1. **Acerca de GNU/Linux.**

* Es case sensitive.
* Es multiusuario y multitarea.
* Tiene shells programables.
* Desde GNU/Linux se puede acceder a los File Systems de Windows.
* Su kernel es de libre distribución.
* En 1992, Torvalds y Stallman, deciden fusionar los proyectos Linux y GNU con el fin de crear GNU/Linux.

1. **Acerca del kernel de GNU/Linux.**

* Es el encargado de interactuar con el intérprete de comandos en base a las respuestas recibidas por los dispositivos de hard.
* Posee licencia de tipo GPL.
* Podemos interactuar con él mediante el Shell.
* Es posible tener más de una imagen del kernel compilado en el S.O
* Su código fuente está disponible.
* Entre sus responsabilidades, entra la administración de la memoria y la CPU.
* El objetivo de los gestores de arranque es cargarlo en memoria.
* La versión estable actual es la 3.5.X
* Es monolítico hibrido.
* La imagen del kernel según el FHS se encuentra ubicada en /boot dentro del filesystem.

1. **Acerca de la administración de discos rígidos.**

* En los discos IDE existe el concepto de Master/Slave.
* Siempre se debe definir un punto de montaje para la partición raíz (/).

1. **Acerca de las siguientes instrucciones del intérprete de comandos de GNU/Linux.**

* Pwd devuelve la ruta completa del directorio donde el usuario se encuentra ubicado al ejecutar el comando.
* Df muestra el tamaño y espacios libres y ocupados de las particiones montadas.
* Who devuelve el nombre de los usuarios logueados actualmente.
* Whoami muestra el nombre de usuario del usuario actual con el que se encuentra logueado.
* Chmod permite manejar los permisos que un grupo tiene sobre un archivo.
* Grep busca archivos por contenido.
* Find busca archivos por nombre.
* Init 0 apaga la máquina.
* Init 6 reinicia la máquina.
* Halt apaga la máquina.
* Reboot reinicia la máquina.
* Shutdown -h now apaga la máquina.
* Shutdown -r now reinicia la máquina.
* Si sabemos el nombre de un proceso podemos matarlo con el comando KILLALL.
* El objetivo del comando CAT es ver el contenido de un archivo.
* ‘ ‘ comillas simples se usan para sustitución.
* # se usa para declarar comentarios.

1. **Con respecto a la utilización de archivos en GNU/Linux.**

* El comando file permite identificar su tipo.
* Se pueden visualizar con el comando cat <nombre\_del\_archivo>.
* Linux soporta diversos tipos de File Systems.
* En /dev encontramos archivos de unidades, que representa dispositos.
* En /etc encontramos archivos de configuración.
* En /home están las carpetas personales de los usuarios.
* En /bin encontramos arhivos binarios ejecutables.
* En /tmp encontramos archivos temporales.
* En /lib encontramos librerías.
* En /var encontramos variables de archivo.

1. **Acerca de la utilización de virtualizadores y emuladores.**

* Permiten hacer creer al S.O que se instala sobre ellos, que el mismo corre en una maquina dedicada.
* Permiten que un equipo pueda correr varios S.O
* Agregan una capa adicional que intercepta las llamadas del S.O emulados/virtualizados y las envía al S.O donde corre el emulador/virtualizador.

1. **Acerca de la instalación de GNU/Linux.**

* Se puede realizar desde un CD o a través de la red, entre otros métodos de instalación.
* El /home puede estar en una partición aparte.
* En el MBR se almacena información que permite arrancar el S.O
* En un mismo equipo pueden convivir varios S.O
* El MBC se almacena en el MBR en el byte 446.
* El instalador a utilizar depende de la arquitectura donde quiera instalar la distribución.

1. **Acerca del manejo de usuarios en GNU/Linux.**

* Cada usuario pertenece al menos a un grupo.
* Las contraseñas de los usuarios se almacenan en el archivo /etc/shadow. (encriptada)
* En el archivo etc/passwd se encuentra el GID principal al que está asociado cada usuario.
* Para agregar un usuario al sistema se puede utilizar el comando adduser o useradd.
* El directorio personal (nombre de los usuarios, uid, gid, interprete de comandos) de los usuarios se especifica en el archivo /etc/passwd.
* Mediante el comando usermod podemos modificar atributos de los usuarios (ej:grupo).

1. **Acerca de los niveles de ejecución de GNU/Linux.**

* El nivel de ejecución 0 hace referencia a HALT (modo de parada).
* El nivel de ejecución 1 hace referencia al modo monousuario.
* El nivel de ejecución 2 hace referencia al modo multiusuario sin red.
* El nivel de ejecución 3 hace referencia a un modo multiusuario con funciones de red.
* El nivel de ejecución 5 hace referencia a un modo multiusuario gráfico.
* El nivel de ejecución 6 hace referencia a REBOOT (modo de reinicio).
* Los runlevels 2 al 5 hacen referencia a modos multiusuario.

1. **Acerca del archivo /etc/inittab.**

* Posee información del nivel de ejecución por defecto.
* Es el encargado de configurar el proceso init según system V.

1. **Acerca del uso de empaquetadores y compresores en GNU/Linux.**

* Para empaquetar se utiliza el comando tar.
* Uno de los comandos que permiten comprimir un archivo es el gzip.
* Usar “tar -czvf resultadonombrearchivo.tar.gz (nombredirectorioacomprimiryempaquetar)” para empaquetar y comprimir todo el contenido de un directorio.

1. **Acerca de la sentencia exit dentro de un script.**

* Se puede averiguar su valor de retorno consultando la variable $?
* Puede devolver cualquier valor entre 0 y 255.
* Se utiliza para causar la terminación de un script.

1. **Acerca de los procesos en GNU/Linux.**

* Todo proceso posee un PID (su identificador).
* Todo proceso pertenece a un usuario del sistema.
* Todos los procesos tienen un proceso padre, menos el proceso init.
* El proceso init tiene PID1.
* Puede visualizarse cuales se encuentran en ejecución a través del comando top.
* Si al ejecutarlo, en la línea de ejecución se le agrega un & al final, se lo ejecuta en background.
* Un proceso que se ejecuta en background puede ser pasado a foreground a través de comando fg.

1. **Acerca del proceso init.**

* Es el padre de todos los procesos.
* Es el encargado de montar los file system.
* Según system V se lo configura a través del archivo /etc/inittab.
* En el arranque basado en systemd es reemplazado por el proceso-demonio “systemd”.
* Posee PID 1.

1. **Acerca de procesos UNIX.**

* Podemos desviar la salida estándar de un proceso a un archivo.
* El pipe (|) nos permite comunicar procesos.
* Podemos desviar la salida estándar de un proceso a la entrada de otro con append (>>).
* Cada proceso posee un PID único.

1. **Acerca de la ejecución de procesos en background.**

* Se lanza su ejecución agregando al final de su invocación un & (ampersand)
* Un proceso que se ejecuta en background puede ser pasado a foreground a través de comando fg.
* Para pasar un proceso en ejecución al background se utiliza el comando bg.
* A través del comando Jobs se puede ver el estado de los procesos que se ejecutan en background.

1. **Acerca del manejo de permisos en GNU/Linux.**

* Se pueden asignar permisos de lectura, escritura y ejecución por separado.
* Se le dan permisos al propietario, grupo y resto.
* Puede utilizarse la notación octal.
* Se manejan con el comando CHMOD.
* Si un archivo tiene el permiso 550 puede ser leído y/o ejecutado por el dueño y su grupo.
* Los permisos se aplican sobre archivos y directorios.

1. **Acerca del proceso de arranque de una máquina.**

* Se denomina Bootstrap.
* En las arquitecturas x86 el responsable de iniciar la carga del SO a través del MBC es el BIOS.
* El gestor de arranque es ejecutado por el BIOS.
* El gestor de arranque se encarga de ejecutar el kernel.
* La última acción del BIOS es leer el MBC. Lo lleva a memoria y lo ejecuta.
* Solo puede haber designado un único disco como Primary Máster Disc.

1. **Acerca del arranque basado en MBR.**

* Se ubica en el cilindro 0, cabeza 0, sector 1.
* Existe un MBR en todos los discos.
* Ocupa 512 bytes.
* Los primeros bytes corresponden al Máster Boot Code (MBC)
* A partir del byte 446 está la tabla de particiones. Es de 64 bytes
* Al final existen 2 bytes libres.
* El MBC está contenido en el MBR.

1. **Acerca de Shell Scripting.**

* Es un archivo de texto que contiene sentencias del intérprete de comandos.
* Por defecto, el alcance de una variable es global.
* Se pueden definir funciones dentro de un script.
* La función test permite evaluar expresiones condicionales.
* A través del comando source puede incluirse código de otros scripts dentro de un script.
* La ejecución de un script a través del siguiente comando bash -x script permite su ejecución en modo debug.

1. **Acerca de insserv.**

* Se utiliza para actualizar y manejar el orden de los enlaces simbólicos del /etc/rcX.d de forma dinámica.
* Alguna de las opciones que utiliza son Default-Start, Requerid-Start y Default-Stop.
* Mejora la performance del arranque en sistemas multiprocesadores.
* Utiliza cabeceras en los scripts del /etc/init.d

1. **Acerca del manejo de particiones.**

* Cada partición puede ser formateada con un tipo de file system distinto.
* En un disco se pueden crear 3 particiones primarias y una extendida.
* Las particiones extendidas de dividen en volúmenes lógicos.
* Puede haber como máximo 3 particiones primarias y una extendida.

1. **Acerca del upstart.**

* Es asíncrono.
* Es un reemplazo de System V y es compatible con él.

1. **Acerca del proceso de arranque basado en System V.**

* El proceso de arranque se divide en niveles.
* Los scripts que se ejecutan durante el arranque están en /etc/init.d
* En /etc/rcX.d(donde x=0..6) hay links a scripts que se ejecutan en cada nivel.

1. **El FHS hace referencia a:**

* Un estándar para particionar sistemas Unix.
* Un estándar para organizar archivos y directorios archivos del file system.

1. **Acerca del arranque de una PC con UEFI.**

* Se puede bootear con el MBR legado.
* El gestor de arranque, ej:Grub, debe ser una aplicación UEFI instalada en el UEFI File System.
* UEFI usa GPT para el particionado.
* UEFI es la realización de la propuesta EFI propuesta por Intel.
* UEFI es compatible con el MBR tradicional.

1. **Orden correcto de la secuencia de booteo del S.O**

* 1. Se ejecuta el código de la BIOS. 2.El hardware lee el sector de arranque. 3. Se carga el sector de arranque. 4. Se carga el kernel.

1. **Acerca del uso de variables en Shell Scripting.**

* Echo ${dirección}
* Echo $nombre
* Dirección=”56 nro 456”
* La sustitución de comandos permite utilizar la salida de comandos como si fuera texto.
* $(ls) permite sustituir el comando ls por su resultado.
* ‘ls’ permite sustituir el comando por su resultado.
* $# informa la cantidad de parámetros enviados, recibidos.
* $\* contiene una lista de argumentos recibidos.
* $? Contiene en todo momento el valor de retorno del ultimo comando ejecutado.

1. **Acerca de systemd.**

* Centraliza la administración de demonios y librerías.
* Los runlevels de system V se reemplaza por targets.