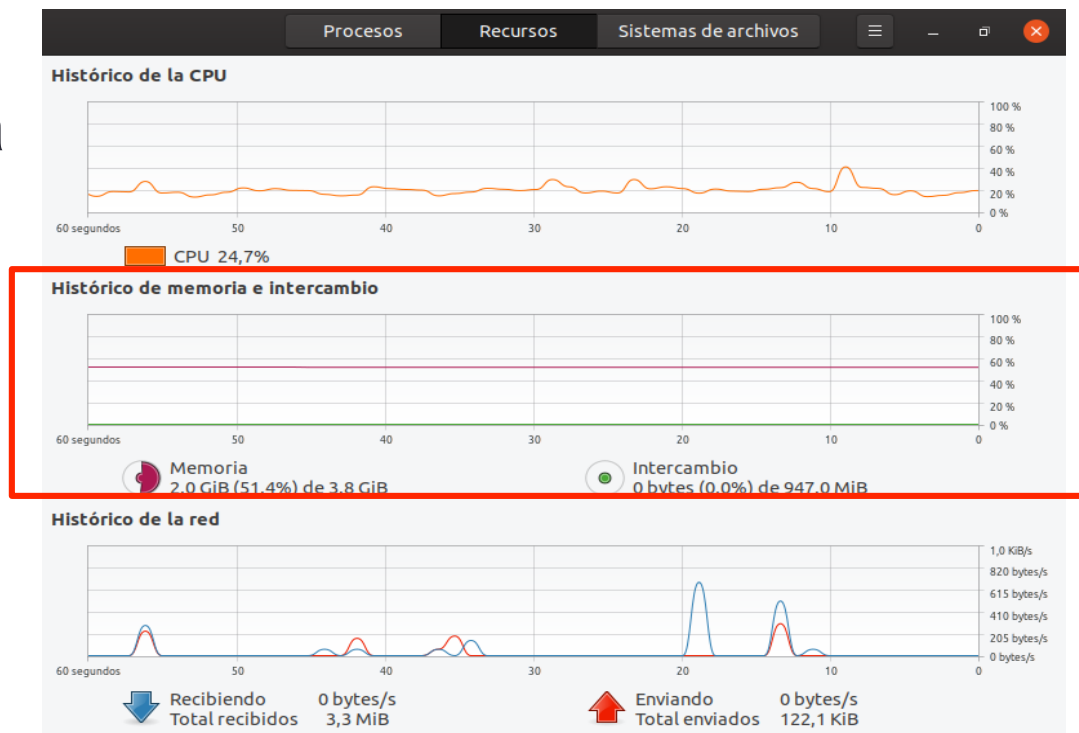
8. Monitorización de la RAM.

- Para monitorizar el uso de RAM por los procesos en Windows utilizamos el **Administrador de tareas** que vimos anteriormente. En este caso, es necesario pulsar sobre **Más detalles**
- Desde ahí debemos acceder a la pestaña **Rendimiento** y seleccionar en la izquierda el segundo elemento **Memoria** donde veremos datos estadísticos de uso



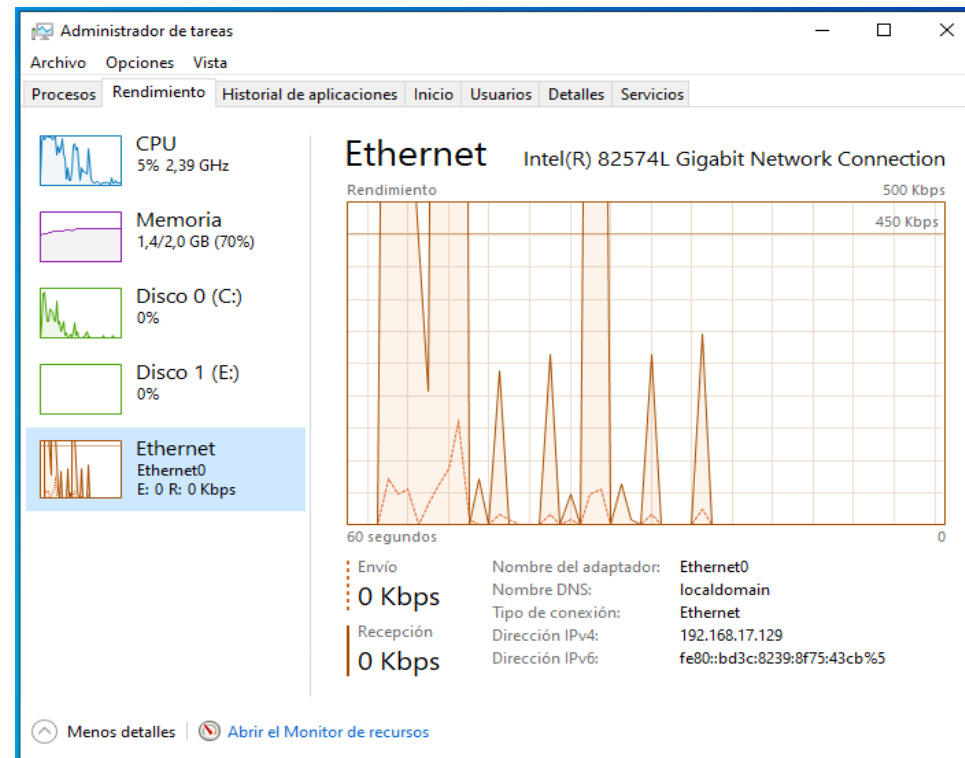
8. Monitorización de la RAM.

- Para monitorizar el uso de RAM por los procesos en Ubuntu utilizamos el **Monitor del sistema**. En este caso, es necesario pulsar sobre **Recursos**
- Desde ahí debemos acceder a la segunda sección **Histórico de la memoria e intercambio** donde se mostrará en tiempo real una estadística de uso de la memoria



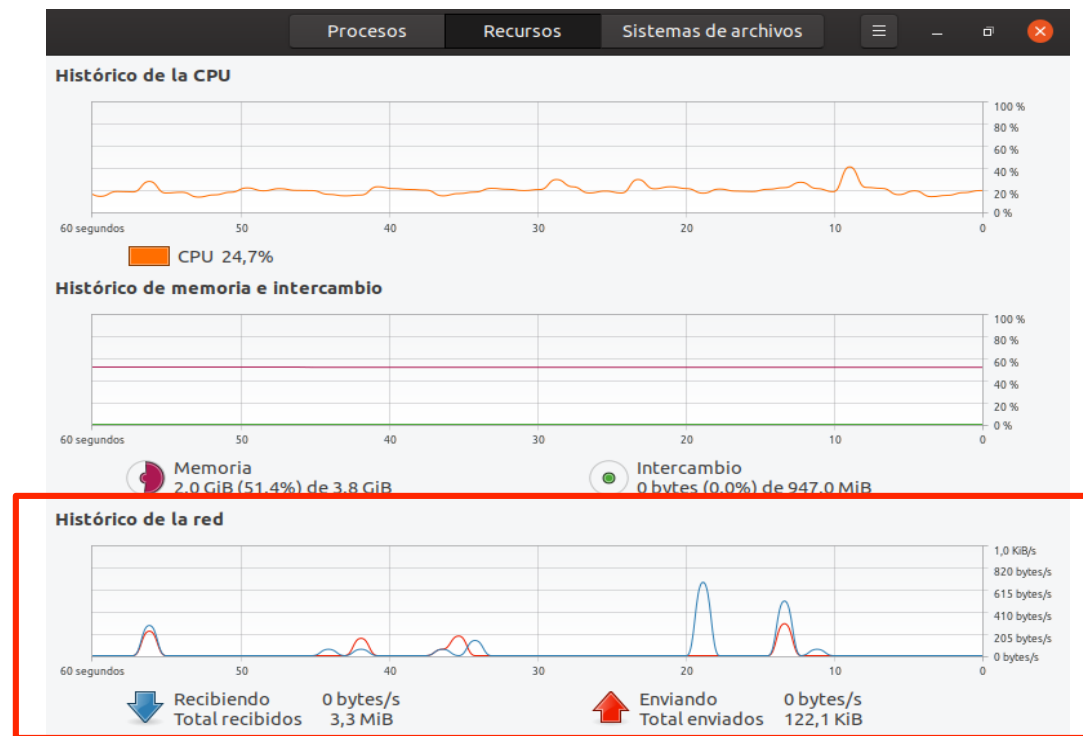
8. Monitorización de la red.

- Para monitorizar el uso de la red por los procesos en Windows utilizamos el **Administrador de tareas** que vimos anteriormente. En este caso, es necesario pulsar sobre **Más detalles**
- Desde ahí debemos acceder a la pestaña **Rendimiento** y seleccionar en la izquierda el tercer elemento **Ethernet** donde veremos datos estadísticos de uso



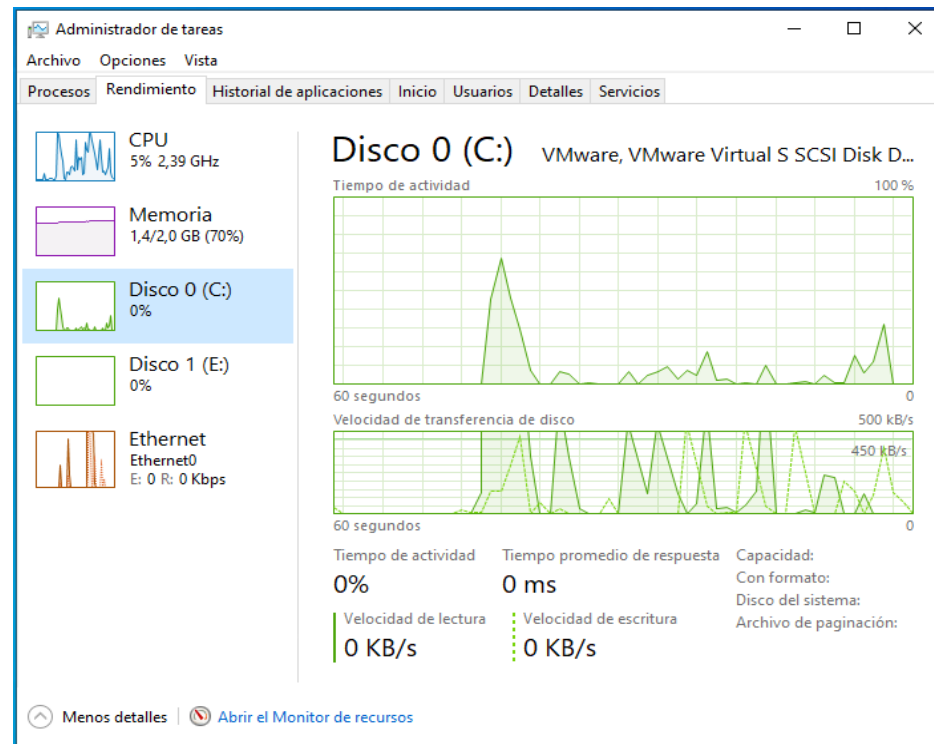
8. Monitorización de la red.

- Para monitorizar el uso de red por los procesos en Ubuntu utilizamos el **Monitor del sistema**. En este caso, es necesario pulsar sobre **Recursos**
- Desde ahí debemos acceder a la tercera sección **Histórico de la red** donde se mostrará en tiempo real una estadística de uso de la red



8. Monitorización del disco.

- Para monitorizar el uso de CPU por los procesos en Windows utilizamos el **Administrador de tareas** que vimos anteriormente. En este caso, es necesario pulsar sobre **Más detalles**
- Desde ahí debemos acceder a la pestaña **Rendimiento** y seleccionar en la izquierda el elemento del **disco** que queremos analizar donde veremos datos estadísticos de uso



8. Monitorización del disco.

- Para monitorizar el uso de disco por los archivos en Ubuntu utilizamos el **Monitor del sistema**. En este caso, es necesario pulsar sobre **Sistemas de archivos**
- Desde ahí debemos observar el disco o los discos que nos interese ver su uso.
- A la derecha de esta pestaña hay un botón con tres líneas que despliega un menú desde el que se puede acceder a **Buscar archivos abiertos** y desde el que podremos mostrar cada uno de los archivos que se está utilizando en el sistema en el momento de mirarlo.

8. Monitorización del sistema utilizando comandos

- Algunos comandos útiles, aparte de los ya vistos en apartados anteriores para monitorizar el sistema son:
- **vmstat** muestra los distintos consumos de memoria.
- **ss** (equivalente a netstat) muestra las conexiones de red
- **df** muestra el uso del disco duro
- **ls** lista de ficheros abiertos en el sistema
- **stat** muestra el estado de un fichero / sistema de archivos
- **who** Muestra los usuarios que están logueados. Su opción **-b** indica cuando se arrancó el sistema.
- **time** lanza un proceso y muestra su tiempo de ejecución
- **du** muestra el tamaño, en bytes, de un directorio