Práctica

Aprendiendo Practicando

GNU/Administración

Ing. José Paredes

www.codigolibre.org y www.acl.edu.do

Aprendiendo Practicando GNU/Administración

En esta práctica usted aprenderá:Procesos, Daemons, Runlevels, Init, Sistema de Archivos, Permisos, Enlaces, Inodo, Administración Usuarios y Grupos, Programar Tareas y Backups, Registros del Sistema. Si completas el 100% de estos ejercicios podremos garantizar su aprendizaje.

Procesos

1. Estados de los procesos.

D:	Suspendido no interrumpible (generalmente esperando E/S)
R:	En ejecución o listo para ejecutarse (en cola)
S:	Suspendido interrumpible (esperando que se complete un evento)
T:	Detenido, por una señal de control de trabajo o porque está siendo traceado
W:	Paginado (no válido a partir del kernel 2.6.xx)
X:	Muerto (nunca debe ser visto)
Z:	Proceso difunto ("zombie"), terminado pero no reclamado por el proceso padre

2. Ver dónde se almacenan todos los procesos.

```
[estudiantes@acl ~]$ ls /proc
```

3. Ver el contenido de un proceso en /proc.

```
[estudiantes@acl ~]$ ls /proc/1/
```

4. Para saber el nombre del proceso 1, desde /proc.

```
[estudiantes@acl ~]$ cat /proc/1/cmdline
```

5. Vemos el estatus del proceso 1, desde /proc.

```
[estudiantes@acl ~]$ cat /proc/1/status
```

6. Muestra los procesos en forma de árbol.

```
[estudiantes@acl ~]$ pstree
```

7. Despliega los procesos de tu shell.

```
[estudiantes@acl ~]$ ps
```

8. Para ver un resumen de las opciones del ps

```
[estudiantes@acl ~]$ ps --help
```

9. En el manual de ps se explican los diferentes valores y opciones:

```
[estudiantes@acl ~]$ man ps
```

10. Muestra todo los proceso de un usuario.

```
[estudiantes@acl ~]$ ps -U estudiantes
```

11. Imprimir un árbol de procesos.

```
[estudiantes@acl ~]$ ps -ejH
```

```
[estudiantes@acl ~]$ ps -axjf
```

12. Obtener información de hilos.

```
[estudiantes@acl ~]$ ps -eLf
```

```
[estudiantes@acl ~]$ ps axms
```

13. Muestra todo los procesos en ejecución.

```
[estudiantes@acl ~]$ ps -aux |less
```

GNUNota: La terminal nos arrojara una tabla dividida en 11 columnas, cada columna con un dato en especifico, así que les explicare de manera rápida cada dato que nos arrojara...

USER:	Muestra de quien es el proceso
PID:	Significa Identificación del numero del proceso(Process Identification Number)
%CPU:	Porcentaje de la CPU que esta tomando el proceso.
%MEM:	Porcentaje de memoria que esta tomando el proceso.
VSZ:	La cantidad de memoria virtual que esta ocupando tal proceso.
RSS:	La cantidad de la memoria residente que esta tomando el proceso.
TTY:	El control de un proceso en terminal.
STAT:	Estado del proceso, y los posibles estados del proceso son: s=el proceso esta durmiendo, R= Proceso alojado en la CPU, D=ininterrumplieble de dormir, T=Proceso tuvo un error o fue detenido, Z=proceso en Zombie(Ni muerto ni vivo!)
START:	Fecha en el que el proceso empezó.
TIME:	Tiempo que el proceso lleva alojado en el CPU
COMMAND:	Nombre del proceso y sus parámetros de la linea de comandos.

14. Muestra las tareas Linux en un modo jerárquico.

```
[estudiantes@acl ~]$ ps -e -o pid,args --forest | less
```

15. Obtener información de seguridad.

```
[estudiantes@acl ~]$ ps -eo euser,ruser,suser,fuser,f,comm,label
```

```
[estudiantes@acl ~]$ ps axZ
```

```
[estudiantes@acl ~]$ ps -eM
```

16. Visualiza todos los procesos que no sean del usuario root.

```
[estudiantes@acl ~]$ ps -u root -N
```

17. Ver todos los procesos ejecutando como root (ID efectivo y real) en formato de usuario:

```
[estudiantes@acl ~]$ ps -U root -u root
```

18. Imprime por pantalla todo los demonios en ejecución.

```
[estudiantes@acl ~]$ ps -t ?
```

19. Entre muchas opciones que tiene ps, es posible ordenar la salida de acuerdo a una columna, para esto se debe utilizar la opción gnu larga "--sort", por ejemplo para ordenar por tiempo de CPU (si deseamos determinar qué proceso ha utilizado más CPU) seleccionamos el código "cputime", correspondiente a la columna TIME:

```
[estudiantes@acl ~]$ ps aux --sort cputime
```

20. Muestra los procesos ordenados por uso de memoria (los que más memoria usan al final).

```
[estudiantes@acl ~]$ ps -e -orss=,args= | sort -b -k1,1n | pr -TW$COLUMNS
```

21. Es posible alterar el comportamiento de ps utilizando variables de entorno, la más significativa es "PS_PERSONALITY" la cual afecta la "personalidad" de ps. Se puede setear "PS_PERSONALITY" utilizando export y observar los resultados.

```
[root@acl ~]$ export PS_PERSONALITY=bsd
```

```
[root@acl ~]$ ps
```

22. Muestra los procesos de forma dinámica.

```
[estudiantes@acl ~]$ top
```

GNUNota: Dentro del programa podemos interactuar con el con varias opciones:

k:	Si se quiere matar el proceso, luego debemos ingresar el numero de su PID.
r:	Cambia la prioridad del proceso
O:	Muestra las posibles columnas que podemos agregar a la lista de procesos
1:	Muestra la información de todos los cores
Z o b:	Agregan colores a la interfaz
c:	Muestra el path absoluto del binario que se esta ejecutando.
n:	Nos permite reducir la lista a "n" procesos.
N:	Ordena los procesos por PID
A:	Ordena los procesos por aparición, primero se encuentran los mas nuevos
P:	Ordena los procesos por uso de CPU, esta opción es la default
M:	Ordena los procesos por memoria residente
T:	Ordena los procesos por tiempo.
W:	Guarda la configuración que hicimos
q:	Salir de Top

23. Muestra los procesos que estan corriendo con ese usuario y sus valores

```
[estudiantes@acl ~]$ top -u estudiantes
```

24. Muestra el proceso seleccionado y sus valores

```
[estudiantes@acl ~]$ top -p 1
```

25. Especificaremos el número de veces que actualizará hasta que finalice la ejecución de Top.

```
[estudiantes@acl ~]$ top -n4
```

26. Visualizar la línea de comando completa de cada proceso.

```
[estudiantes@acl ~]$ top -c
```

27. Es una versión más actualizada del comando top.

```
[estudiantes@acl ~]$ htop
```

GNUNotas: Explique la diferencia con top.

28. Visualiza los estados de los procesos ejecutados en background o segundo plano.

```
[estudiantes@acl ~]$ jobs
```

29. Para enviar un proceso a segundo plano lo llamamos desde la linea de comandos seguido de '&'.

```
[estudiantes@acl ~]$ charmap &
```

```
[estudiantes@acl ~]$ jobs
```

GNUNota: **Ya lo hemos mandado a segundo plano y podemos seguir usando la consola, pero ¿que es el [1] 20556? el [1] es el numero de tarea del proceso yes, si volviéramos a ejecutar el mismo comando el numero seria [2].**
20556 es el PID, el numero que el sistema asigna a cada proceso, estos dos números nos servirán para identificar al proceso mas tarde.

30. También podemos usar el comando ps.

```
[estudiantes@acl ~]$ ps
```

31. Llama una aplicación por el terminal. Ejemplo gedit.

```
[estudiantes@acl ~]$ gedit
```

32. Vamos enviar la aplicación a segundo plano.

```
[estudiantes@acl ~]$ Ctrl + z
```

```
[estudiantes@acl ~]$ jobs
```

GNUNota: Explique al GNU/Instructor la diferencia de CTRL+Z y &.

33. Para saber las tareas en segundo plano con sus PID.

```
[estudiantes@acl ~]$ jobs -l
```

34. El proceso que está en segundo plano detenido, vamos ponerlo a correr en segundo plano.

```
[estudiantes@acl ~]$ bg
```

```
[estudiantes@acl ~]$ jobs
```

35. Si queremos enviar a primer plano el proceso que está en segundo plano detenido.

```
[estudiantes@acl ~]$ fg %2
```

36. Si queremos cerrar el proceso que esta en primer plano.

```
[estudiantes@acl ~]$ Ctrl + c
```

```
[estudiantes@acl ~]$ jobs
```

37. Muestra sólo el identificador de proceso para los trabajos listados, su PID en la tabla de procesos.

```
[estudiantes@acl ~]$ jobs -p
```

38. Listado de todas las señales disponibles para el comando kill.

```
[estudiantes@acl ~]$ kill -l
```

39. Vamos enviar un proceso ejecutándose en segundo plano.

```
[estudiantes@acl ~]$ gnome-calculator &
```

```
[estudiantes@acl ~]$ jobs
```

40. Vamos detener el proceso numero 1 “charmap” que esta corriendo en background, enviando señal por su numero de tarea.

```
[estudiantes@acl ~]$ kill -19 %1
```

```
[estudiantes@acl ~]$ jobs
```

GNUNota: Trate de utilizar el proceso “charmap”.

41. Indicamos al proceso que esta detenido que vuelva ejecutarse, enviando señal por su numero de tarea.

```
[estudiantes@acl ~]$ kill -18 %1
```

```
[estudiantes@acl ~]$ jobs
```

GNUNota: Trate de utilizar el proceso “charmap”.

42. Podemos matar(terminar) el proceso con kill, seguido por % y el numero de trabajo en background.

```
[estudiantes@acl ~]$ kill -1 %1
```

```
[estudiantes@acl ~]$ jobs
```

43. Vamos detener el proceso numero 2 “gedit” que esta corriendo en background, enviando la señal por su nombre, al numero de tarea.

```
[estudiantes@acl ~]$ kill -STOP %2
```

```
[estudiantes@acl ~]$ jobs
```

GNUNota: Trate de utilizar gedit.

44. Indicamos al proceso que esta detenido que vuelva ejecutarse, enviando señal por su nombre, al numero de tarea.

```
[estudiantes@acl ~]$ kill -CONT %2
```

```
[estudiantes@acl ~]$ jobs
```

GNUNota: Trate de utilizar gedit.

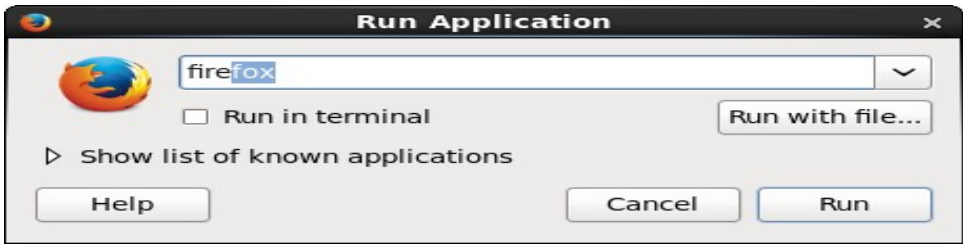
45. Podemos matar(terminar) el proceso numero 2 “gedit”, utilizando el nombre de la señal(9).

```
[estudiantes@acl ~]$ kill -l
```

```
[estudiantes@acl ~]$ kill -KILL %2
```

```
[estudiantes@acl ~]$ jobs
```

46. Vamos a correr una aplicación y buscar su proceso, ALT+F2 escribimos el nombre de la aplicación, estará en background?.



47. Vamos buscar el PID de la aplicación “firefox” .

```
[estudiantes@acl ~]$ jobs
```

```
[estudiantes@acl ~]$ ps -aux | grep  firefox |grep -v grep
```

GNUNota: Explique al GNU/Instructor que función hace “grep -v”.

48. Vamos buscar el PID del proceso “firefox”, utilizando pgrep.

```
[estudiantes@acl ~]$ pgrep firefox
```

49. Vemos que se creo un directorio en /proc con el nombre de su PID.

```
[estudiantes@acl ~]$ ls /proc/10835
```

50. Comprobemos el nombre del proceso.

```
[estudiantes@acl ~]$ cat /proc/10835/cmdline
```

51. Ahora enviemos la señal -19, “Parar el proceso”.

```
[estudiantes@acl ~]$ Kill -19 10835 → Este es el PID del proceso
```

GNUNota: Trate de utilizar la aplicación.

52. Ahora enviamos la señal -18 “Continua el proceso”.

```
[estudiantes@acl ~]$ Kill -18  10835
```

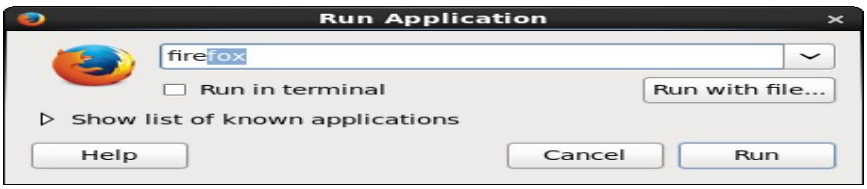
GNUNota: Trate de utilizar la aplicación.

53. Terminemos el proceso de manera forzosa

```
[estudiantes@acl ~]$ kill -9 10835
```

```
[estudiantes@acl ~]$ ps -aux | grep  firefox |grep -v grep
```

54. Iniciemos el proceso y enviemos la señal 15 para terminar el proceso, ALT+F2 “firefox”.



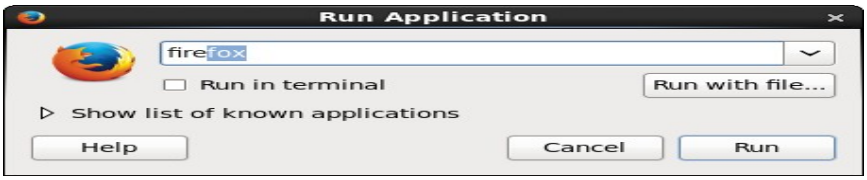

```
[estudiantes@acl ~]$ pgrep firefox
```

```
[estudiantes@acl ~]$ kill -15 12426
```

```
[estudiantes@acl ~]$ ps -aux | grep  firefox |grep -v grep
```

GNUNota: Explicar al GNU/Instructor porque cambio su PID.

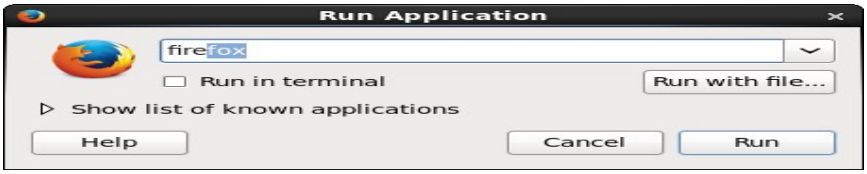
55. Iniciemos un proceso, ALT+F2 “firefox” y enviemos una señal utilizando “pkill” para terminar el proceso.




```
[estudiantes@acl ~]$ pkill firefox
```

```
[estudiantes@acl ~]$ ps aux | grep  firefox |grep -v grep
```

56. Iniciemos un proceso, ALT+F2 “firefox” y enviemos una señal utilizando “killall” para terminar el proceso.

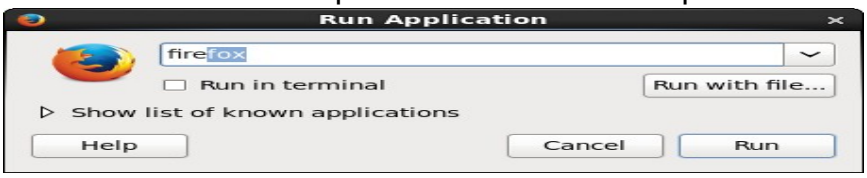



```
[estudiantes@acl ~]$ killall firefox
```

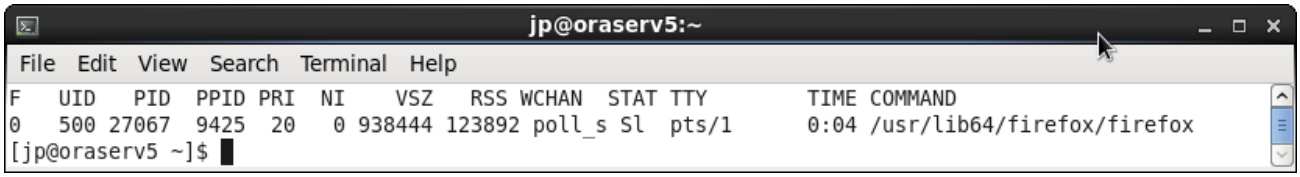
```
[estudiantes@acl ~]$ ps aux | grep  firefox |grep -v grep
```

GNUNota: Investigue y practique varias señales “kill -l”.

57. Vamos observar las prioridades de un proceso.




```
[estudiantes@acl ~]$ ps al
```



GNUNota: Vemos la prioridad del proceso 27067 → “firefox” en la columna “PRI” es 20.

58. Obtener la ayuda del comando nice.

```
[estudiantes@acl ~]$ nice --help
```

59. Inicia proceso con prioridad 0.

```
[estudiantes@acl ~]$ charmap &
```

```
[estudiantes@acl ~]$ ps al
```

GNUNota: Vemos la columna “NI”.

```
[estudiantes@acl ~]$ killall charmap
```

60. Inicia proceso con prioridad 10.

```
[estudiantes@acl ~]$ nice charmap &
```

```
[estudiantes@acl ~]$ ps al
```

GNUNota: Vemos la columna “NI”.

```
[estudiantes@acl ~]$ killall charmap
```

61. Inicia proceso con prioridad 5.

```
[estudiantes@acl ~]$ nice -n 5 charmap &
```

```
[estudiantes@acl ~]$ ps al
```

GNUNota: Vemos la columna “NI”.

```
[estudiantes@acl ~]$ killall charmap
```

62. Intenta iniciar proceso con prioridad negativa -5 y nos da un error por no tener permisos, deja prioridad 0.

```
[estudiantes@acl ~]$ nice -n -5 charmap &
```

```
[estudiantes@acl ~]$ ps al
```

GNUNota: Vemos la columna “NI”.

```
[estudiantes@acl ~]$ killall charmap
```

63. Obtener la ayuda del comando renice.

```
[estudiantes@acl ~]$ renice --help
```

64. Cambiamos la prioridad del proceso a prioridad negativa -5, usando root.

```
[estudiantes@acl ~]$ su -
```

```
[root@acl ~]$ charmap &
```

```
[root@acl ~]$ renice -n -5 PID &
```

```
[estudiantes@acl ~]$ ps al
```

GNUNota: Vemos la columna “NI”.

65. Si ejecutamos el comando nice sin parámetros nos indicará el valor asignado actualmente para nice:

```
[estudiantes@acl ~]$ su -
```

```
[root@acl ~]$ nice
```

66. Ejecutar un proceso con la mayor prioridad disponible (en este caso sera firefox haríamos lo siguiente.

```
[root@acl ~]$ pgrep firefox
```

```
[root@acl ~]$ renice -n -13 PID &
```

```
[root@acl ~]$ ps al | grep PRI | grep -v grep  && ps al |grep firefox |grep -v grep
```

67. Vamos modificar la prioridad de un proceso que tiene como nombre firefox pero desde su PID, utilizando el comando “renice”.

```
[root@acl ~]$ renice +3 29524
```

GNUNota: 29524: old priority -13, new priority 3.

```
[root@acl ~]$ ps al | grep PRI | grep -v grep  && ps al |grep firefox |grep -v grep
```

68. Mostrar una lista de ficheros abiertos por procesos.

```
[estudiantes@acl ~]$ lsof -p $$
```

69. Mostrar una lista de ficheros abiertos por un directorio.

```
[estudiantes@acl ~]$ lsof +D /var/log
```

70. Lista de archivos abiertos en base a los nombres de procesos que comienzan con ssh y init.

```
[estudiantes@acl ~]$ lsof -c ssh -c init
```

71. Mostrar las llamadas del sistema hechas y recibidas por un proceso.

```
[estudiantes@acl ~]$ strace -c ls >/dev/null
```

72. Mostrar las llamadas a la biblioteca.

```
[estudiantes@acl ~]$ strace -f -e open ls >/dev/null
```

73. Mostrar interrupciones en tiempo real.

```
[estudiantes@acl ~]$ watch -n1 'cat /proc/interrupts'
```

74. Puede haber uno o muchos segmentos de memoria compartida por proceso. Los procesos intercambian mensajes entre ellos (“comunicaciones entre procesos”, o IPC) y utilizan semáforos. Para desplegar información sobre segmentos de memoria compartida, colas de mensajes IPC y semáforos, usted puede utilizar un solo comando: ipcs.

```
[estudiantes@acl ~]$ ipcs
```

75. Puede obtener un panorama detallado de un segmento de memoria compartida utilizando el valor shmid. La opción –i logra esto.

Usted verá los detalles de shmid “3702792“.

GNUNota: El valor es tomado del ejercicio anterior.

```
[estudiantes@acl ~]$ ipcs -m -i 71532612
```

GNUTips:) La opción “-m” es muy popular, muestra los segmentos de memoria compartida. Si no se especifica ninguna opción, el programa muestra un resumen de la información administrativa que se almacena para los semáforos, memoria compartida y colas de mensajes..

76. Si queremos ver estadísticas de uso de la memoria, cpu, lecturas/escrituras en disco.

```
[estudiantes@acl ~]$ vmstat 5 10
```

77. Desplegar el uso de cada CPU individualmente.

```
[estudiantes@acl ~]$ mpstat -P ALL 5 2
```

GNUTips:) El comando mpstat aparece primero sin diferencias con el informe de utilización de CPU producido por iostat:

78. Si queremos mostrar una descripción general de las estadísticas del CPU y E/S de disco.

```
[estudiantes@acl ~]$ iostat
```

79. Para crear un informe básico que muestre el uso de CPU y el porcentaje de tiempo gastado esperando E/S, ejecutamos sar sin ningún argumento.

```
[estudiantes@acl ~]$ sar
```

80. También se pueden desplegar datos en tiempo real, similar a vmstat o mpstat. Para obtener los datos cada 5 segundos, durante 10 veces, utilice:

```
[estudiantes@acl ~]$ sar 5 10
```

81. ¿Notó el valor “all” debajo el título CPU? Esto significa que las estadísticas fueron cargadas para todas las CPU. Para un sistema de procesador único, esto está bien; pero en sistemas con múltiples procesadores usted desea obtener las estadísticas tanto para cada CPU individual, como para el conjunto. La opción -P ALL cumple con esto.

```
[estudiantes@acl ~]$ sar -P ALL 2 2
```

82. El comando sar no solo es para estadísticas relacionadas con la CPU. También es útil obtener las estadísticas relacionadas con la memoria. La opción -r muestra la utilización de memoria extensiva.

```
[estudiantes@acl ~]$ sar -r
```

83. Para obtener un output similar para la actividad relacionada con el swapping, usted puede utilizar la opción -W.

```
[estudiantes@acl ~]$ sar -W
```

84. Para obtener las estadísticas de los dispositivos de disco, utilice la opción -d.

```
[estudiantes@acl ~]$ sar -d
```

85. Si queremos ver los ficheros abiertos por un proceso.

```
[estudiantes@acl ~]$ lsof -p 15939 -Este es el pid del proceso
```

86. Ver algunos de los ficheros abiertos asociados a las dependencia del programa.

```
[estudiantes@acl ~]$ ldd /usr/sbin/sshd
```

87. Actualiza las librerías utilizadas por el sistema, recomendable ejecutarlo cada vez que se instale un programa.

```
[root@acl ~]$ ldconfig
```

88. El comando sync sirve para forzar la grabación de los datos de la cache.

```
[root@acl ~]$ sync
```

GNUTips:) También es recomendado hacerlo antes de desmontar un dispositivo más storage, como una cámara fotográfica digital, un pendrive, o una pda. O bien si hubo modificaciones y movimiento de información (cp y mv por ejemplo) entre particiones, antes de desmontar.

89. Para cambiar la resolución de la pantalla desde la consola debemos utilizar el comando xrandr.

```
[root@acl ~]$ xrandr
```

GNUNota: Nos mostrará un listado de resoluciones y nos dirá cual es la actual.

Runlevels (Nivel de ejecución)

90. Para ver la tabla de los runlevel.

```
[root@acl ~]$ cat /etc/inittab
```

91. La configuración de los archivos init se encuentran en.

```
[root@acl ~]$ ls /etc/rc.d
```

92. Vemos la ayuda para el comando init.

```
[root@acl ~]$ init --help
```

93. Bajar a mono-usuario o single user.

```
[root@acl ~]$ init 1
```

94. Baja a multiusuario con soporte de red sin X11 (GRAFICO).

```
[root@acl ~]$ init 3
```

95. Ir al runlevel 5, multiusuario con soporte de red con X11 (GRÁFICO).

```
[root@acl ~]$ init 5
```

96. Vamos reiniciar nuestro server.

```
[root@acl ~]$ init 6
```

97. Si queremos apagar el sistema.

```
[root@acl ~]$ init 0
```

GNUNota: Recuerde que en algunas distros usted debe ser root.

98. Vemos la ayuda para el comando telinit.

```
[root@acl ~]$ telinit --help
```

99. Pasa al runlevel 6 en 10 segundos (reinicia la máquina).

```
[root@acl ~]$ telinit -t 10 6
```

100. Vemos la ayuda para el comando initctl.

```
[root@acl ~]$ initctl --help
```

101. Para ver todos los servicios advenedizos registrados con init, ejecute lo siguiente.

```
[root@acl ~]$ initctl list
```

102. Es posible hacer que init relea el fichero de configuración /etc/inittab.

```
[root@acl ~]$ telinit q
```

103. Ver el listado de los servicios del sistema y en qué nivel se están ejecutando.

```
[root@acl ~]$ chkconfig --list
```

GNUNota: Para realizar este ejercicio y algunas mas debe ser root.

104. Verifiquemos que lo anterior se ha realizado con éxito.

```
[root@acl ~]$ chkconfig --list sshd
```

105. Desactivando el servicio ssh.

```
[root@acl ~]$ chkconfig sshd off
```

106. Activando el servicio ssh.

```
[root@acl ~]$ chkconfig sshd on
```

107. Bajemos el sshd los niveles 2,3 y 5.

```
[root@acl ~]# chkconfig --level 235 sshd off
```

108. Vamos a poner el servicio ssh que se ejecute al iniciar los runlevel 2,3 y 5.

```
[root@acl ~]$ chkconfig --level 235 sshd on
```

109. Verifiquemos que lo anterior sea realizado con éxito.

```
[root@acl ~]# chkconfig --list sshd
```

110. Verifica que el servicio vsftpd esté en la lista de servicios de chkconfig.

```
[root@acl ~]# chkconfig --list vsftpd
```

111. Si este servicio se encuentra en la lista de servicios, lo eliminados.

```
[root@acl ~]# chkconfig --del vsftpd
```

112. Verifiquemos que lo anterior se ha realizado con éxito.

```
[root@acl ~]# chkconfig --list vsftpd
```

113. Si este servicio no se encuentra en la lista de servicios agrégalo.

```
[root@acl ~]# chkconfig --add vsftpd
```

114. Verifiquemos que lo anterior se ha realizado con éxito.

```
[root@acl ~]# chkconfig --list vsftpd
```

115. Vamos a ver los servicios encendido que tenemos.

```
[root@acl ~]# nmap localhost
```

GNUNota: Si tenemos el ssh instalado tomaremos este para realizar los siguientes ejercicios, si no instalarlo:).

116. Para iniciar el servicio ssh.

```
[root@acl ~]# service sshd start
```

GNUNota: Repitamos el ejercicio 115 para ver si tenemos el servicio ssh iniciado con el puerto 22 abierto.

117. Para ver el estatus de un servicio.

```
[root@acl ~]# service sshd status
```

118. Ahora vamos detener el servicio sshd

```
[root@acl ~]# service sshd stop
```

GNUNota: Repitamos el ejercicio 115 para ver si tenemos el servicio ssh iniciado con el puerto 22 abierto.

119. Para reiniciar un servicio

```
[root@acl ~]# service sshd restart
```

120. También podremos iniciar los servicios desde /etc/init.d/.

```
[root@acl ~]# /etc/init.d/sshd start
[root@acl ~]# /etc/init.d/sshd stop
[root@acl ~]# /etc/init.d/sshd restart
[root@acl ~]# /etc/init.d/sshd status
```

121. Compruebe el estado de todos los servicios.

```
[root@acl ~]# service --status-all
```

122. Para poner un servicio autónomo en Debian.

```
[root@acl ~]# update-rc.d apache default
```

123. La mayor parte de usuarios del sistemas pueden comprobar el nivel de ejecución actual con cualquiera de los comandos siguientes.

```
[root@acl ~]# runlevel
```

```
[root@acl ~]# who -r
```

124. Mostrar quien está registrado, e imprimir hora del último sistema de importación, procesos muertos, procesos de registro de sistema, procesos activos producidos por init, funcionamiento actual y últimos cambios del reloj del sistema.

```
[root@acl ~]# who -a
```

125. Para ir desde el shell al singler user también lo puede hacer.

```
[root@acl ~]# init S
```

GNUNota: Investigue el comando ivoke-rc.d y update-rc.d.

Permisos

```
0  =  ---  =  sin acceso
1  =  --x  =  ejecución
2  =  -w-  =  escritura
3  =  -wx  =  escritura y ejecución
4  =  r--  =  lectura
5  =  r-x  =  lectura y ejecución
6  =  rw-  =  lectura y escritura
7  =  rwx  =  lectura, escritura y ejecución
```

GNUNota: Para iniciar estos ejercicios crearemos un directorio llamado “permisos” y dentro del mismo copiaremos passwd y inittab.

126. Verifica los permisos de los archivos en un directorio.

```
[root@acl permisos]# ls -l
```

127. Ve los detalles de un archivo en específico.

```
[root@acl permisos]# ls -l passwd
```

GNUNota: Explique al GNU/Instructor el detalle de esa salida.

128. Una información muy completa acerca de archivos o sistemas de ficheros

```
[root@acl permisos]# stat inittab
```

129. Nos muestra el User ID del propietario del fichero.

```
[root@acl permisos]# stat -c%u passwd
```

130. Nos muestra el nombre de usuario del propietario del fichero.

```
[root@acl permisos]# stat -c%U passwd
```

131. Nos muestra el Group ID del propietario del fichero.

```
[root@acl permisos]# stat -c%g passwd
```

132. Nos muestra los derechos de acceso en formato octal.

```
[root@acl permisos]# stat -c%a passwd
```

133. Nos muestra los derechos de acceso.

```
[root@acl permisos]# stat -c%A passwd
```

134. Vamos asignarle permiso de ejecución al archivo passwd

```
[root@acl permisos]# chmod +x passwd
```

GNUNota: Cuando ejecuto ls que hay de diferente con los demás archivos que hay en el directorio.

135. Vamos crear un archivo vacío llamado “permitir.txt”, luego veremos con los permisos que fue creado.

```
[root@acl permisos]# touch permitir.txt
```

```
[root@acl permisos]# stat permitir.txt
```

```
[root@acl permisos]# ls -l permitir.txt
```

GNUNota: Explique porque nació con esos permisos si usted no los asignó.

136. Vamos quitar todos los permisos al archivo passwd.

```
[root@acl permisos]# chmod 000 passwd
```

137. Agrégale permiso de escritura al usuario en el passwd.

```
[root@acl permisos]# chmod u+w passwd
```

GNUNota: Siempre que aplique cambios a un archivo verifique si los mismos fueron aplicados.

138. Quita el permiso de lectura al archivo inittab.

```
[root@acl permisos]# chmod u-r inittab
```

139. Trate de ver el contenido del archivo inittab.

```
[root@acl permisos]# cat inittab
```

140. Asigne a los usuarios que no son el dueño del archivo, ni pertenecen al grupo del archivo el permiso de ver el contenido del archivo.

```
[root@acl permisos]# chmod o+r passwd
```

141. Asigne todos los permisos al dueño y permisos de lectura y ejecución al resto.

```
[root@acl permisos]# chmod 755 passwd
```

142. Le otorga permisos de lectura y escritura al dueño y permisos de lectura al resto.

```
[root@acl permisos]# chmod go-rw,a+x passwd
```

143. Para ver los permisos del directorio utilizamos el comando ls, con la opción -d, para que no muestre el contenido, sino el directorio en si.

```
[root@acl permisos]# ls -ld /bin/
```

144. Vamos crear un directorio para ver con cuales permisos nace.

```
[root@acl permisos]# mkdir acl
```

Luego

```
[root@acl permisos]# ls -ld acl
```

GNUNota: Explique al GNU/Instructor cuáles permisos tiene el directorio creado.

145. Para ver la máscara de permiso por defecto.

```
[root@acl permisos]# umask
```

146. Para ver la máscara en formato simbólico.

```
[root@acl permisos]# umask -S
```

GNUNota:Lo anterior indica que un directorio y archivos ejecutables se crearán con los permisos 775 y los archivos comunes con los permisos 664. Esto se logra restando de 777 el valor de umask (777-002) y (666-002) respectivamente. El primer valor de umask corresponde para valores de Sticky bit, GUID o SUID, que por default es 0.

147. Para asignar permisos especiales en un archivo.

```
[root@acl permisos]# chmod 1777 inittab
```

148. El bit SUID o setuid se activa sobre un fichero añadiéndole 4000 a la representación octal de los permisos del archivo y otorgándole además permiso de ejecución al propietario del mismo.

```
[root@acl permisos]# chmod 4777 inittab
```

149. También puede activarse usando el modo simbólico.

```
[root@acl permisos]# chmod +s passwd
```

GNUNota: Los posibles valores serían los siguientes:

- - - - - - - - -	= 0	Predeterminado, sin permisos especiales. No se requiere indicar.
- - - - - - - - t	= 1	Bit de persistencia, sticky bit
- - - - s - - -	= 2	Bit sgid de grupo
- - - - s - - t	= 3	Bit sgid y sticky
- - s - - - - -	= 4	Bit suid
- - s - - - - - t	= 5	Bit suid y sticky
- - s - - s - - -	= 6	Bit suid y sgid
- - s - - s - - t	= 7	Bit suid, sgid y sticky

```
[root@acl permisos]# chmod 1644 passwd
```

```
[root@acl permisos]# ls -l passwd
```

```
[root@acl permisos]# stat passwd
```

```
[root@acl permisos]# chmod 2644 passwd
```

```
[root@acl permisos]# ls -l passwd
```

```
[root@acl permisos]# stat passwd
```

```
[root@acl permisos]# chmod 3644 passwd
```

```
[root@acl permisos]# ls -l passwd
```

```
[root@acl permisos]# stat passwd
```

```
[root@acl permisos]# chmod 4644 passwd
```

```
[root@acl permisos]# ls -l passwd
```

```
[root@acl permisos]# stat passwd
```

```
[root@acl permisos]# chmod 5644 passwd
```

```
[root@acl permisos]# ls -l passwd
```

```
[root@acl permisos]# stat passwd
```

```
[root@acl permisos]# chmod 6644 passwd
```

```
[root@acl permisos]# ls -l passwd
```

```
[root@acl permisos]# stat passwd
```

```
[root@acl permisos]# chmod 7644 passwd
```

```
[root@acl permisos]# ls -l passwd
```

```
[root@acl permisos]# stat passwd
```

150. Si queremos quitar el permiso.

```
[root@acl permisos]# chmod -s passwd
```

```
[root@acl permisos]# ls -l passwd
```

```
[root@acl permisos]# stat passwd
```

151. Este archivo lista todos los códigos de cifrado utilizados por el kernel de Linux, incluyendo detalles adicionales para cada uno.

```
[root@acl permisos]# cat /proc/crypto
```

«Chown»

152. Crea un directorio llamado pelo1.

```
[root@acl permisos]# mkdir pelo1
```

153. Verifica el propietario de este directorio.

```
[root@acl permisos]# stat pelo1
```

154. Cambia el propietario del directorio a estudiantes.

```
[root@acl permisos]# chown estudiantes pelo1
```

155. Luego de haber realizado el ejercicio anterior crea un subdirectorio llamado Cabeza.

```
[root@acl permisos]# mkdir pelo1/cabeza
```

156. Cambiemos el propietario de forma recursiva del directorio principal así como del subdirectorio.

```
[root@acl permisos]# chown -R estudiantes pelo1
```

157. Dentro del subdirectorio, crea un archivo llamado test.txt.

```
[root@acl permisos]# touch pelo1/cabeza/test.txt
```

158. Cambia el propietario y el grupo del archivo test.txt a root.

```
[root@acl permisos]# chown -R root:root pelo1/cabeza/test.txt
```

159. Comprueba que se ha realizado todo bien.

```
[root@acl permisos]# stat pelo1/cabeza/test.txt
```

«Chgrp»

160. Crea un archivo llamado test2.txt.

```
[root@acl permisos]# touch test2.txt
```

161. Verifica el grupo a que pertenece el archivo test2.txt.

```
[root@acl permisos]# stat test2.txt
```

162. Verifica que el grupo audio exista.

```
[root@acl permisos]# grep -i audio /etc/group
```

163. Cambia el grupo del archivo al grupo audio.

```
[root@acl permisos]# chgrp audio test2.txt
```

164. Verifica que todo ha salido con éxito.

```
[root@acl permisos]# stat test2.txt
```

165. Crea varios archivos dentro del directorio Cabeza.

```
[root@acl permisos]# cd pelo1/cabeza; touch arch1 arch2 arch3; cd  ../../
```

166. Verifica su estadística para ver a que grupo pertenecen los archivos creados.

```
[root@acl permisos]# stat pelo1/cabeza/arch*
```

167. Cambia el grupo de cada unos de estos archivos.

```
[root@acl permisos]# chgrp audio pelo1/cabeza/*
```

«Ln»

168. Crea un archivo llamado enlace.txt, de la salida de /etc/profile.

```
[root@acl permisos]# cat /etc/profile > enlace.txt
```

169. Creamos un enlace duro del archivo enlace.txt llamado enlace.ln.

```
[root@acl permisos]# ln enlace.txt enlace.ln
```

170. Verifica si este está apuntando al mismo inodo del archivo archivo original.

```
[root@acl permisos]# stat enlace.txt enlace.ln
```

171. Ahora borraremos el archivo llamado enlace.txt.

```
[root@acl permisos]# rm enlace.txt
```

172. Veremos qué pasa con el archivo enlace.ln.

```
[root@acl permisos]# cat enlace.ln
```

173. Creemos otro archivo llamado enlace2.txt.

```
[root@acl permisos]# cat /etc/group > enlace2.txt
```

174. Creamos un enlace suave del archivo enlace2.txt llamado enlace2.ln.

```
[root@acl permisos]# ln -s enlace2.txt enlace2.ln
```

175. Verifica a dónde está apuntando el enlace simbólico y si el inodo es el mismo que el del archivo original.

```
[root@acl permisos]# stat enlace2.*
```

o

```
[root@acl permisos]# ls -l
```

176. Ahora borremos el archivo original enlace2.txt.

```
[root@acl permisos]# rm enlace2.txt
```

177. Veremos qué pasa con el archivo enlace2.ln.

```
[root@acl permisos]# cat enlace2.ln
```

```
[root@acl permisos]# ls -l
```

```
[root@acl permisos]# file enlace2.ln
```

Sistema de archivos

178. Este archivo muestra los diversos dispositivos de caracteres y de bloque actualmente configurados (no incluye dispositivos cuyos módulos no están cargados).

```
[root@acl permisos]# cat /proc/devices
```

179. El archivo contiene información sobre la asignación de bloques de particiones.

```
[root@acl permisos]# cat /proc/partitions
```

180. Este archivo muestra una lista de los tipos del sistema de archivos soportados actualmente por el kernel.

```
[root@acl permisos]# cat /proc/filesystems
```

181. Este archivo graba el número de interrupciones por IRQ en la arquitectura x86.

```
[root@acl permisos]# cat /proc/interrupts
```

182. Este archivo muestra el mapa actual de la memoria del sistema para los diversos dispositivos.

```
[root@acl permisos]# cat /proc/iomem
```

183. La salida de /proc/ioports proporciona una lista de las regiones de puertos registrados actualmente utilizados para la comunicación de entrada y salida con un dispositivo.

```
[root@acl permisos]# cat /proc/ioports
```

184. Este archivo ofrece una vista de la carga promedio del procesador con respecto al sobre-tiempo de CPU y de E/S, así como también datos adicionales utilizados por uptime y otros comandos.

```
[root@acl permisos]# cat /proc/loadavg
```

185. Este archivo muestra los archivos bloqueados en la actualidad por el kernel.

```
[root@acl permisos]# cat /proc/locks
```

186. Este archivo contiene la información actual sobre la configuración de discos múltiples de RAID.

```
[root@acl permisos]# cat /proc/mdstat
```

GNUNota:Si su sistema no contiene dicha configuración, el archivo /proc/mdstat será parecido a: unused devices: <none>

187. Este es uno de los archivos más utilizados en el directorio /proc/, ya que proporciona mucha información importante sobre el uso actual de RAM en el sistema.

```
[root@acl permisos]# cat /proc/meminfo
```

188. Este archivo muestra una lista de todos los módulos cargados en el sistema.

```
[root@acl permisos]# cat /proc/modules
```

189. Este archivo virtual identifica el tipo de procesador usado por su sistema.

```
[root@acl permisos]# cat /proc/cpuinfo
```

190. Este archivo proporciona una lista de todos los montajes en uso por el sistema.

```
[root@acl permisos]# cat /proc/mounts
```

191. Este archivo se refiere a la actual Memory Type Range Registers (MTRRs), en uso dentro del sistema.

```
[root@acl permisos]# cat /proc/mtrr
```

192. Este archivo mantiene un registro de las diferentes estadísticas sobre el sistema desde que fue reiniciado por última vez.

```
[root@acl permisos]# cat /proc/stat
```

193. Este archivo mide el espacio swap y su uso.

```
[root@acl permisos]# cat /proc/swaps
```

194. El archivo contiene información sobre el tiempo que lleva encendido el sistema desde el último reinicio.

```
[root@acl permisos]# cat /proc/uptime
```

195. Este archivo muestra la versión del kernel de Linux y gcc en uso, así como la versión de Red Hat Enterprise Linux instalada en el sistema:

```
[root@acl permisos]# cat /proc/version
```

196. El fstab es el archivo donde se guardan los diferentes datos sobre el montaje de los dispositivos físicos.

```
[root@acl permisos]# cat /etc/fstab
```

197. En el mtab estan las líneas del fstab que corresponden a los puntos montados actualmente.

```
[root@acl permisos]# cat /etc/mtab
```

198. Si queremos ver el tamaño de todos los sistema de archivos montados actualmente.

```
[root@acl permisos]# df -h
```

GNUNota: Para el ejercicio siguientes debe tener una memoria USB conectada.

199. Crearemos un directorio en /mnt llamado usb y hay montaremos nuestro dispositivo usb.

```
[root@acl ~]# mkdir -p /mnt/usb
```

```
[root@acl ~]# mount /dev/sdb1 /mnt/usb/
```

GNUNota: Ejecute df -TH, para ver si el dispositivo está montado, luego entre al directorio /mnt/usb para ver su contenido.

200. Para desmontar el dispositivo.

```
[root@acl ~]# umount /mnt/usb
```

201. Si queremos darle otro formato al dispositivo.

```
[root@acl ~]# mkfs.ext3 /dev/sdb
```

GNUNotas1:

- **mkfs.ext3 : Formateo para ext3**
- **mkfs.ext4 : Formateo para ext4**
- **mkfs.ntfs : Formateo para ntfs**
- **mkfs.vfat : Formateo para fat**

GNUNota: Repita el ejercicio 199 para montar otra vez el dispositivo.

202. Veremos si está montado y qué sistema archivo tiene.

```
[root@acl ~]# df -Th
```

203. Para verificación y reparación de sistemas de archivos dañados.

```
[root@acl ~]# fsck /dev/sdb
```

GNUNota: El dispositivo debe estar desmontado.

Entra al menú de particionamiento de una memoria usb.

```
[root@acl ~]# fdisk /dev/sdb1
```

204. ¿Quieres guardar la salida de una página de manual como texto plano?

```
[root@acl ~]# man fsck | col -b > fsck.txt
```

```
[root@acl ~]# ls -l
```

```
[root@acl ~]# cat fsck.txt
```

205. Crea una tubería con un nombre asociado.

```
[root@acl ~]# mkfifo  tuberia
```

GNUNota: Esto hará que se cree un archivo con el contenido almacenado en la tubería y que la consola permanezca bloqueada hasta que se lea la salida de la tubería desde otra parte. Abrimos otra consola y escribimos:

```
[root@acl ~]# head  tuberia
```

206. Comprobamos que es un tipo de archivo Fifo.

```
[root@acl ~]# file  tuberia
```

```
[root@acl ~]# ls -l  tuberia
```

Administración de Usuario

207. Información de todos los usuarios del sistema y saber quién puede acceder al sistema de manera legítima.

```
[root@acl ~]# cat /etc/passwd
```

208. Chequear la sintaxis correcta, el formato de fichero de '/etc/passwd' y la existencia de usuarios.

```
[root@acl ~]# pwck /etc/passwd
```

209. Información de todos los grupos del sistema.

```
[root@acl ~]# cat /etc/group
```

210. Chequear la sintaxis correcta y el formato del fichero '/etc/group' y la existencia de grupos.

```
[root@acl ~]# grpck /etc/group
```

211. El archivo /etc/shadow contiene información sobre las contraseñas encriptadas de los usuarios del sistema.

```
[root@acl ~]# cat /etc/shadow
```

212. En el archivo de configuración /etc/login.defs están definidas las variables que controlan los aspectos de la creación de usuarios y de los campos de shadow usadas por defecto.

```
[root@acl ~]# cat /etc/login.defs
```

213. Para ver los valores por defecto de useradd.

```
[root@acl ~]# cat /etc/default/useradd
```

214. Proporciona una forma de estar seguro de que todos los nuevos usuarios de tu sistema **LFS** tienen la misma configuración inicial. El directorio /etc/skel es usado por el programa /usr/sbin/useradd.

```
[root@acl ~]# ls -a /etc/skel/
```

GNUNota: Si queremos que un usuario al ser creado contenga cierta variables o directorios, lo creamos en este directorio y al utilizar useradd estará disponible en cada usuario.

215. Contraseñas encriptadas de los grupos.

```
[root@acl ~]# cat /etc/gshadow
```

216. Existen tres ficheros en el directorio de un usuario que tienen un significado especial para el shell Bash. Estos ficheros permiten al usuario configurar el entorno de su cuenta automáticamente cuando entra en el sistema, cuando arranca un sub-shell o ejecutar comandos cuando sale del sistema.

Los nombres de estos ficheros son .bash_profile, .bashrc y .bash_logout.

```
[root@acl ~]# ls -a
```

217. El /etc/profile es utilizado por el sistema como fichero de configuración de bash.

```
[root@acl ~]# cat /etc/profile
```

218. Hacer login con la cuenta root sin cargar su variable de entorno, si era su desbloquearse.

```
[root@acl ~]# su - estudiantes
```

```
[estudiantes@acl ~]$ su
```

219. Una vez haya ejecutado la sentencia anterior, verifica quién eres.

```
[root@acl estudiantes]# whoami
```

220. Ahora verifica en qué lugar está.

```
[root@acl estudiantes]# pwd
```

221. Para salir o desloguearse del usuario root.

```
[root@acl estudiantes]# exit
```

222. Cámbiate a la cuenta root y cargar su variable de entorno.

```
[estudiantes@acl ~]$ su -
```

223. Ahora verifica en qué lugar está.

```
[root@acl ~]# pwd
```

224. Para el identificador actual y real de usuarios y grupos.

```
[root@acl ~]# id
```

225. El fichero shells contiene una lista de los intérpretes de comandos de ingreso en el sistema.

```
[root@acl]# cat /etc/shells
```

226. **C**rea un usuario llamado brianna de modo simple.

```
[root@acl]# useradd brianna
```

227. Comprueba que el usuario se ha creado.

```
[root@acl]# grep -i brianna /etc/passwd
```

228. Asignarle un password al usuario

```
[root@acl]# passwd brianna
```

229. Cambia el login del usuario brianna a dinora.

```
[root@acl]# usermod -l dinora brianna
```

230. Comprueba la fecha de espiración del usuario.

```
[root@acl]# chage -l dinora
```

231. Ponle un comentario al usuario que diga: “usuario de prueba”.

```
[root@acl]# usermod -c "usuario de prueba" dinora
```

232. Vamos a ver el comentario.

```
[root@acl ~]# grep -i "dinora" /etc/passwd
```

233. Vamos a cambiar el shell del usuario dinora.

```
[root@acl ~]# usermod -s /bin/sh dinora
```

GNUNota: Loguearse con el usuario dinora para comprobar su shell.

234. Vamos a mostrar la propiedad por defecto de los nuevos usuarios que se añadan.

```
[root@acl ~]# useradd -D
```

235. Vamos a crear el usuario orafcld con su UID 800

```
[root@acl ~]# useradd -u 800 orafcld
```

GNUNota:Inicie una sesión con el usuario orafcld y ejecute el comando id.

236. Crearemos un usuario llamado fcld con el comentario "Fundación Código Libre".

```
[root@acl ~]# useradd -c "Fundación Código Libre" fcld
```

237. Vamos a crear el usuario estudiantesaredes con su directorio de trabajo diferente a su login "fcld".

```
[root@acl ~]# useradd -d /home/fcld estudiantesaredes
```

238. Vamos a crear el usuario orapepe desactivando la posibilidad de ejecutar un shell.

```
[root@acl ~]# useradd -s /bin/false orapepe
```

239. Vamos a signar un password al usuario orapepe.

```
[root@acl ~]# passwd oraopepe
```

240. Vamos a deshabilitar la cuenta orapepe eliminando su password.

```
[root@acl ~]# passwd -d oraopepe
```

241. Vamos a bloquear la cuenta del usuario orapepe poniendo un signo ! delante de su password en el archivo /etc/shadow.

```
[root@acl ~]# passwd -l oraopepe
```

```
[root@acl ~]# su - oraopepe
```

242. Para desbloquear la cuenta del orapepe.

```
[root@acl ~]# passwd -u -f oraopepe
```

243. Vamos eliminar el usuario orapepe con su directorio home.

```
[root@acl ~]# userdel -r orapepe
```

244. Si queremos cambiar el shell al usuario orafcld.

```
[root@acl ~]# usermod -s /bin/csh orafcld
```

245. Agregar al usuario dinora a los grupos orasintall y dba.

```
[root@acl ~]# usermod -G oinstall,dba dinora
```

246. Pondremos una fecha de expiración al usuario dinora.

```
[root@acl ~]# usermod -e 2012-03-26 dinora
```

247. Si queremos ver información del usuario dinora.

```
[root@acl ~]# finger dinora
```

248. Si queremos cambiar el shell del usuario dinora.

```
[root@acl ~]# chsh dinora
```

249. La cuenta del usuario orapepe expirará el 28 de marzo del 2012.

```
[root@acl ~]# chage -E 2012-03-28 dinora
```

250. Daremos dos días para que el usuario dinora cambie su password.

```
[root@acl ~]# chage -M  2 dinora
```

251. Si queremos ver información de los cambios de la cuenta dinora.

```
[root@acl ~]# chage -l dinora
```

252. Crea otro usuario con toda las opciones anteriores en una sola linea asignándole otro directorio home llamado mguerrero.

```
[root@acl ~]# useradd -e 2012-02-14 -d /opt -c "usuario de prueba" mguerrero
```

253. Crear un grupo identificado como gacl.

```
[root@acl ~]# groupadd gacl
```

254. Comprueba que se haya realizado con éxito.

```
[root@acl ~]# grep -i gacl /etc/group
```

255. Ponle un password al grupo que acabas de crear.

```
[root@acl ~]# gpasswd gacl
```

256. Cambia el GID del grupo.

```
[root@acl ~]# groupmod -g 655  gacl
```

GNUNota:Verificar el nuevo GID del grupo.

257. Agregar el usuario dinora al grupo gacl.

```
[root@acl ~]# gpasswd -a dinora gacl
```

258. Borra el grupo creado.

```
[root@acl ~]# groupdel  gacl
```

Programando tareas cron y at

259. Vamos iniciar el servicio del cron.

```
[root@acl ~]# service crond start
```

260. Controla archivos de cron para usuarios individuales o para el usuario root.

```
[root@acl ~]# cat /etc/crontab
```

261. Archivos de cron personalizados para programas específicos.

```
[estudiantes@acl ~]$ ls /etc/cron.d
```

262. Scripts de usuarios o de programas específicos que se ejecutan cada día, según lo definido en crontab.

```
[estudiantes@acl ~]$ ls /etc/cron.daily/
```

263. Scripts de usuarios o de programas específicos que se ejecutan cada hora, según lo definido en crontab.

```
[estudiantes@acl ~]$ ls /etc/cron.hourly/
```

264. Scripts de usuarios o de programas específicos que se ejecutan cada mes, según lo definido en crontab.

```
[estudiantes@acl ~]$ ls /etc/cron.monthly/
```

265. Si desea cambiar el editor para editar los crontab.

```
[root@acl ~]# export EDITOR=nano
```

266. Para editar o crear un cron.

```
[root@acl ~]# crontab -e
```

267. Vamos crear nuestro primer cron. Envía la salida del comando date cada 3 minutos a un archivo llamado hora.txt.

```
[root@acl ~]# crontab -e
*/3 * * * * /bin/date  >> /home/brianna/hora.txt
```

```
[root@acl ~]# cat /home/brianna/hora.txt
```

268. Lista las tareas programadas del usuario actual.

```
[root@acl ~]# crontab -l
```

269. Lista las tareas programadas del usuario especificado como root.

```
[root@acl ~]# crontab -u root -l
```

270. Ejecuta la orden who todos los lunes a las 10:30 y guarda la salida en el archivo quien.txt si no está creado él lo crea.

```
[root@acl ~]# crontab -e
30 10 * * 1 /usr/bin/who >> /home/estudiantes/quien.txt
```

271. Crontab para realizar backup de /etc cada 5 minutos.

```
[root@acl ~]# crontab -e
*/5 * * * * /bin/tar -cvzf /root/backetc.tar.gz /etc
```

272. Vamos realizar un scripts para ponerlo correr cada 10 minutos, él mismo hará un backup de /boot.

Creamos el scripts:

```
[root@acl ~]# vim backboot.sh
#!/bin/bash
echo “Inicio del Scripts”
/bin/tar -cvzf ~/backboot.tar.gz /boot
echo “Fin del Scripts”
exit 0
```

273. Ahora lo ponemos a correr en el cron.

```
[root@acl ~]# crontab -e
*/10 * * * * /bin/sh ~/backboot.sh
```

GNUNota: Recuerde probar el scripts antes de ponerlo en el cron.

274. Ahora monitorear los log para ver si el cron se ejecutó.

```
[root@acl ~]# tail -f /var/log/cron
```

GNUNota: Usted debe saber donde están los Logs del sistema.

275. Vamos a ver los archivos cron de cada usuario.

```
[root@acl ~]# cd /var/spool/cron
```

```
[root@acl cron]# pwd
```

```
[root@acl cron]# ls
```

```
[root@acl cron]# cat root
```

GNUNota: Puede ver su contenido con el comando cat.

276. Si queremos eliminar el crontab del usuario actual.

```
[root@acl ~]# crontab -r
```

AT

277. Ejecuta un ls de tu directorio home en 5 minutos con at e imprime la salida a un archivo llamado ls.txt.

```
[root@acl ~]# at now +5 minute <ENTER>
ls -l ~ > ls.txt
CRTL+D
```

GNUNota: Verifique si el archivo ls.txt fue creado.

```
[root@acl ~]# at now +1 days <ENTER>
ls -l ~ > ls.txt
```

278. Verifica que una tarea está en ejecución.

```
[root@acl ~]# atq
```

279. También podemos darle un archivo de comandos a ejecutar a at así.

```
[estudiantes@acl]$ at -f expdp_final.sh -v now
```

280. El mismo ejercicio pero ejecutará el scripts en 3 minuto.

```
[estudiantes@acl]$ at -f expdp_final.sh -v now + 3 minutes
Thu Feb 23 10:03:00 2012
```

```
job 7 at 2012-02-23 10:03
You have new mail in /var/spool/mail/oracle
```

```
[estudiantes@acl]$ atq
7     2012-02-23 10:03 a oracle
```

281. Para eliminar.

```
[estudiantes@acl]$ atrm 7
```

282. Programar que su server se apague en 15 minutos.

```
[estudiantes@acl ~]$ at now + 15 minutes
at> poweroff
at> <EOT> (Esto es igual a CRLT + D)
job 9 at 2012-02-23 10:20
```

GNUNota: La ultima linea informa la hora que el server se apagara.

283. Para ver los trabajos activos.

```
[estudiantes@acl ~]$ at -l
```

Registros del sistema, Syslog

284. Los logs se guardan en archivos ubicados en el directorio /var/log.

```
[root@acl ~]# ls /var/log/
```

285. Aquí encontraremos los logs que llegan con prioridad info (información), notice (notificación) o warn (aviso).

```
[root@acl ~]# cat  /var/log/messages
```

286. Ver las últimas 10 líneas del archivo messages, que son por defecto las más necesarias o más recientes.

```
[root@acl ~]# tail /var/log/messages
```

287. En este log se registran los login en el sistema, las veces que hacemos su, etc. Los intentos fallidos se registran en líneas con información del tipo invalid password o authentication failure.

```
[root@acl ~]# tail /var/log/secure
```

288. En este archivo se almacena la información que genera el kernel durante el arranque del sistema. Podemos ver su contenido con el comando dmesg.

```
[root@acl ~]# tail /var/log/dmesg
```

GNUNota: Utilice la opción -f. Explique al GNU/Instructor la misma.

289. Vamos a ver su contenido ejecutando el comando dmegs.

```
[root@acl ~]# dmesg
```

290. Información del boot.

```
[root@acl ~]# cat /var/log/boot.log
```

291. Ver configuración del archivo del logrotate.

```
[root@acl ~]# cat /etc/logrotate.conf
```

292. Ver archivo de configuración de los servicios a rotar.

```
[root@acl ~]# ls /etc/logrotate.d/
```

293. El archivo lastlog es un fichero binario guardado generalmente en /var/adm/, y que contiene un registro para cada usuario con la fecha y hora de su última conexión; podemos visualizar estos datos para un usuario dado mediante órdenes como who o finger:

```
[root@acl ~]# lastlog
```

294. Ver el listado de los últimos usuarios logueados.

```
[root@acl ~]# last
```

295. Mostrar el historial de reinicio.

```
[root@acl ~]# last reboot
```

296. Si necesitamos saber los últimos dos login en el sistema haríamos.

```
[root@acl ~]# last -n 2
```

297. Ver los últimos login en ese shell para ello.

```
[root@acl ~]# last tty1
```

298. El modificador -i, nos dice desde que dirección de IP se loguearon en nuestro sistema.

```
[root@acl ~]# last -i
```

299. Muestra lo registrado en la lista de usuarios con las entradas de apagado y los cambios en los niveles de ejecución.

```
[root@acl ~]# last -x
```

300. Muestra la fecha y hora del último reinicio del sistema.

```
[root@acl ~]# who -b
```

301. Muestra qué usuarios están trabajando en la máquina en ese momento y qué están haciendo. Utiliza el archivo /var/log/utmp. Este archivo se encuentra en constante cambio.

```
[root@acl ~]# w
```