Objetivo

El objetivo de esta práctica es que el alumno se familiarice con los conceptos básicos del sistema operativo GNU/Linux, así como con su entorno y comandos principales.

1. Características de GNU/Linux :

S.O: El término correcto para el SO, no es "Linux", sino "GNU/Linux". *Intermediario entre el usuario y el hardware*, el que gestiona todo el sistema de cómputo y sus recursos.

Linux es en sí un núcleo (**kernel**), que junto al sistema GNU, dieron origen al SO "*GNU/Linux*".

(a) Mencione y explique las características más relevantes de GNU/Linux.

- Es un Sistema Operativo, **LIBRE** significa que los usuarios tienen la libertad de ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software. >>>> ¡***Nunca*** voy a poder restringir su uso!
- S.O. diseñado por programadores.
- S.O. **gratuito** y de **libre** distribución.
- Existen diversas distribuciones (customizaciones)
- **Es código abierto**, lo que nos permite estudiarlo, personalizarlo, auditar, aprovecharnos de la documentación, etc...
- ¡Podemos ver cómo está hecho!

(b) Mencione otros sistemas operativos y compárelos con GNU/Linux en cuanto a los puntos mencionados en el inciso a.

- ***Windows***: es un sistema operativo privativo, desarrollado por *microsoft*, cuenta con una sola línea orientada para usuarios, una sola orientada a servidores, etc.
- **Mac Os X**: otro sistema operativo privativo , desarrollado por *Apple* , la mayoría de su software es pago , está fuertemente ligado a un hardware muy específico.

- ***Solaris***: sistema operativo privativo ,desarrollado por Sun Microsystem, esta Está fuertemente orientado a Servidores y estaciones de trabajo.

(c) ¿Qué es GNU?

- GNU = **G**NU **N**o es **U**nix
- GNU es un sistema operativo de tipo Unix, lo cual significa que se trata de una colección de muchos programas: aplicaciones, bibliotecas, herramientas de desarrollo y hasta juegos. El desarrollo de GNU, iniciado en enero de 1984, se conoce como Proyecto GNU.

(d) Indique una breve historia sobre la evolución del proyecto GNU.

- GNU fue un proyecto iniciado en 1983 por ***Richard Stallman*** con el objetivo de crear un Unix que fuera libre.

- Para 1990 contaba con muchas aplicaciones, editor de texto, un compilador, y muchas bibliotecas, pero no tenía lo fundamental un núcleo.
  Había varios proyectos de núcleo pero ninguno prosperó, hasta que en 1992 decidieron fusionar sus dos proyectos y crearon GNU/LINUX.

- d) En 1985 se funda la Free Software Foundation con el propósito de dar soporte logístico , legal y financiero al proyecto GNU. En 1990 GNU ya tenía un compilador de C llamado GCC (GNU C Compiler , que luego se convirtió en un compilador multilenguaje). de 1985 a 1988 se seguía trabajando en distintos núcleos basados en otros. En 1992 se fusiona con Linux, y lo adopta como nuevo núcleo, naciendo Gnu/Linux.

(e) Explique qué es la multitarea, e indique si GNU/Linux hace uso de ella.

- **Multitarea**: Capacidad para ejecutar varios programas al mismo tiempo.

  LINUX utiliza la llamada *multitarea preventiva*, la cual asegura que todos los programas que se están utilizando en un momento dado serán ejecutados, siendo el sistema operativo el encargado de ceder tiempo de microprocesador a cada programa. Esto último es lo que lo hace preventivo y hace que Linux sea un sistema muy robusto, no se cuelga simplemente por que un programa usuario deja de andar.

(f) ¿Qué es POSIX?

- **POSIX** es el acrónimo de *Portable Operating System Interface*; la X viene de UNIX como señal de identidad de la API. **Es un estándar** de las *system calls* que debe proveer todo sistema operativo consiguiendo así que se puedan crear programas multiplataforma con mayor facilidad. Tiene como objetivo estandarizar las api de los SO.

- Es un estándar que define, de alguna manera, como deben estar diseñados los S.O. para ser compatibles entre sí.

2. Distribuciones de GNU/Linux :

(a) ¿Qué es una distribución de GNU/Linux ? Nombre al menos 4 distribuciones de GNU/Linux y cite diferencias básicas entre ellas.

- Una distribución de GNU/Linux es un conjunto de aplicaciones reunidas que permiten brindar mejoras para orientar el sistema operativo hacia ciertas funcionalidades en particular.

Las principales distribuciones de GNU/Linux son:

-Debian: Es una de las más grandes ramas de GNU/Linux desarrollada por la comunidad *Debian* , es muy estable y utiliza el formato de paquetes ".deb".

-Ubuntu: Es una distribución basada en Debian , orientada al usuario promedio , se caracteriza por su facilidad de uso , es desarrollada por *canonical* y se considera la distribución, junto con sus derivadas, más amigable con la que empezar a utilizar GNU/Linux

-Red Hat: Aunque ya fuera del mercado Red Hat Linux , creada por red hat , es una de las distribuciones de GNU/Linux más conocidas y de la cual nacieron muchas otras distribuciones. Fue la primera en usar el formato rpm en sus paquetes. A partir del 2003 Red Hat se unió al proyecto Fedora Core. Está principalmente orientado a Servidores.

-Slackware: Una de las primeras distribuciones de Linux y sigue en vigencia, es una distribución "profesional" , para usuarios avanzados.

-Gentoo: Es una distribución para usuarios con bastante experiencia en GNU/Linux.

Además del núcleo Linux, las distribuciones incluyen habitualmente las bibliotecas y herramientas del proyecto GNU y el sistema de ventanas X Window System. Dependiendo del tipo de usuarios a los que la distribución esté dirigida se incluye también otro tipo de software como procesadores de texto, hoja de cálculo, reproductores multimedia, herramientas administrativas, etcétera. En el caso de incluir herramientas del proyecto GNU, también se utiliza el término distribución GNU/Linux.

(b) ¿En qué se diferencia una distribución de otra?

- las distribuciones Linux pueden ser:
    - Comerciales o no comerciales.
    - Ser completamente libres o incluir software privativo.
    - Diseñadas para uso en el hogar o en las empresas.
    - Diseñadas para servidores, escritorios o dispositivos empotrados.
    - Orientadas a usuarios regulares o usuarios avanzados.
    - De uso general o para dispositivos altamente especializados, como un cortafuegos, un enrutador o un cluster computacional.
    - Diseñadas e incluso certificadas para un hardware o arquitectura específicos.
    - Orientadas hacia grupos en específico, por ejemplo a través de la internacionalización y localización del lenguaje, o por la inclusión de varios paquetes para la producción musical o para computación científica.
    - Configuradas especialmente para ser más seguras, completas, portables o fáciles de usar.
    - Soportadas bajo distintos tipos de hardware.
    - La diversidad de las distribuciones Linux es debido a cuestiones técnicas, de organización y de puntos de vista diferentes entre usuarios y proveedores. El modo de licenciamiento del software libre permite que cualquier usuario con los conocimientos e interés suficiente pueda adaptar o diseñar una distribución de acuerdo a sus necesidades.

(c) ¿Qué es Debian? Acceda al sitio e indique cuáles son los objetivos del proyecto y una breve cronología del mismo.

- Debian es un sistema operativo *estable y seguro* basado en Linux.
- Nació en el año 1993, de la mano del *proyecto Debian*, con la idea de crear un sistema GNU usando Linux como núcleo. El proyecto Debian es la organización responsable de su mantenimiento en la actualidad, y también desarrolla sistemas GNU basados en otros núcleos
- Uno de sus principales objetivos es separar en sus versiones el software libre del software no libre. El modelo de desarrollo es independiente a empresas, creado por los propios usuarios, sin depender de ninguna manera de necesidades comerciales. Debian no vende directamente su software, sino que lo pone a disposición de cualquiera en Internet, aunque sí permite a personas o empresas distribuir comercialmente este software mientras se respete su licencia.

- ***El proyecto Debian***
    - El proyecto Debian fue fundado en el año 1993 por Ian Murdock. Él escribió el manifiesto de Debian, que utilizó como base para la creación de la distribución Linux Debian. Dentro de este texto, los puntos destacables son: mantener la distribución de manera abierta, coherente al espíritu del núcleo Linux y de GNU.

    - El nombre de este se basa en la combinación del nombre de su entonces novia (posteriormente esposa) "**Deb**orah" con su propio nombre: "**Ian**", formando el acrónimo: *Debian*.

    - El proyecto creció lentamente al principio y lanzó sus primeras versiones 0.9x en 1994 y 1995. Las primeras portabilidades a otras arquitecturas fueron a comienzos de 1995, siendo la primera versión 1.x de Debian lanzada en 1996.

- En 1996, Bruce Perens sustituyó a Ian Murdock como el líder del proyecto. Por sugerencia del desarrollador Ean Schuessler, él dirigió el proceso de actualización del contrato social de Debian y de las pautas del software de Debian libremente, definiendo los puntos fundamentales para el desarrollo de la distribución. Él también inició la creación de la licencia de software legal de la organización.

- Bruce Perens se retiró en 1998, antes del lanzamiento de la primera versión Debian basada en glibc, bautizada como Debian 2.0. El proyecto procedió a elegir a nuevos líderes y a hacer dos revisiones de la versión 2.x, cada una incluyendo más portabilidades a otras arquitecturas y más paquetes. Convenientemente fue lanzado durante este periodo y la primera portabilidad a un núcleo no basado en el núcleo Linux, Debian GNU/Hurd. Las primeras distribuciones de Linux basadas en Debian, Corel Linux y la Stormix's Linux de Stormix, fueron comenzadas en 1999. Aunque no estuvieron desarrolladas por mucho tiempo, estas distribuciones eran las primeras de muchas distribuciones basadas en Debian.

- A finales de 2000, el proyecto realizó el mayor cambio a la estructura de los archivos y la organización de las versiones, reorganizando los procesos de liberación de paquetes del software con el nuevo "package pools" y creando una rama de prueba, relativamente estable para el lanzamiento siguiente. En 2001, los desarrolladores comenzaron a llevar a cabo una conferencia anual llamada: Debconf, con negociaciones y los talleres para los desarrolladores y los usuarios técnicos.

.

3. Estructura de GNU/Linux :

(a) Nombre cuales son los 3 componentes fundamentales de GNU/Linux.

- El ***kernel*** (también conocido como *núcleo*) es la parte **fundamental** de un sistema operativo. El kernel o núcleo de linux se podría definir como el corazón de este sistema operativo. Es el encargado de que el software y el hardware de una computadora puedan trabajar juntos.
- El ***Shell*** (intérprete de comandos) es el programa que recibe lo que se escribe en la terminal y lo convierte en instrucciones para el sistema operativo. Un intérprete de comandos es un programa que lee las entradas del usuario y las traduce a instrucciones que el sistema es capaz de entender y utilizar.
- El ***Filesystem*** se traduce como "sistema de archivos"; y es la forma en que dentro de un SO se organizan y se administran los archivos.

(b) Mencione y explique la estructura básica del Sistema Operativo GNU/Linux.

kernel (Núcleo):

- Ejecuta programas y gestiona dispositivos de hardware
- Es el encargado de que el software y el hardware puedan trabajar juntos
- Sus funciones más importantes son la **administración** de *memoria, CPU y la E/S*
- Es un núcleo monolítico híbrido:(*Un núcleo monolítico es un núcleo donde todos los servicios (sistema de archivos, VFS, controladores de dispositivos, etc.), así como la funcionalidad central (programación, asignación de memoria, etc.) son un grupo muy unido que comparte el mismo espacio. Esto se opone directamente a un microkernel).*

• El ***kernel*** de un sistema es el componente central que sirve para dar vida al hardware. Es la capa responsable de asegurar que todos los programas y procesos tengan acceso a los recursos que necesitan de la máquina (memoria RAM, acceso a disco y el control de la CPU, por ejemplo) al mismo tiempo, de modo que haya un recurso compartido de estos. En otras palabras, **es el cerebro del sistema operativo**; el responsable de coordinar el acceso al hardware y los datos entre los diferentes componentes del sistema. Constituye en torno al 10% del código de programación total del sistema.

- El ***kernel*** proporciona acceso al hardware de los distintos programas instalados. Además es el encargado de gestionar los recursos a través de servicios de "System Calls". Linux es el kernel de este sistema operativo libre GNU/Linux y está escrito prácticamente en su totalidad en código "C".

- Linux continúa liberando nuevas versiones del kernel del sistema, con modificaciones que implementan las compañías o los propios usuarios, ya que, debemos recordar que el kernel es accesible para todos los usuarios. Para distinguir cual es una distribución "estable" del kernel, debemos fijarnos en los números de versión que contienen. El segundo número que acompaña al normal del kernel nos dirá si es una versión estable, cuando tenemos números pares: 2.6.. Y será una versión inestable o en desarrollo cuando tenemos números impares.

- Para ver la versión de compilación del kernel de nuestro sistema operativo, tendremos que colocar el siguiente comando:
    - *uname -a*
        - Proporciona toda la información acerca del kernel
    - *uname -r*
        - Para solamente ver que versión es la que tenemos.

- Actualmente Ubuntu dispone de la versión 4.18.0 del kernel, por lo que sabemos que es una versión estable.

Shell

- También conocido como CLI (*Command Line Interface*)
- Modo de comunicaci´on entre el usuario y el SO
- Ejecuta programas a partir del ingreso de comandos
- Cada usuario puede tener una interfaz o shell
- Se pueden personalizar
- Son programables
- Bourne Shell (sh), Korn Shell (ksh), Bourne Again Shell (bash)(autocompletado, history, alias)

- En la siguiente capa en comunicación directa con el kernel, tenemos el caparazón del shell. **La shell actúa como un intérprete de comandos.** En un sistema operativo existen distintos shell para recoger los comandos y acciones que introduce un usuario. De esta forma el sistema operativo realizará una serie de acciones en respuesta por ejemplo a un comando que escribamos en el terminal, o un clic de ratón.

- Este programa siempre estará en funcionamiento en nuestro sistema registrando todas las órdenes para decodificarlas y pasarlas al núcleo para realizar las acciones solicitadas. Prácticamente todas las órdenes que enviamos a la shell son programas ejecutables que están dentro del sistema de ficheros del sistema operativo.

- *La shell principal de nuestro sistema, recibe el nombre de* ***Bash***, (Bourne-again shell) es un programa intérprete de comandos de Unix.

Filesystem

- Organiza la forma en que se almacenan los archivos en dispositivos de almacenamiento (fat, ntfsm ext2, ext3, reiser, etc.)
- El adoptado por GNU/Linux es el Extended (v2, v3, v4)
- Hace un tiempo se está debatiendo el reemplazo de ext por Btrfs (B-tree FS) de Oracle
    - Soporte de mayor tama˜no de archivos
    - Más tolerante a fallas y comprobación sin necesidad de desmontar el FS
    - Indexación
    - Snapshots
    - Compresi´on
    - Desfragmentación

- **Directorios más importantes según FHS (Filesystem Hierarchy Standard)**
    - "**/**" Tope de la estructura de directorios. Es como el C:\
    - "**/home**" Se almacenan archivos de usuarios (Mis documentos)
    - "**/var**" Información que varía de tamaño (logs, BD, spools)
    - "**/etc**" Archivos de configuraci´on
    - "**/bin**" Archivos binarios y ejecutables
    - "**/dev**" Enlace a dispositivos
    - "**/usr**" Aplicaciones de usuarios

- Ofrece recursos capaces de garantizar la interacción con el usuario y popularmente conocida como la capa de software. Esta capa permite que las aplicaciones de usuario se ejecuten.

- En otras palabras, al núcleo del sistema **no** se puede acceder directamente por el usuario o administrador del sistema; esto solo puede ser posible a través de programas de utilidad del sistema, así

como la terminal de línea de comandos (CLI), software para la recopilación, software de gestión de disco/memoria o el control de los procesos del sistema.

- componente **fundamental** de GNU/Linux (es parte de la estructura básica), que define la forma en que se organizan y administran los archivos en el sistema: cómo se pueden acceder, cómo se almacenan, cómo se referencian, cómo se comparten y protegen, cómo se administra el espacio de la memoria secundaria para mantener a los archivos como así también al espacio libre.

4. Kernel :

(a) ¿Qué es? Indique una breve reseña histórica acerca de la evolución del Kernel de GNU/Linux:

- El kernel (también conocido como núcleo) ***es la parte fundamental de un sistema operativo***. El kernel o núcleo de linux se podría definir como el corazón de este sistema operativo. Es el encargado de que el software y el hardware de una computadora puedan trabajar juntos.

- En 1991 Linus Torvalds empieza a crear un sistema operativo basado en Minx (clon de Unix).
- El 5 de octubre se anunció la primera versión de Linux.
- En 1992 se combina con Gnu formando GNU/LINUX.
- La versión 1.0 apareció el 14 de marzo de 1994.
- Desarrollo continuo por miles de programadores alrededor del mundo.
- En mayo de 1996 se decide adoptar a Tux como mascota oficial de Linux (pinguino)

(b) ¿Cuáles son sus funciones principales?

- Las funciones principales del kernel son:
    - -Administración de la memoria, para todos los programas en ejecución.
    - -Administración del tiempo de procesador, que estos programas en ejecución utilizan.
    - -Acceder a los periféricos/elementos y hardware de entrada y salida de una forma práctica y cómoda.

(c) ¿Cuál es la versión actual? ¿Cómo se definía el esquema de versionado del Kernel en versiones anteriores a la 2.4? ¿Qué cambió en el versionado se impuso a partir de la versión 2.6?

- Actualmente Ubuntu dispone de la versión 5.13.13 (26/08/2021) del kernel
- En 1996 se lanza la versión 2.0 y se define un sistema de *nomenclatura*.
- En 1999 se lanza la versión 2.2 que provee mejoras de portabilidad y se desarrolla hasta febrero de 2004, llegando a la versión 2.2.26.
- En 2001 se lanza la versión 2.4 y se deja de desarrollar en 2010 llegando a 2.4.37.11
    - La versión 2.4 fue la que catapultó a Linux como un sistema operativo estable y robusto.Durante ese periodo es que se empieza a usar Linux más asiduamente.

- En 2003 se lanza la versión 2.6, que incluía soporte de hilos, soporte para nuevo hardware, y mejoras en la planificación.
- En Agosto de 2011 se lanzó la versión 2.6.3.9.4
- En Julio de 2011 se lanzó la versión 3.0
- En Abril de 2015 se lanzó la versión 4.0
- En marzo de 2019 se lanzó la versión 5.0

- La manera de versionar el sistema operativo consta de cuatro dígitos “**A.B.C.D**”, donde “**A**” denota versión, el “**B**” denota mayor revisión.
- el esquema de versionado del Kernel en versiones anteriores a la 2.4, los números impares indican desarrollo y los pares producción.

- El número **C** indica una *revisión mayor en el núcleo*. En la forma anterior de versiones con tres números, esto fue cambiado cuando se implementaron en el núcleo los parches de seguridad, bug fixes, nuevas características o drivers. Con la nueva política, solo es cambiado cuando se introducen nuevos controladores o características; cambios menores se reflejan en el número **D**.
- El número **D** se produjo cuando un grave error, que requiere de un arreglo inmediato, se encontró en el código NFS de la versión 2.6.8. Sin embargo, no había otros cambios como para lanzar una nueva revisión (la cual hubiera sido 2.6.9). Entonces se lanzó la versión 2.6.8.1, con el error arreglado como único cambio. Con 2.6.11, esto fue adoptado como la nueva política de versiones. Bug-fixes y parches de seguridad son actualmente manejados por el cuarto número dejando los cambios mayores para el número **C**.

(d) ¿Es posible tener más de un Kernel de GNU/Linux instalado en la misma máquina?

- **Si**, es posible tener *más de un kernel* de GNU/Linux y tener la posibilidad de elegir con cual bootear. Hay que mencionar además que un usuario puede bajar el código fuente del kernel y compilarlo por sus propios medios.

(e) ¿Dónde se encuentra ubicado dentro del File System?

- Se encuentra en: "/boot".

(f) ¿El Kernel de GNU/Linux es monolítico? Justifique.

- Linux es un núcleo monolítico híbrido, es decir un núcleo grande y complejo, que engloba todos los servicios del sistema. Sin embargo, cualquier cambio a realizar en cualquier servicio requiere la recopilación del núcleo y el reinicio del sistema para aplicar los nuevos cambios. Linux tiene la capacidad de cargar y descargar los controladores de dispositivos y las extensiones del S.O. Como módulos es por eso que se considera híbrido al kernel.

5. Intérprete de comandos (Shell)):

(a) ¿Qué es?

- El ***Shell*** (intérprete de comandos) es el modo de comunicación entre el usuario y el SO.
-

(b) ¿Cuáles son sus funciones?

- Ejecuta programas a partir del ingreso de comandos.
- Traduce las entradas del usuario a señales que el sistema puede entender y manejar.

(c) Mencione al menos 3 intérpretes de comandos que posee GNU/Linux y compárelos entre ellos.

- Existen tres tipos básicos de intérpretes de comandos:
  - Korn-Shell(ksh),
  - Bourne-Shell(sh) ,
  - C-Shell(csh)
  - y el más usado bourne-again shell (**bash**).

- -Korn Shell : Es compatible con el Bourne Shell y contiene muchos elementos del interprete de comandos C.La version ksh93 soporta arreglos asociativos y aritmética de punto flotante.

- -C-Shell: Tiene una sintaxis muy parecida al lenguaje C , edición de comandos y asignación de alias. El indicador es el signo de porcentaje.
- -Bourne Shell(sh): Muy usada en Unix y una de las más extendidas. Los Scripts pueden ser invocados como comandos usando el nombre del archivo. Permite ejecución sincrónica y asincrónica de los comandos.Soporta redireccionamientos de Entrada/Salida y pipelines, variables sin tipo.
- -Bourne-again shell (**bash**): Es el interprete de ordenes usado por defecto en la mayorías de las distribuciones de Linux. A diferencia de la shell de Bourne, en el bash los parámetros se pasan como **$1** y se puede saber cuántos se han recibido con **$#**. Además se dispone del array **$@** que contiene todos los parámetros. También difiere en la forma en la que se realizan los cálculos con enteros y el redireccionamiento de E/S.

(d) ¿Dónde se ubican (path) los comandos propios y externos al Shell?

- */usr/bin/ls*

(e) ¿Por qué considera que el Shell no es parte del Kernel de GNU/Linux ?

- **El Shell** no es parte del kernel ya que no provee ninguna de las funcionalidades que un kernel debe proveer, **simplemente es un medio de comunicación** entre el SO y el usuario.

(f) ¿Es posible definir un intérprete de comandos distinto para cada usuario? ¿Desde dónde se define? ¿Cualquier usuario puede realizar dicha tarea?

- **?**

6. Sistema de Archivos (File System):

(a) ¿Qué es?

- Es el encargado de organizar la forma en que se almacenan los archivos en los dispositivos de almacenamiento.
- Es el componente ***fundamental*** de GNU/Linux (es parte de la estructura básica), que define la forma en que se organizan y administran los archivos en el sistema: cómo se pueden acceder, cómo se almacenan, cómo se referencian, cómo se comparten y protegen, cómo se administra el espacio de la memoria secundaria para mantener a los archivos como así también al espacio libre.

(b) Mencione sistemas de archivos soportados por GNU/Linux.

- Sistemas soportados en GNU/Linux y actualmente más usados:
  - -Ext2.
  - -Ext3.
  - -Ext4.
  - -ReiserFS.
  - -XFS.

(c) ¿Es posible visualizar particiones del tipo FAT y NTFS en GNU/Linux ?

- Si, es posible.

(d) ¿Cuál es la estructura básica de los File System en GNU/Linux ? Mencione los directorios más importantes e indique qué tipo de información se encuentra en ellos. ¿A qué hace referencia la sigla FHS?

- **FHS** = *file hierarchy system.*
  - / — Directorio raíz.
  - /bin— Binarios y comandos esenciales de todo el sistema Linux.
  - /boot— Archivos fundamentales para el arranque.
  - /dev— Archivos de dispositivos.
  - /etc— Se almacenan los archivos de configuración del sistema.
  - /home— Directorio para las cuentas de usuario.
  - /lib— Directorio de las librerías compartidas.
  - /mnt— Directorio de montaje de dispositivos temporales.
  - /root— Directorio de home del usuario root.

- /sbin— Binarios fundamentales del sistema.
- /tmp— Ficheros temporales.
- /usr— Aplicaciones de usuario.
- /var— Directorio de información variable, log etc.

7. Particiones:

(a) Definición. Tipos de particiones. Ventajas y Desventajas.

- Existen 3 tipos diferentes de particiones:

  • ***Partición primaria*:** Son las divisiones crudas o primarias del disco, **solo pueden haber 4 de estas**. Depende de una tabla de particiones. Un disco físico completamente formateado, consiste en realidad de una partición primaria que ocupa todo el espacio del disco, y posee un sistema de archivos. A este tipo de particiones, prácticamente cualquier sistema operativo puede detectarlas y asignarles una unidad, siempre y cuando el sistema operativo reconozca su formato (sistema de archivos).

  • ***Partición extendida*:** Es otro tipo de partición que actúa como una partición primaria; sirve para contener infinidad de unidades lógicas en su interior. Fue ideada para romper la limitación de 4 particiones primarias en un solo disco físico. Solo puede existir una partición de este tipo por disco, y solo sirve para contener particiones lógicas. Por lo tanto, es el único tipo de partición que no soporta un sistema de archivos directamente.

  • ***Unidad lógica*:** Ocupa un trozo de partición extendida o la totalidad de la misma, la cual se ha formateado con un tipo específico de sistema de archivos (FAT32, NTFS, ext2,...) y se le a asignado una unidad, si el sistema operativo reconoce las particiones lógicas o su sistema de archivos.

Ventajas:

- Se puede guardar una copia de seguridad de los datos del usuario en otra partición del mismo disco, para evitar la pérdida de información importante.
- Si tienes un error/problema en una de ellas, las demás no se verán afectadas.
- Puedes tener diferentes sistemas operativos en tu PC, totalmente independientes unos de otros.
- Puedes tener tus archivos de datos en particiones totalmente independientes.
- Puedes borrar/cambiar el contenido de una partición, sin que esto afecte a las demás.

(b) ¿Cómo se identifican las particiones en GNU/Linux ? (Considere discos IDE, (SATA = SCSI) y SCSI).

- Disqueteras
  - Primera disquetera: /dev/fd0 (en Windows sería la disquetera A:)
  - Segunda disquetera: /dev/fd1

- /dev/hda, /dev/hdb, /dev/hdc, etc... Sería para los dispositivos IDE (antiguos)
- /dev/sda, /dev/sdb, /dev/sdc, etc... Sería para los dispositivos SCSI (actuales)

- Discos duros (en general: /dev/hdx#, donde x es el disco y # es la partición)
  - Primer disco duro: (todo el disco) /dev/hd**a**
    - Particiones primarias
      - Primera partición primaria: /dev/hda1
      - Segunda partición primaria: /dev/hda2
      - Tercera partición primaria: /dev/hda3
      - Cuarta partición primaria: /dev/hda4
    - Particiones lógicas
      - Primera partición lógica: /dev/hda5
      - Sucesivamente: /dev/hda#
  - Segundo disco duro: (todo el disco) /dev/hd**b**
    - Particiones primarias
      - Primera partición primaria: /dev/hdb1
      - Segunda partición primaria: /dev/hdb2
      - Tercera partición primaria: /dev/hdb3

- Cuarta partición primaria: /dev/hdb4
- Particiones lógicas
  - Primera partición lógica: /dev/hdb5
  - Sucesivamente: /dev/hdb#

- Discos SCSI
  - Primer disco SCSI: /dev/sda
  - Segundo disco SCSI: /dev/sdb
    - Sucesivamente ...

- Primer CD-ROM SCSI: /dev/scd0,
- también conocido como /dev/sr0

(c) ¿Cuántas particiones son necesarias como mínimo para instalar GNU/Linux ? Nómbrelas indicando tipo de partición, identificación, tipo de File System y punto de montaje.

- Como mínimo 2, la raíz "/" y la swap o de intercambio.

(d) Ejemplifique diversos casos de particionamiento dependiendo del tipo de tarea que se deba realizar en su sistema operativo.

- **?**

(e) ¿Qué tipo de software para particionar existe? Menciónelos y compare.

- ***partman*:**

  Herramienta original de Linux para particionar discos. Esta «navaja suiza» también puede ajustar el tamaño de las particiones, crear sistemas de ficheros (como se llama en Windows a "formatear") y asignarlos a sus respectivos puntos de montaje.

- ***fdisk*:**

  Es la herramienta original de Linux para particionar discos, buena para expertos. Sea cuidadoso si tiene una partición de FreeBSD en su máquina. Los núcleos instalados traen soporte para

éste tipo de partición, pero la manera en que fdisk la representa, puede (o no) ser un poco diferente. Para más Información, revise el CÓMO de Linux+FreeBSD.

- ***cfdisk*:**

  Una herramienta para particionar a pantalla completa, muy fácil de usar. Recomendada para la mayoría de los usuarios. *cfdisk* no reconoce las particiones de FreeBSD, y nuevamente, los dispositivos mostrados en pantalla pueden ser un tanto diferentes a los que realmente tiene.

8. Arranque (bootstrap) de un Sistema Operativo:

(a) ¿Qué es el BIOS? ¿Qué tarea realiza?

- El **BIOS** (*Basic Input Output System*) es un software de bajo nivel que se halla en el motherboard.
- Cuando se arranca la computadora el BIOS se ejecuta, realizando el POST (Power-on self-test), que incluye rutinas que, entre otras actividades, fijan valores de las señales internas, y ejecutan test internos (RAM, el teclado, y otros dispositivos a través de los buses ISA y PCI).

(b) ¿Qué es UEFI? ¿Cuál es su función?

- UEFI (Unified Extensible Firmware Interface)
- La característica principal es que el firmware BIOS y el software utilizado estará firmado digitalmente.

(c) ¿Qué es el MBR? ¿Qué es el MBC?

- **MBR** (*master boot record*): El primer sector del disco de inicio (seleccionado de entre un conjunto de posibles dispositivos de arranque)
- El **MBR** puede contener un código de arranque denominado **MBC** (*master boot code*) y una marca de 2 bytes que indica su presencia o puede solamente contener la tabla de particiones. En el último caso el BIOS ignora este MBR
- (*el MBR*) A *veces*, se emplea para el arranque del sistema operativo con bootstrap, otras veces es usado para almacenar una tabla de particiones y, en ocasiones, se usa sólo para identificar un

dispositivo de disco individual, aunque en algunas máquinas esto último no se usa y es ignorado.
Si el disco es "bootable", los primeros 446 bytes del MBR (sector de arranque), están ocupados por un pequeño trozo de código denominado código maestro de carga MBC ("Master Boot Code") o cargador inicial (bootstrap loader), que es cargado por la BIOS para comenzar el proceso de carga. El bootstrap loader repasa la tabla maestra de particiones buscando una partición activa. En caso de encontrarla, busca su sector inicial (8.1.2c2), carga su código en memoria, y le transfiere el control. Dicho código es ya capaz de cargar y ejecutar cualquier otro programa situado en cualquier partición del disco. Que a su vez inicializará directamente el SO, o tal vez una utilidad conocida como gestor de arranque, que permite elegir entre distintas alternativas.

(d) ¿A qué hacen referencia las siglas GPT? ¿Qué sustituye? Indique cuál es su formato.

- La ***tabla de particiones* GUID** (GPT) es un estándar para la colocación de la tabla de particiones en un disco duro físico. Es parte del estándar "*Extensible Firmware Interface*" (EFI) propuesto por Intel para reemplazar el viejo BIOS del PC, heredado del IBM PC original.
- La **GPT** sustituye al **MBR** (Master Boot Record) usado con el BIOS.
- Especifica la ubicación y formato de la tabla de particiones en un disco duro.

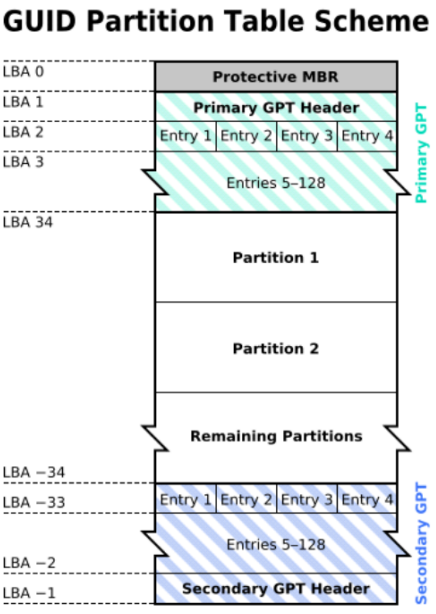

Figura 2: Esquema de particionado GPT.

Formato: (*Figura 2*)

- La GPT en sí, empieza en la cabecera de la tabla de particiones.
- GPT usa modo de direccionamiento lógico (**LBA**, logical block addressing) en vez del modo cilindro-cabeza-sector usado con el MBR. El MBR "heredado" se almacena en el LBA 0.

- En el LBA 1 está la cabecera GPT. La tabla de particiones en sí está en los bloques sucesivos. GPT proporciona asimismo redundancia. La cabecera GPT y la tabla de particiones están escritas tanto al principio como al final del disco lo cual provee redundancia, una importante característica del GPT.

(e) ¿Cuál es la funcionalidad de un "Gestor de Arranque"? ¿Qué tipos existen? ¿Dónde se instalan? Cite gestores de arranque conocidos.

- Un gestor de arranque es un pequeño programa para arrancar un sistema operativo. Habitualmente se instala en el MBR y asume el rol de MBC.
- ***Gestor de arranque*** (boot loader en inglés):

  Es un programa sencillo (que no tiene la totalidad de las funcionalidades de un sistema operativo) diseñado exclusivamente para preparar todo lo que necesita el sistema operativo para funcionar. Normalmente se utilizan los cargadores de arranque multietapas, en los que varios programas pequeños se suman los unos a los otros, hasta que el último de ellos carga el sistema operativo. En los ordenadores modernos, el proceso de arranque comienza con la CPU ejecutando los programas contenidos en la memoria ROM en una dirección predefinida.

- La diferencia entre ***GRUB y LILO*** es que GRUB es compatible con **varios** sistemas operativos, mientras que LILO se limita solo a dispositivos basados en Linux. GRUB se puede utilizar en Windows, macOS, Linux, Unix, mientras que el uso de LILO es uno. **GRUB** es un nuevo cargador de arranque predeterminado, mientras que **LILO** es un antiguo cargador de arranque predeterminado.

- En computación, el Grand Unified Bootloader (**GRUB**) es un gestor de arranque múltiple que se usa comúnmente para iniciar dos o más sistemas operativos instalados en un mismo ordenador.

- 1. El BIOS busca un dispositivo de inicio (como el disco duro) y pasa el control AL registro maestro de inicio (Master Boot Record, **MBR**, los primeros 512 bytes del disco duro).

- 2. El MBR contiene la fase 1 de **GRUB**. Como el MBR es pequeño (512 bytes), la fase 1 sólo carga la siguiente fase del GRUB (ubicado físicamente en cualquier parte del disco duro). La fase 1 puede cargar ya sea la fase 1.5 o directamente la 2
- 3. GRUB fase 1.5 está ubicada en los siguientes 30 kilobytes del disco duro. La fase 1.5 carga la fase 2.
- 4. GRUB fase 2 (cargada por las fases 1 o 1.5) recibe el control, y presenta al usuario el menú de inicio de GRUB.
- 5. GRUB carga el kernel seleccionado por el usuario en la memoria y le pasa el control.
- ***LILO*** ("Linux Loader") es un gestor de arranque de Linux que permite iniciar este sistema operativo junto con otras plataformas en el mismo ordenador. Fue desarrollado inicialmente por Werner Almesberger, actualmente está a cargo de John Coffman.
- LILO funciona en una variedad de sistemas de archivos y puede arrancar un sistema operativo desde el disco duro o desde un disco flexible externo.
- LILO permite seleccionar entre 16 imágenes en el arranque.
- LILO puede instalarse también en el master boot record (MBR).

(f) ¿Cuáles son los pasos que se suceden desde que se prende una computadora hasta que el Sistema Operativo es cargado (proceso de bootstrap)?

- A continuación se describe el *proceso de arranque* normal de UEFI:
    - Se busca una partición identificada por el código GUID “*C12A7328-F81F- 11D2-BA4B-00A0C93EC93B*” en los dispositivos de almacenamiento particionados con GPT, o bien una partición con tipo 0xEF marcada como partición de arranque en dispositivos particionados con el esquema MBR. Esta es la “EFI System Partition” que debe tener formato UEFI filesystem.
    - Si se encuentra esa partición, el firmware accede a su filesystem y carga el bootloader por defecto desde \EFI\BOOT\boot{arch}.efi o bien el bootloader o shell que tenga configurado el firmware.

- El bootloader es ejecutado en un entorno en “modo protegido” o en “long mode” dependiendo de la arquitectura del equipo.
- Utilizando los servicios provistos por el firmware UEFI, el bootloader puede cargar el sistema deseado, en caso que el kernel a cargar se encuentre en una partición con un UEFI filesystem. No es necesario que el bootloader tenga soporte para ningún filesystem ya que puede acceder al mismo usando los servicios de acceso a archivos provistos por el firmware.

(g) Analice el proceso de arranque en GNU/Linux y describa sus principales pasos.

Al **encender** el ordenador, en lo que se denomina un arranque en frío, éste se comprueba a sí mismo usando un código que se denomina **POST** (Power On Self Test).
El objetivo de dicha prueba es verificar que los elementos imprescindibles para su funcionamiento están presentes y disponibles (la memoria RAM, por ejemplo, entre otros).
Dados los escasos recursos con los que cuenta un ordenador al que pudieran fallarle partes muy básicas, los fallos suelen ser notificados al usuario por medio de indicaciones acústicas a través del zumbador o altavoz que suele llevar toda placa base, usando secuencias de pitidos predeterminadas para cada tipo de fallo.

Si la prueba se **pasa con éxito**, se ejecuta a continuación un programa denominado “bootstrap loader” **(cargador de inicio**) que reside en la memoria de la BIOS. El objetivo del programa cargador es encontrar un sector de arranque en alguno de los dispositivos de almacenamiento disponibles (antes tan sólo discos duros o flexibles, pero hoy en día es posible arrancar de prácticamente cualquier dispositivo de almacenamiento - llaves USB, tarjetas CF (Compact Flash) - que pueda montar como disco, o incluso desde una red local).
Un **sector de arranque** es el primer sector de un disco y contiene un pequeño programa que es capaz de cargar un Sistema Operativo.
Los sectores de arranque están marcados con la firma 0xAA55 en los bytes 0x1FE y siguiente. Esto es lo que comprueba el programa cargador. El cargador de inicio tiene una lista de lugares en que buscar un sector de arranque. Esto es lo que se programa cuando en la BIOS seleccionamos los

dispositivos de los que arrancar y su orden. El pequeño programa que se ejecuta desde el sector de arranque puede ser de varios tipos. En el caso de Linux, el más frecuentemente usado era **LILO**, pero ya hace tiempo que se usa en bastantes distribuciones un cargador alternativo, llamado **GRUB**. Otros Sistemas Operativos tienen su propio programa cargador.
Usaremos **LILO** en la descripción, pues es más ilustrativo.

En el caso concreto de **LILO**, lo que se carga en el sector de arranque es una parte de éste, denominada "first stage boot loader" (**primer paso del cargador de inicio**). Su misión es cargar y ejecutar el segundo paso del cargador de inicio.

Esta segunda parte suele mostrar una selección de Sistemas Operativos a cargar, procediendo a cargar a continuación el sistema escogido por el usuario (o bien el que se haya predeterminado como sistema por defecto, tras un tiempo de espera, si no escogemos nada). Esta información está incluida dentro del cargador de inicio y, para introducirla, se usa la orden 'lilo' que a su vez usa el contenido de '/etc/lilo.conf'. Todo ello sucede, por supuesto, con el ordenador ya en marcha. Una vez LILO ha cargado el "**kernel**" (núcleo) de Linux, le pasa el control a éste.
Al cargarlo, le ha pasado algunos parámetros. De éstos, el más importante es el que le dice al núcleo qué dispositivo usar como sistema de ficheros raíz, es decir, lo que en UNIX se denomina **'/'**. En un ordenador de sobremesa, la raíz sería típicamente una partición de un disco duro, pero en sistemas incrustados es frecuente usar como raíz una partición virtual basada en memoria (Flash, RAM,...).

Si el núcleo ha conseguido montar el sistema de ficheros raíz, **lo siguiente a ejecutar es el programa 'init'.** Sólo si dicho programa es estático (es decir, no usa librerías de funciones externas), no será necesario tener acceso a dichas librerías en la raíz. La librería básica en todo sistema GNU/Linux es la librería estándar C, "***glibc***". En un sistema mínimo, es decir, con una funcionalidad muy concreta, inmutable y sencilla, con tener solamente el programa 'init' enlazado estáticamente sería suficiente (y el núcleo, claro). En ese caso, init sería en realidad nuestro Programa de aplicación al completo.

En general, **'init'** es sólo el programa que se encarga de arrancar el resto de procesos que la máquina debe ejecutar. Entre sus tareas está el comprobar y montar sistemas de archivos, así como iniciar programas

servidores (daemons) para cada función necesaria. Otra tarea importante es la de arrancar procesos 'getty' cuya misión es proporcionar consolas donde poder registrarse y entrar en el sistema.

Las órdenes a seguir por '**init**' están en el fichero '**/etc/inittab**'. A partir de ese punto, y en función del sistema de inicialización utilizado (el más frecuente es el denominado "***System* V**") el proceso seguido por 'init' es distinto, pero en el fondo obedece más a un factor de forma, es decir, a una estrategia de ordenamiento de los "scripts" de inicialización de los distintos procesos que a un factor de fondo.

Una vez iniciados todos los servidores y procesos de entrada de usuario, o bien estamos delante de una consola de texto en la que el ordenador nos pide que nos identifiquemos, o bien estamos ante una consola gráfica que nos pide lo mismo, o bien estamos ante una pantalla llena de opciones sobre qué ejecutar (escuchar música, ver películas, por ejemplo) si el sistema arranca bajo un usuario predeterminado y no nos pide registrarnos. Esto es, si es que hablamos de un ordenador de sobremesa que, típicamente, nos ofrece una interfaz basada en dispositivos de entrada (teclado, ratón, mando a distancia) y de salida (monitor,TV, audio) para interactuar con él. Pero si el ordenador que se ha iniciado es un dispositivo con una funcionalidad concreta y su misión es controlar una serie de procesos y accedemos a él a través de medios indirectos (como pueda ser un navegador Web), el ordenador se inicia cuando está en disposición de prestar sus servicios, aun cuando no haya una indicación visual de dicho estado.

(h) ¿Cuáles son los pasos que se suceden en el proceso de parada (shutdown) de GNU/Linux ?

Cuando se lanza ***shutdown***, se notifica a los usuarios de este hecho y, además, se bloquea el sistema para que nadie más pueda acceder —creando el archivo **/etc/nologin**—, exceptuando el root. Acto seguido, se envía la señal SIGTERM a todos los procesos no definidos en *inittab* para el siguiente run level, provocando que terminen su ejecución de modo ordenado. Poco después, se envía una señal SIGKILL para que los procesos que no hayan atendido a SIGTERM concluyan también su ejecución —pero en este caso no de una manera "limpia"—.
shutdown lleva a cabo su cometido enviando una señal a ***init*** para que cambie a uno de estos niveles de ejecución, en función del efecto que se desee conseguir:

1. **Apagar o detener el sistema (nivel 0, opción "-r").** ´
2. **Entrar en modo monousuario (nivel 1, opción por defecto).** ´
3. **Reiniciar el sistema (nivel 6, "-h").**

Una vez cambiado el nivel de ejecución, si procede, se invoca a halt, reboot o poweroff según sea necesario.

(i) ¿Es posible tener en una PC GNU/Linux y otro Sistema Operativo instalado? Justifique.

- Si, ya que un disco rígido puede particionarse y en cada partición podemos tener un sistema de archivos distinto, sería como tener varios discos distintos, uno con cada SO, por lo tanto se necesitará un gestor de arranque como los descritos anteriormente.

9. Archivos:

(a) ¿Cómo se identifican los archivos en GNU/Linux ?

- La base del sistema de archivos de Linux, es obviamente *el archivo*, que no es otra cosa que la estructura empleada por el sistema operativo para almacenar información en un dispositivo físico como un disco duro, un disquete, un CD-ROM o un DVD. Como es natural un archivo puede contener cualquier tipo de información, desde una imagen en formato PNG o JPEG a un texto o una página WEB en formato HTML,

... ***El sistema de archivos es la estructura que permite que Linux maneje los archivos que contiene***...

Todos los archivos de Linux tienen un nombre, el cual debe cumplir unas ciertas reglas:

• Un nombre de archivo puede tener entre 1 y 255 caracteres.

• Se puede utilizar cualquier carácter excepto la barra inclinada / y no es recomendable emplear los caracteres con significado especial en Linux, que son los siguientes: = ^ ~ ' " ` * ; - ? [ ] ( ) ! & ~ < >.

Para emplear ficheros con estos caracteres o espacios hay que introducir el nombre del fichero entre comillas.

• Se pueden utilizar números exclusivamente si así se desea. Las letras mayúsculas y minúsculas se consideran diferentes, y por lo tanto no es lo mismo carta.txt que Carta.txt ó carta.Txt .

Como en Windows, se puede emplear un cierto criterio de "tipo" para marcar las distintas clases de ficheros empleando una serie de caracteres al final del nombre que indique el tipo de fichero del que se trata. Así, los ficheros de texto, HTML, las imágenes PNG o JPEG tienen extensiones .txt, .htm (o .html), .png y .jpg (o .jpeg) respectivamente.

Pese a esto Linux sólo distingue tres tipos de archivos:

• **Archivos o ficheros ordinarios**, son los mencionados anteriormente.

• **Directorios** (o carpetas), es un archivo especial que agrupa otros ficheros de una forma estructurada.

• **Archivos especiales**, son la base sobre la que se asienta Linux, puesto que representan los dispositivos conectados a un ordenador, como puede ser una impresora. De esta forma introducir información en ese archivo equivale a enviar información a la impresora. Para el usuario estos dispositivos tienen el mismo aspecto y uso que los archivos ordinarios.

(b) Investigue el funcionamiento de los editores vi y mcedit, y los comandos cat y more.

"Vi" es un editor con dos modos: edición y comandos. En el modo de edición el texto que ingrese será agregado al texto, en modo de comandos las teclas que oprima pueden representar algún comando de **vi**.

Cuando comience a editar un texto estará en modo para dar comandos.

El comando para salir es ":" seguido de "que" y ENTER

--con este comando saldrá si no ha hecho cambios al archivo o los cambios ya están salvados, para salir ignorando cambios ingrese ":q!" seguido de ENTER.

- Puede insertar texto (pasar a modo edición) con varias teclas:
  - • i
- Inserta texto antes del carácter sobre el que está el cursor.
  - • a
- Inserta texto después del carácter sobre el que está el cursor.
  - • I
- Inserta texto al comienzo de la línea en la que está el cursor.
  - • A
- Inserta texto al final de la línea en la que está el cursor.
  - • o
- Abre espacio para una nueva línea después de la línea en la que está el cursor y permite insertar texto en la nueva línea.
  - O

Análogo al anterior, pero abre espacio en la línea anterior.
Para pasar de modo edición a modo de comandos se emplea la tecla ESC, para desplazarse sobre el archivo puede emplear las flechas, PgUp, PgDn, también se pueden utilizar las teclas j (abajo), k (arriba), h (izquierda) y l (derecha).
MC: Es un editor de texto como el EDIT del DOS.
CAT: Se utiliza para concatenar archivos y mostrarlos por la salida estándar (normalmente la pantalla).
MORE: es un filtro para paginar texto, mostrando una pantalla cada vez. Esta versión es especialmente primitiva. Los usuarios deben tener en cuenta que **less** ("menos") provee, **more** ("más") emulación y muchas más mejoras.

(c) Cree un archivo llamado "prueba.exe" en su directorio personal usando el "vi". El mismo debe contener su número de alumno y su nombre.
(d) Investigue el funcionamiento del comando file. Pruébelo con diferentes archivos. ¿Qué diferencia nota?

10. Indique qué comando es necesario utilizar para realizar cada una de las siguientes acciones. Investigue su funcionamiento y parámetros más importantes:

(a) Cree la carpeta redictadoISO

- mkdir *nombre*: crea una nueva carpeta con el nombre especificado

(b) Acceda a la carpeta (cd)
(c) Cree dos archivos con los nombres redictadoiso-1 y redictadoiso-2 (touch)
(d) Liste el contenido del directorio actual (ls)
(e) Visualizar la ruta donde estoy situado (pwd)
(f) Busque todos los archivos en los que su nombre contiene la cadena "iso*" (find)

- *carpeta_dnd_busco* find /home/..ruta.. / -name "*XXX*"

(g) Informar la cantidad de espacio libre en disco (df -h)
(h) Verifique los usuarios conectado al sistema (who)
(i) Acceder a el archivo redictadoiso-1 e ingresar Nombre y Apellido
(j) Mostrar en pantalla las últimas líneas de un archivo (tail)

11. Investigue su funcionamiento y parámetros más importantes:

(a) shutdown (root/ systemctl +)
(b) reboot (root/ systemctl +)

(c) halt

(d) locate: provee una fácil y rápida manera de buscar archivos en el sistema completo basado en patrones de nombres.

(e) uname = ok = "linux"

(f) gmesg : El comando dmesg es usado con el fin de escribir los mensajes del kernel en Linux y otros sistemas operativos similares a Unix en una salida estándar de forma mucho más organizada.

(g) lspci ok, = info

(h) at

(i) netstat

(j) mount

(k) umount

(l) head

(m) losetup

(n) write (ñ) mkfs

(o) fdisk (con cuidado) y fisk

12. Investigue su funcionamiento y parámetros más importantes:

(a) Indique en qué directorios se almacenan los comandos mencionados en el ejercicio anterior.