TP Conceptos de Arquitectura y Sistemas Operativos

1er cuatrimestre 2024

Reeves Lara Agustina

Salguero Valentino Roger

Serantes Elias

Trabajo práctico

Conceptos de Arquitecturas y Sistemas Operativos

1C/2024

1. Introducción

1.1. Objetivos

Al terminar este trabajo Ud. habrá aprendido a:

1. Instalar Linux sobre un sistema de virtualización.
2. Utilizar los principales comandos de Linux.
3. Compilar y ejecutar programas escritos en lenguaje C.
4. Adquirir algunas habilidades y conocimientos de administración
5. Modificar el kernel de Linux utilizando módulos.
6. Resolver algunos problemas de concurrencia

1.2. Normas de entrega

1. La fecha y hora de entrega para este trabajo práctico es la que figura en el cronograma de la materia. Se alienta y acepta la entrega del trabajo, en su totalidad, en forma anticipada.
2. Los trabajos deben ser entregados personalmente a alguno de los docentes de la materia en los horarios de clase o de consulta. No se aceptarán trabajos depositados en otro lugar.
3. No se aceptarán trabajos incompletos.
4. No se aceptarán trabajos que no contengan las pruebas utilizadas por los alumnos para cotejar sus resultados (en los casos requeridos) ni las distintas passwords necesarias para ingresar al sistema y las cuentas**. Esta información deberá estar claramente indicada en el informe.**

1.3. Formato de entrega

Se deberá entregar la imagen de disco utilizada en el sistema de virtualización, con las resoluciones de los ejercicios y sus pruebas incorporadas, así como también los archivos que resuelven las consignas, fuera de la imagen, para poder ser eventualmente revisados por separado.

Se deberá entregar además un documento impreso. Ese documento debe reunir las siguientes características:

1. Formato de presentación: Impreso en hojas de tamaño A4 encarpetadas.

2. Secciones obligatorias del documento:
a) Carátula:

1) Asignatura

2) Número y descripción del trabajo práctico.

3) Año y cuatrimestre de cursada.

4) Identificación del grupo.

5) Nombre, apellido y dirección de correo electrónico de todos los integrantes del grupo.

b) Sección principal: Aquí debe incluirse la resolución de cada uno de los problemas planteados y sus correspondientes pruebas. Para cada respuesta debe indicarse el número y título del problema al que corresponde tal como aparece en el enunciado y los comandos y/o programas utilizados para resolverlo. Se deberá indicar claramente en qué directorio y bajo qué nombre se encuentran los fuentes, los ejecutables y los programas de prueba, en caso de haberlos.

Toda la documentación solicitada, deberá ser entregada en fecha y hora determinada por los docentes. De no cumplirse con este requerimiento, el trabajo será considerado nulo y, en consecuencia, no se aprobará la cursada.

3. Consignas

Antes de empezar, ejecute:

sudo apt-get install man-db manpages manpages-dev
De esta manera tendrá acceso a ayuda en línea ejecutando:
man <comando>
Por ejemplo:
man cp
Puede además instalar la versión en castellano de la ayuda ejecutando:
sudo apt-get install manpages-es
Para acceder a la ayuda en castellano ejecute por ejemplo:
man -L es cp
Tenga presente que no todos los comandos poseen ayuda en castellano.

sudo permite a usuarios normales ejecutar comandos que requieren permisos de administrador. Al ejecutar un comando con sudo el sistema le pedirá su password, y no el password del administrador (llamado root en Linux, siguiendo la tradición de Unix). Esto sucede ya que el sistema permite que ciertos usuarios (que deberán corresponderse con usuarios “privilegiados" del sistema) puedan utilizar sudo ingresando solamente su propio password. El usuario por defecto creado en una instalación de Ubuntu tiene este permiso y por lo tanto en Ubuntu no es necesario una cuenta de administrador o root.

apt-get es el manejador de paquetes de la distribución Ubuntu. Permite instalar, actualizar y desinstalar programas. Más adelante lo utilizaremos para instalar las herramientas necesarias para compilar programas en Linux.

Si se encontrara detrás de un proxy, antes de utilizar apt-get debe configurar el proxy. Ejecute el siguiente comando:

sudo echo "Acquire::http::Proxy \"http://miproxy.midominio.ar:8080\";" >/etc/apt/apt.conf

3.1. Comandos básicos de Unix

3.1.1. *pwd* Indique qué directorio pasa a ser su directorio actual si ejecuta:

a)cd /usr/bin

```
tpcaso@ubuntu:~$ cd /usr/bin
tpcaso@ubuntu:/usr/bin$ pwd
/usr/bin
tpcaso@ubuntu:/usr/bin$
```

b)cd

```
tpcaso@ubuntu:/usr/bin$ cd
tpcaso@ubuntu:~$ pwd
/home/tpcaso
tpcaso@ubuntu:~$
```

c) ¿Cómo explica el punto anterior?
Cuando se ingresa cd /usr/bin se ingresa a la carpeta indicada y al ingresar cd solo se regresa al home del usuario.

3.1.2.*cat* ¿Cuál es el contenido del archivo */home/<usuario>/.profile*?

```
tpcaso@ubuntu:~$ cat /home/tpcaso/.profile
# ~/.profile: executed by the command interpreter for login shells.
# This file is not read by bash(1), if ~/.bash_profile or ~/.bash_login
# exists.
# see /usr/share/doc/bash/examples/startup-files for examples.
# the files are located in the bash-doc package.

# the default umask is set in /etc/profile; for setting the umask
# for ssh logins, install and configure the libpam-umask package.
#umask 022

# if running bash
if [ -n "$BASH_VERSION" ]; then
    # include .bashrc if it exists
    if [ -f "$HOME/.bashrc" ]; then
        . "$HOME/.bashrc"
    fi
fi

# set PATH so it includes user's private bin directories
PATH="$HOME/bin:$HOME/.local/bin:$PATH"
tpcaso@ubuntu:~$
```

3.1.3.*find* Liste todos los archivos que comienzan con vmlinuz.

```
tpcaso@ubuntu:~$ sudo find / -name 'vmlinux*'
/usr/src/linux-headers-4.4.0-142/arch/m68k/kernel/vmlinux-nommu.lds
/usr/src/linux-headers-4.4.0-142/arch/m68k/kernel/vmlinux-sun3.lds
/usr/src/linux-headers-4.4.0-142/arch/m68k/kernel/vmlinux-std.lds
/usr/src/linux-headers-4.4.0-142/arch/mn10300/boot/compressed/vmlinux.lds
/usr/src/linux-headers-4.4.0-142/arch/h8300/boot/compressed/vmlinux.lds
/usr/src/linux-headers-4.4.0-142/arch/sh/include/asm/vmlinux.lds.h
/usr/src/linux-headers-4.4.0-142/include/asm-generic/vmlinux.lds.h
/usr/src/linux-headers-4.4.0-142/scripts/gdb/vmlinux-gdb.py
tpcaso@ubuntu:~$ _
```

3.1.4.*mkdir* Genere un directorio */home/<usuario>/tp*.

```
tpcaso@ubuntu:~$ mkdir /home/tpcaso/tp
tpcaso@ubuntu:~$ pwd
/home/tpcaso
tpcaso@ubuntu:~$ ls
tp
tpcaso@ubuntu:~$ _
```

3.1.5.*cp* Copie el archivo */etc/passwd* al directorio */home/<usuario>/tp*.

```
tpcaso@ubuntu:~$ cp /etc/passwd /home/tpcaso/tp
tpcaso@ubuntu:~$ ls
tp
tpcaso@ubuntu:~$ cd tp
tpcaso@ubuntu:~/tp$ ls
passwd
tpcaso@ubuntu:~/tp$ _
```

3.1.6.*chgrp* Cambie el grupo del archivo */home/<usuario>/tp/passwd* para que sea el suyo.

```
tpcaso@ubuntu:~$ chgrp tpcaso /home/tpcaso/tp/passwd
tpcaso@ubuntu:~$ ls -l tp/passwd
-rw-r--r-- 1 tpcaso tpcaso 1567 jun 13 13:26 tp/passwd
tpcaso@ubuntu:~$
```

3.1.7.*chown* Cambie el dueño del archivo */home/<usuario>/tp/passwd* para que sea su usuario.

```
tpcaso@ubuntu:~$ chown tpcaso /home/tpcaso/tp/passwd
tpcaso@ubuntu:~$ _
```

3.1.8.*chmod* Cambie los permisos del archivo /home/<usuario>/tp/passwd para que: el propietario tenga permisos de lectura, escritura y ejecución, el grupo tenga sólo permisos de lectura y ejecución, el resto tenga sólo permisos de ejecución.

```
tpcaso@ubuntu:~$ chmod u=rwx,g=rx,o=x /home/tpcaso/tp/passwd
tpcaso@ubuntu:~$ ls -l
total 4
drwxrwxr-x 2 tpcaso tpcaso 4096 jun 13 13:26 tp
tpcaso@ubuntu:~$ cd tp
tpcaso@ubuntu:~/tp$ ls -l
total 4
-rwxr-x--x 1 tpcaso tpcaso 1567 jun 13 13:26 passwd
tpcaso@ubuntu:~/tp$ _
```

3.1.9.*grep*

Muestre las líneas que tienen el texto “localhost" en el archivo */etc/hosts*.

```
tpcaso@ubuntu:~$ grep -n 'localhost' /etc/hosts
1:127.0.0.1     localhost
5:::1     localhost ip6-localhost ip6-loopback
tpcaso@ubuntu:~$ _
```

Muestre todas las líneas que tengan el texto “POSIX" de todos los archivos (incluyendo subdirectorios) en */etc*. Evite los archivos binarios y aquellos archivos y directorios que no tienen permiso de lectura para su usuario.

```
tpcaso@ubuntu:~$ sudo grep -rl 'POSIX' /etc
[sudo] password for tpcaso:
/etc/lvm/lvm.conf
/etc/security/limits.conf
tpcaso@ubuntu:~$ _
```

3.1.10.*passwd* Cambie su password. (anote la nueva password en el informe)

Antigua contraseña: 1234
Nueva contraseña: unsam2024

```
tpcaso@ubuntu:~$ passwd
Cambiando la contraseña de tpcaso.
(actual) contraseña de UNIX:
Introduzca la nueva contraseña de UNIX:
Vuelva a escribir la nueva contraseña de UNIX:
passwd: password updated successfully
tpcaso@ubuntu:~$
```

3.1.11. *rm* Borre el archivo /home/<usuario>/tp/passwd

```
tpcaso@ubuntu:~$ rm /home/tpcaso/tp/passwd
tpcaso@ubuntu:~$ cd tp
tpcaso@ubuntu:~/tp$ ls -l
total 0
tpcaso@ubuntu:~/tp$ cd
tpcaso@ubuntu:~$ ls -l
total 4
drwxrwxr-x 2 tpcaso tpcaso 4096 jun 13 13:41 tp
tpcaso@ubuntu:~$ _
```

3.1.12.*ln*

Enlazar el archivo */etc/passwd* a los archivos */tmp/contra1* y */tmp/contra2*.

```
tpcaso@ubuntu:~$ sudo ln /etc/passwd /tmp/contra1
[sudo] password for tpcaso:
tpcaso@ubuntu:~$ sudo ln /etc/passwd /tmp/contra2
tpcaso@ubuntu:~$
```

Hacer un *ls -l* para ver cuantos enlaces tiene */etc/passwd*.

```
tpcaso@ubuntu:~$ ls -l /etc/passwd
-rw-r--r-- 3 root root 1567 jun 13 13:07 /etc/passwd
tpcaso@ubuntu:~$ _
```

Estos enlaces se llaman “hardlinks”. Cada nuevo enlace referencia el mismo espacio ocupado del disco rígido, y por lo tanto cada hardlink es igual de representativo de esos bytes ocupados del disco rígido. El espacio ocupado solamente se liberará cuando todos los enlaces hayan sido borrados.

Ahora enlace el archivo */etc/passwd* de manera “soft” archivo contra3.

```
tpcaso@ubuntu:~$ sudo ln -s /etc/passwd contra3
tpcaso@ubuntu:~$
```

Verifique con *ls -l* que no aumentó la cantidad de enlaces de */etc/passwd*.

```
tpcaso@ubuntu:~$ ls -l /etc/passwd
-rw-r--r-- 3 root root 1567 jun 13 13:07 /etc/passwd
tpcaso@ubuntu:~$ _
```

Estos enlaces se llaman “softlinks” y apuntan no a los bytes del disco rígido sino a la ruta del archivo a ser enlazado. Operar sobre el softlink es igual que operar sobre el archivo, sin embargo los softlinks no cuentan en la cantidad de enlaces (ya que no apuntan a los bytes ocupados del disco rígido) y pueden ser borrados sin afectar al archivo original, aunque si se borra el archivo original el softlink quedar a huérfano y no apuntará a nada.

3.1.13.*mount*

Monte el CD-ROM de instalación de Ubuntu y liste su contenido.

Para hacer esto deberá especificar la ISO de instalación de Ubuntu como CD-ROM de la máquina virtual. Si bien puede hacer esto como lo hizo para instalar el sistema, si la máquina virtual está corriendo debe hacer click derecho en el ícono con forma de CD-ROM en la esquina inferior derecha de la máquina virtual, y seleccionar CD/DVD-ROM Image... (ver Figura anterior). En la ventana que aparece seleccione la ISO de instalación

Presente los filesystems que tiene montados.

```
tpcaso@ubuntu:~$ mount
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
udev on /dev type devtmpfs (rw,nosuid,relatime,size=1011512k,nr_inodes=209835,mode=755)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000)
tmpfs on /run type tmpfs (rw,nosuid,noexec,relatime,size=206216k,mode=755)
/dev/sda1 on / type ext4 (rw,relatime,errors=remount-ro,data=ordered)
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev)
tmpfs on /run/lock type tmpfs (rw,nosuid,nodev,noexec,relatime,size=5120k)
tmpfs on /sys/fs/cgroup type tmpfs (ro,nosuid,nodev,noexec,mode=755)
cgroup on /sys/fs/cgroup/systemd type cgroup (rw,nosuid,nodev,noexec,relatime,xattr,release_agent=/l
ib/systemd/systemd-cgroups-agent,name=systemd)
pstore on /sys/fs/pstore type pstore (rw,nosuid,nodev,noexec,relatime)
cgroup on /sys/fs/cgroup/cpu,cpuacct type cgroup (rw,nosuid,nodev,noexec,relatime,cpu,cpuacct)
cgroup on /sys/fs/cgroup/net_cls,net_prio type cgroup (rw,nosuid,nodev,noexec,relatime,net_cls,net_p
rio)
cgroup on /sys/fs/cgroup/freezer type cgroup (rw,nosuid,nodev,noexec,relatime,freezer)
cgroup on /sys/fs/cgroup/memory type cgroup (rw,nosuid,nodev,noexec,relatime,memory)
cgroup on /sys/fs/cgroup/hugetlb type cgroup (rw,nosuid,nodev,noexec,relatime,hugetlb)
cgroup on /sys/fs/cgroup/pids type cgroup (rw,nosuid,nodev,noexec,relatime,pids)
cgroup on /sys/fs/cgroup/cpuset type cgroup (rw,nosuid,nodev,noexec,relatime,cpuset)
cgroup on /sys/fs/cgroup/blkio type cgroup (rw,nosuid,nodev,noexec,relatime,blkio)
cgroup on /sys/fs/cgroup/devices type cgroup (rw,nosuid,nodev,noexec,relatime,devices)
cgroup on /sys/fs/cgroup/perf_event type cgroup (rw,nosuid,nodev,noexec,relatime,perf_event)
hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime)
systemd-1 on /proc/sys/fs/binfmt_misc type autofs (rw,relatime,fd=31,pgrp=1,timeout=0,minproto=5,max
proto=5,direct)
mqueue on /dev/mqueue type mqueue (rw,relatime)
debugfs on /sys/kernel/debug type debugfs (rw,relatime)
fusectl on /sys/fs/fuse/connections type fusectl (rw,relatime)
lxcfs on /var/lib/lxcfs type fuse.lxcfs (rw,nosuid,nodev,relatime,user_id=0,group_id=0,allow_other)
tmpfs on /run/user/1000 type tmpfs (rw,nosuid,nodev,relatime,size=206216k,mode=700,uid=1000,gid=1000
)
binfmt_misc on /proc/sys/fs/binfmt_misc type binfmt_misc (rw,relatime)
tracefs on /sys/kernel/debug/tracing type tracefs (rw,relatime)
tpcaso@ubuntu:~$ _
```

3.1.14.*df* ¿Qué espacio libre tiene cada uno de los filesystems montados?

```
tpcaso@ubuntu:~$ df
S.ficheros     bloques de 1K  Usados Disponibles Uso% Montado en
udev                1011512       0     1011512   0% /dev
tmpfs                206216    3260      202956   2% /run
/dev/sda1           3077148 1401612     1499512  49% /
tmpfs               1031072       0     1031072   0% /dev/shm
tmpfs                  5120       0        5120   0% /run/lock
tmpfs               1031072       0     1031072   0% /sys/fs/cgroup
tmpfs                206216       0      206216   0% /run/user/1000
tpcaso@ubuntu:~$ _
```

3.1.15.*ps* ¿Cuántos procesos de usuario tiene ejecutando? Indique cuántos son del sistema.

```
tpcaso@ubuntu:~$ ps -u tpcaso
  PID TTY          TIME CMD
 1100 ?        00:00:00 systemd
 1102 ?        00:00:00 (sd-pam)
 1108 tty1     00:00:00 bash
 9896 tty1     00:00:00 ps
tpcaso@ubuntu:~$ _
```

3.1.16.*umount* Desmonte el CD-ROM de instalación de Ubuntu.

```
tpcaso@ubuntu:~$ umount

Usage:
 umount [-hV]
 umount -a [options]
 umount [options] <source> | <directory>

Unmount filesystems.

Opciones:
 -a, --all               unmount all filesystems
 -A, --all-targets       unmount all mountpoints for the given device in the
                           current namespace
 -c, --no-canonicalize   don't canonicalize paths
 -d, --detach-loop       if mounted loop device, also free this loop device
     --fake              dry run; skip the umount(2) syscall
 -f, --force             force unmount (in case of an unreachable NFS system)
 -i, --internal-only     don't call the umount.<type> helpers
 -n, --no-mtab           don't write to /etc/mtab
 -l, --lazy              detach the filesystem now, clean up things later
 -O, --test-opts <list>  limit the set of filesystems (use with -a)
 -R, --recursive         recursively unmount a target with all its children
 -r, --read-only         in case unmounting fails, try to remount read-only
 -t, --types <list>      limit the set of filesystem types
 -v, --verbose           say what is being done

 -h, --help     display this help and exit
 -V, --version  output version information and exit

For more details see umount(8).
tpcaso@ubuntu:~$
```

3.1.17.*uptime* ¿Cuanto tiempo lleva ejecutando su máquina virtual?

```
tpcaso@ubuntu:~$ uptime
 01:30:29 up  1:02,  1 user,  load average: 0,00, 0,00, 0,00
tpcaso@ubuntu:~$
```

3.1.18.*uname* ¿Qué versión del kernel de Linux está utilizando?

```
tpcaso@ubuntu:~$ uname -a
Linux ubuntu 4.4.0-142-generic #168-Ubuntu SMP Wed Jan 16 21:01:15 UTC 2019 i686 athlon i686 GNU/Lin
ux
tpcaso@ubuntu:~$ _
```

3.1.19 Sistemas operativos y el hardware

3.1.19.1 Muestre en pantalla todos los dispositivos PCI presentes en el sistema.

```
tpcaso@ubuntu:~$ lspci
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371AB/EB/MB PIIX4 IDE (rev 01)
00:02.0 VGA compatible controller: VMware SVGA II Adapter
00:03.0 Ethernet controller: Intel Corporation 82540EM Gigabit Ethernet Controller (rev 02)
00:04.0 System peripheral: InnoTek Systemberatung GmbH VirtualBox Guest Service
00:05.0 Multimedia audio controller: Intel Corporation 82801AA AC'97 Audio Controller (rev 01)
00:06.0 USB controller: Apple Inc. KeyLargo/Intrepid USB
00:07.0 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 08)
00:0b.0 USB controller: Intel Corporation 82801FB/FBM/FR/FW/FRW (ICH6 Family) USB2 EHCI Controller
00:0d.0 SATA controller: Intel Corporation 82801HM/HEM (ICH8M/ICH8M-E) SATA Controller [AHCI mode] (
rev 02)
tpcaso@ubuntu:~$
```

3.1.19.2 Muestre la mayor cantidad de información distinta de los mismos.

```
tpcaso@ubuntu:~$ lspci -v
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
        Flags: fast devsel

00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
        Flags: bus master, medium devsel, latency 0

00:01.1 IDE interface: Intel Corporation 82371AB/EB/MB PIIX4 IDE (rev 01) (prog-if 8a [ISA Compatibi
lity mode controller, supports both channels switched to PCI native mode, supports bus mastering])
        Flags: bus master, fast devsel, latency 64
        [virtual] Memory at 000001f0 (32-bit, non-prefetchable) [size=8]
        [virtual] Memory at 000003f0 (type 3, non-prefetchable)
        [virtual] Memory at 00000170 (32-bit, non-prefetchable) [size=8]
        [virtual] Memory at 00000370 (type 3, non-prefetchable)
        I/O ports at d000 [size=16]
        Kernel driver in use: ata_piix
        Kernel modules: pata_acpi

00:02.0 VGA compatible controller: VMware SVGA II Adapter (prog-if 00 [VGA controller])
        Subsystem: VMware SVGA II Adapter
        Flags: bus master, fast devsel, latency 64, IRQ 18
        I/O ports at d010 [size=16]
        Memory at e0000000 (32-bit, prefetchable) [size=16M]
        Memory at f0000000 (32-bit, non-prefetchable) [size=2M]
        Expansion ROM at <unassigned> [disabled]
        Kernel driver in use: vmwgfx
        Kernel modules: vmwgfx
```

```
00:03.0 Ethernet controller: Intel Corporation 82540EM Gigabit Ethernet Controller (rev 02)
        Subsystem: Intel Corporation PRO/1000 MT Desktop Adapter
        Flags: bus master, 66MHz, medium devsel, latency 64, IRQ 19
        Memory at f0200000 (32-bit, non-prefetchable) [size=128K]
        I/O ports at d020 [size=8]
        Capabilities: <access denied>
        Kernel driver in use: e1000
        Kernel modules: e1000

00:04.0 System peripheral: InnoTek Systemberatung GmbH VirtualBox Guest Service
        Flags: fast devsel, IRQ 20
        I/O ports at d040 [size=32]
        Memory at f0400000 (32-bit, non-prefetchable) [size=4M]
        Memory at f0800000 (32-bit, prefetchable) [size=16K]
        Kernel driver in use: vboxguest
        Kernel modules: vboxguest
```

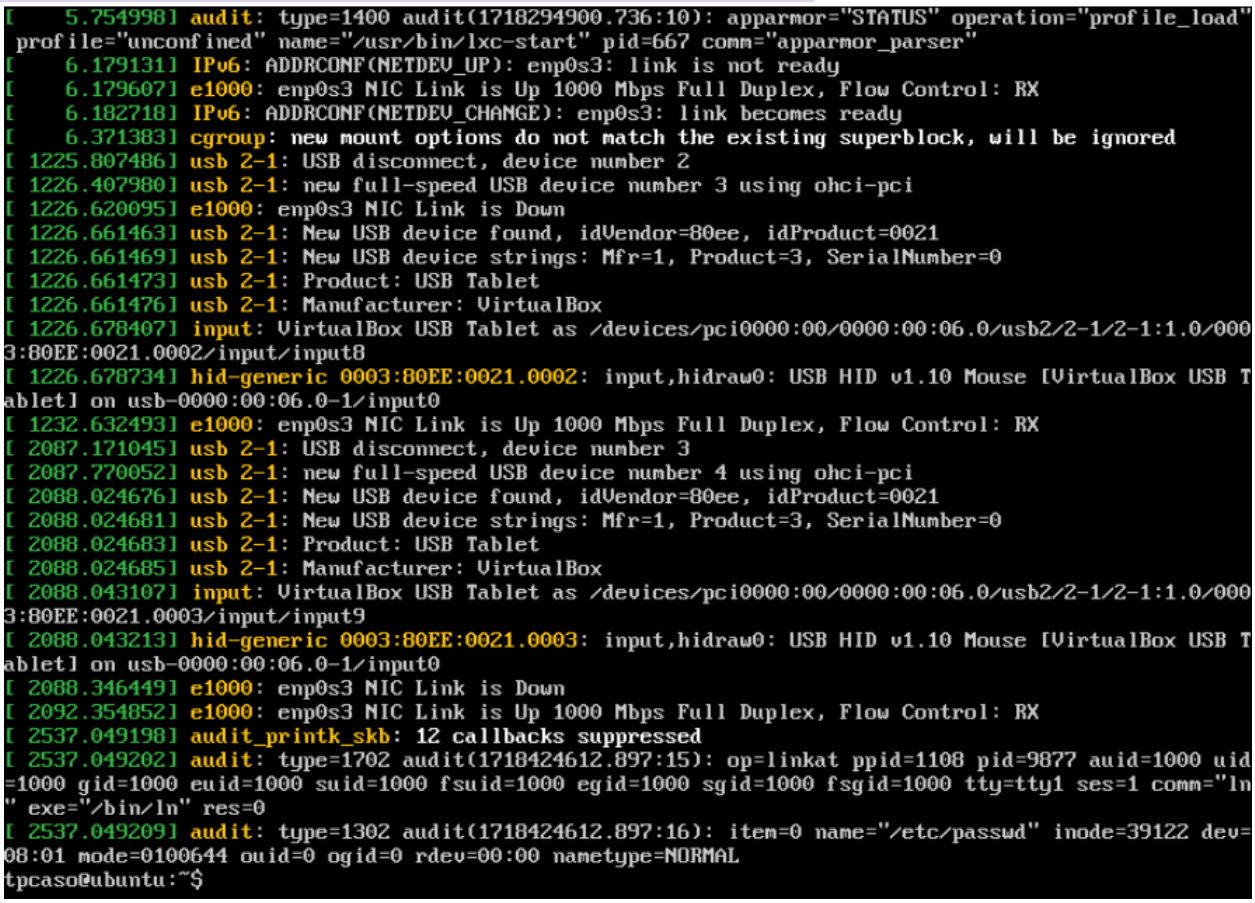
```
00:05.0 Multimedia audio controller: Intel Corporation 82801AA AC'97 Audio Controller (rev 01)
        Subsystem: Dell 82801AA AC'97 Audio Controller
        Flags: bus master, medium devsel, latency 64, IRQ 21
        I/O ports at d100 [size=256]
        I/O ports at d200 [size=64]
        Kernel driver in use: snd_intel8x0
        Kernel modules: snd_intel8x0

00:06.0 USB controller: Apple Inc. KeyLargo/Intrepid USB (prog-if 10 [OHCI])
        Flags: bus master, fast devsel, latency 64, IRQ 22
        Memory at f0804000 (32-bit, non-prefetchable) [size=4K]
        Kernel driver in use: ohci-pci

00:07.0 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 08)
        Flags: medium devsel, IRQ 9
        Kernel driver in use: piix4_smbus
        Kernel modules: i2c_piix4
```

```
00:0b.0 USB controller: Intel Corporation 82801FB/FBM/FR/FW/FRW (ICH6 Family) USB2 EHCI Controller (
prog-if 20 [EHCI])
        Flags: bus master, fast devsel, latency 64, IRQ 19
        Memory at f0805000 (32-bit, non-prefetchable) [size=4K]
        Kernel driver in use: ehci-pci

00:0d.0 SATA controller: Intel Corporation 82801HM/HEM (ICH8M/ICH8M-E) SATA Controller [AHCI mode] (
rev 02) (prog-if 01 [AHCI 1.0])
        Flags: bus master, fast devsel, latency 64, IRQ 21
        I/O ports at d240 [size=8]
        I/O ports at d248 [size=4]
        I/O ports at d250 [size=8]
        I/O ports at d258 [size=4]
        I/O ports at d260 [size=16]
        Memory at f0806000 (32-bit, non-prefetchable) [size=8K]
        Capabilities: <access denied>
        Kernel driver in use: ahci
        Kernel modules: ahci
```

3.1.19.3 De la misma manera que el anterior muestre el subsistema USB y los dispositivos conectados.

```
tpcaso@ubuntu:~$ lsusb
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 004: ID 80ee:0021 VirtualBox USB Tablet
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
tpcaso@ubuntu:~$
```

3.1.19.4 Idem anterior muestre el contenido del registro del núcleo, lo que implica que al comienzo se encuentra toda la información relativa a la detección del hardware.



- Debido a que aparecía gran cantidad de información decidimos únicamente colocar las últimas líneas de la misma.

3.1.19.5 Busque una forma sencilla de mostrar información que le indique

a) Información de los procesadores

```
tpcaso@ubuntu:~$ cat /proc/cpuinfo
processor       : 0
vendor_id       : AuthenticAMD
cpu family      : 23
model           : 24
model name      : AMD Ryzen 5 3400G with Radeon Vega Graphics
stepping        : 1
cpu MHz         : 3693.054
cache size      : 512 KB
physical id     : 0
siblings        : 1
core id         : 0
cpu cores       : 1
apicid          : 0
initial apicid  : 0
fdiv_bug        : no
f00f_bug        : no
coma_bug        : no
fpu             : yes
fpu_exception   : yes
cpuid level     : 13
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mm
x fxsr sse sse2 ht syscall nx mmxext fxsr_opt rdtscp lm constant_tsc rep_good nonstop_tsc extd_apici
d pni pclmulqdq monitor ssse3 cx16 sse4_1 sse4_2 x2apic movbe popcnt aes xsave avx rdrand hypervisor
 lahf_lm cr8_legacy abm sse4a misalignsse 3dnowprefetch ssbd vmmcall fsgsbase bmi1 avx2 bmi2 rdseed
clflushopt arat
bugs            : fxsave_leak sysret_ss_attrs spectre_v1 spectre_v2 spec_store_bypass
bogomips        : 7386.10
clflush size    : 64
cache_alignment : 64
address sizes   : 48 bits physical, 48 bits virtual
power management:

tpcaso@ubuntu:~$ _
```

b) Información de la memoria

El comando utilizado es **cat /proc/meminfo**

```
Unevictable:       3428 kB
Mlocked:           3428 kB
HighTotal:      1183688 kB
HighFree:       1032284 kB
LowTotal:        878460 kB
LowFree:         676244 kB
SwapTotal:       998396 kB
SwapFree:        998396 kB
Dirty:                0 kB
Writeback:            0 kB
AnonPages:        16360 kB
Mapped:           29812 kB
Shmem:             3284 kB
Slab:             44836 kB
SReclaimable:     34100 kB
SUnreclaim:       10736 kB
KernelStack:        928 kB
PageTables:         760 kB
NFS_Unstable:         0 kB
Bounce:               0 kB
WritebackTmp:         0 kB
CommitLimit:    2029468 kB
Committed_AS:    263300 kB
VmallocTotal:    122880 kB
VmallocUsed:          0 kB
VmallocChunk:         0 kB
AnonHugePages:        0 kB
CmaTotal:             0 kB
CmaFree:              0 kB
HugePages_Total:      0
HugePages_Free:       0
HugePages_Rsvd:       0
HugePages_Surp:       0
Hugepagesize:      2048 kB
DirectMap4k:      28664 kB
DirectMap2M:     884736 kB
tpcaso@ubuntu:~$ _
```

c) Información de los números más bajos y más altos de los dispositivos de almacenamiento.

```
tpcaso@ubuntu:~$ sudo fdisk -l
[sudo] password for tpcaso:
Disk /dev/sda: 4 GiB, 4294967296 bytes, 8388608 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xe5b9ecc9

Disposit.  Inicio   Start   Final Sectores  Size Id Tipo
/dev/sda1  *         2048 6387711  6385664    3G 83 Linux
/dev/sda2         6389758 8386559  1996802  975M  5 Extendida
/dev/sda5         6389760 8386559  1996800  975M 82 Linux swap / Solaris
tpcaso@ubuntu:~$ _
```

3.2. Salida estándar y pipes

3.2.1. STDOUT

a) Conserve en el archivo *~/home/<usuario>/tp/config* la salida del comando ls que muestra todos los archivos del directorio */etc* y de los subdirectorios bajo */etc*.

```
tpcaso@ubuntu:~$ sudo ls -R /etc > /home/tpcaso/tp/config
tpcaso@ubuntu:~$
```

Luego para corroborar que todo se haya hecho correctamente se introduce cat /home/tpcaso/tp/config.

b) Presente cuántas líneas, palabras y caracteres tiene */home/<usuario>/tp/config*.

```
tpcaso@ubuntu:~$ wc /home/tpcaso/tp/config
 2039  1847 27903 /home/tpcaso/tp/config
tpcaso@ubuntu:~$ _
```

wc = word count

c) Agregue el contenido, ordenado alfabéticamente, del archivo */etc/passwd* al final del archivo */home/<usuario>/tp/config*.

```
tpcaso@ubuntu:~$ sort /etc/passwd >> /home/tpcaso/tp/config
tpcaso@ubuntu:~$ tail /home/tpcaso/tp/config
syslog:x:104:108::/home/syslog:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
sys:x:3:3:sys:/dev:/usr/sbin/nologin
tpcaso:x:1000:1000:Tpcaso,,,:/home/tpcaso:/bin/bash
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
uuidd:x:108:112::/run/uuidd:/bin/false
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
tpcaso@ubuntu:~$ _
```

d) Presente cuantas líneas, palabras y caracteres tiene */home/<usuario>/tp/config*.

```
tpcaso@ubuntu:~$ wc /home/tpcaso/tp/config
 2069  1889 29470 /home/tpcaso/tp/config
tpcaso@ubuntu:~$ _
```

3.2.2. Pipes

a) Liste en forma amplia los archivos del directorio /usr/bin que comiencen con la letra "s". Del resultado obtenido, seleccione las líneas que contienen el texto "sys" e informe la cantidad de caracteres, palabras y líneas. Está prohibido, en este ítem, usar archivos temporales de trabajo.

```
tpcaso@ubuntu:~$ cd /usr/bin
tpcaso@ubuntu:/usr/bin$ ls s* | wc sys*
   6830   44755 1448548 systemd-analyze
     99     724   30388 systemd-cat
   1224    8455  305244 systemd-cgls
   1299    8706  317580 systemd-cgtop
    252    1654   67336 systemd-delta
     96     728   30380 systemd-detect-virt
    125     891   38572 systemd-path
   1315    9137  338012 systemd-resolve
   1510   10706  387356 systemd-run
   1552   10466  366676 systemd-stdio-bridge
  14302   96222 3330092 total
tpcaso@ubuntu:/usr/bin$
```

3.3. Scripting

3.3.1

Escriba un script de shell que cada 5 minutos diga "HOLA" y que todos los días a las 19 hrs también lo haga. (Pista: busque comandos de control de tiempo para disparar las acciones).

```
"hola.sh" [Nuevo] 3L, 36C escritos
tpcaso@ubuntu:~/tp$ chmod 777 hola.sh
tpcaso@ubuntu:~/tp$ _
```

Accedemos al crontab con el comando "crontab -e" y colocamos las siguientes líneas al final.

```
*/5 * * * * /home/tpcaso/tp/hola.sh
0 19 * * * /home/tpcaso/tp/hola.sh
```

Para comprobar que funciona decidimos poner la hora en la que nos encontrábamos trabajando.

```
*/5 * * * * /home/tpcaso/tp/hola.sh
25 20 * * * /home/tpcaso/tp/hola.sh




crontab: installing new crontab

Broadcast message from tpcaso@ubuntu (somewhere) (Fri Jun 28 20:25:01 2024):

Hola


Broadcast message from tpcaso@ubuntu (somewhere) (Fri Jun 28 20:25:01 2024):

Hola


Broadcast message from tpcaso@ubuntu (somewhere) (Fri Jun 28 20:30:01 2024):

Hola
```

3.3.2

Escriba un script de shell que tome parámetros desde línea de comandos y genere un archivo de salida
en el home del usuario que ejecuta el script con el nombre salida.txt.
El contenido de salida.txt dependerá del parámetro indicado en la línea de comando:

SCRIPT -u <usuario> : Mostrará la información separada por tabuladores del archivo /etc/passwd correspondiente al usuario indicado o en caso de no existir el usuario "Usuario no encontrado". En caso de no especificar usuario alguno se mostrará la información de todos los usuarios.

SCRIPT -g <grupo> : Mostrará la información de todos los usuarios, separada por tabuladores que pertenezcan al grupo indicado. En caso de no existir el grupo en /etc/group deberá mostrar la leyenda "Grupo no existente". En caso de existir el grupo pero no tener usuarios asignados, se deberá mostrar la leyendo "Grupo sin usuarios".

```
#!/bin/bash
inicio() {
        echo "Debe ingresar -u para usuario o -g para grupo"
                echo "Uso incorrecto de argumento para -u"
        else
                fi
                echo "No se ha encontrado el grupo" > /home/tpcaso/tp/salidaGrupo.txt
#!/bin/bash
if [ $# = "0" ]; then
        echo "Debe ingresar -u para usuario o -g para grupo"
fi
if [ "$1" = "-u" ]; then
        if [ $# = 1 ]; then
                cat /etc/passwd | tr ":" \\t > /home/tpcaso/tp/salida.txt
        fi
        if [ $# = 2 ]; then
                if [ grep -q "$2" "etc/passwd" ]; then
                        grep "$2" "/etc/passwd" | tr \\t > /home/tpcaso/tp/salida.txt
                else
                        echo "No se ha encontrado el usuario" > /home/tpcaso/tp/salida.txt
                fi
        fi
fi
```

```
if [ "$1" = "-g" ]; then
        if [ $# = 1 ]; then
                cat /etc/group | tr ":" \\t > /home/tpcaso/tp/salidaGrupo.txt
        fi
        if [ $# = 2 ]; then
                if [ grep -q "2" "/etc/group" ]; then
                        grep "2" "/etc/group" | tr \\t > /home/tpcaso/tp/salidaGrupo.txt
                else
                        echo "Grupo sin usuarios" > /home/tpcaso/tp/salidaGrupo.txt
                fi
        else
                echo "No se ha encontrado el grupo" > /home/tpcaso/tp/salidaGrupo.txt
        fi
fi
```

```
Debe ingresar -u para usuario o -g para grupo
tpcaso@ubuntu:~/tp$ ./script.sh -u
tpcaso@ubuntu:~/tp$ less salida.txt
root    x       0       0       root    /root   /bin/bash
daemon  x       1       1       daemon  /usr/sbin       /usr/sbin/nologin
bin     x       2       2       bin     /bin    /usr/sbin/nologin
sys     x       3       3       sys     /dev    /usr/sbin/nologin
sync    x       4       65534   sync    /bin    /bin/sync
games   x       5       60      games   /usr/games      /usr/sbin/nologin
man     x       6       12      man     /var/cache/man  /usr/sbin/nologin
lp      x       7       7       lp      /var/spool/lpd  /usr/sbin/nologin
mail    x       8       8       mail    /var/mail       /usr/sbin/nologin
news    x       9       9       news    /var/spool/news /usr/sbin/nologin
uucp    x       10      10      uucp    /var/spool/uucp /usr/sbin/nologin
proxy   x       13      13      proxy   /bin    /usr/sbin/nologin
www-data        x       33      33      www-data        /var/www        /usr/sbin/nologin
backup  x       34      34      backup  /var/backups    /usr/sbin/nologin
list    x       38      38      Mailing List Manager    /var/list       /usr/sbin/nologin
irc     x       39      39      ircd    /var/run/ircd   /usr/sbin/nologin
gnats   x       41      41      Gnats Bug-Reporting System (admin)      /var/lib/gnats  /usr/sbin/no
login
nobody  x       65534   65534   nobody  /nonexistent    /usr/sbin/nologin
systemd-timesync        x       100     102     systemd Time Synchronization,,, /run/systemd    /bin
/false
systemd-network x       101     103     systemd Network Management,,,   /run/systemd/netif      /bin
/false
systemd-resolve x       102     104     systemd Resolver,,,     /run/systemd/resolve    /bin/false
systemd-bus-proxy       x       103     105     systemd Bus Proxy,,,    /run/systemd    /bin/false
syslog  x       104     108             /home/syslog    /bin/false
_apt    x       105     65534           /nonexistent    /bin/false
lxd     x       106     65534           /var/lib/lxd/   /bin/false
messagebus      x       107     111             /var/run/dbus   /bin/false
uuidd   x       108     112             /run/uuidd      /bin/false
dnsmasq x       109     65534   dnsmasq,,,      /var/lib/misc   /bin/false
sshd    x       110     65534           /var/run/sshd   /usr/sbin/nologin
tpcaso  x       1000    1000    Tpcaso,,,       /home/tpcaso    /bin/bash
```

```
tpcaso@ubuntu:~/tp$ ./script.sh
Debe ingresar -u para usuario o -g para grupo
tpcaso@ubuntu:~/tp$ ./script.sh -g
tpcaso@ubuntu:~/tp$ less salidaGrupo.txt
No se ha encontrado el grupo
```

3.3.2

Haga que un script que escriba en la pantalla "Hola Mundo" cada vez que se loguee un usuario.

```
#!/bin/bash
echo "Hola mundo"
wall "Hola mundo"
```

```
# ~/.profile: executed by the command interpreter for login shells.
# This file is not read by bash(1), if ~/.bash_profile or ~/.bash_login
# exists.
# see /usr/share/doc/bash/examples/startup-files for examples.
# the files are located in the bash-doc package.

# the default umask is set in /etc/profile; for setting the umask
# for ssh logins, install and configure the libpam-umask package.
#umask 022

# if running bash
if [ -n "$BASH_VERSION" ]; then
    # include .bashrc if it exists
    if [ -f "$HOME/.bashrc" ]; then
        . "$HOME/.bashrc"
    fi
fi

# set PATH so it includes user's private bin directories
PATH="$HOME/bin:$HOME/.local/bin:$PATH"
/home/tpcaso/tp/saludo.sh
```

```
Ubuntu 16.04.6 LTS ubuntu tty1

ubuntu login: tpcaso
Password:
Last login: Fri Jun 21 16:28:44 -03 2024 on tty1
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-142-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Pueden actualizarse 196 paquetes.
137 actualizaciones son de seguridad.

New release '18.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Hola mundo

Broadcast message from tpcaso@ubuntu (tty1) (Fri Jun 21 16:30:02 2024):

Hola mundo

tpcaso@ubuntu:~$
```

Haga que un script escriba en la pantalla "Bye Bye" cada vez que se desloguee un usuario.

```
#!/bin/bash
echo "Bye Bye"
sleep 5
```

```
# ~/.bash_logout: executed by bash(1) when login shell exits.

# when leaving the console clear the screen to increase privacy

if [ "$SHLVL" = 1 ]; then
    [ -x /usr/bin/clear_console ] && /usr/bin/clear_console -q
fi
/home/tpcaso/tp/chau.sh
```

```
Bye Bye
```

Haga que un script que escriba en la pantalla "Buen Día" cada vez que se enciende la máquina en Linux.
Haga que un script escriba en la pantalla "Chau Mundo" cada vez que apague la máquina. (No usar opciones del shutdown)

```
#!/bin/bash
inicio() {
        wall "Buen Dia"
}
apagar() {
        wall "Chau Mundo"
}
case "$1" in
        inicio)
                inicio
                ;;
        apagar)
                apagar
                ;;
        *)
        echo $"usage: $0 {iniciolapagar}"
        RETVAL=1
esac
exit 0
```

```
tpcaso@ubuntu:~$ chmod 777 /home/tpcaso/tp/holaChau.sh
tpcaso@ubuntu:~$ cd /etc/init.d
tpcaso@ubuntu:/etc/init.d$ cd /etc/rc0.d
tpcaso@ubuntu:/etc/rc0.d$ sudo ln -s ../init.d/holaChau.sh S0holaChau.sh
[sudo] password for tpcaso:
Lo sentimos, vuelva a intentarlo.
[sudo] password for tpcaso:
tpcaso@ubuntu:/etc/rc0.d$ sudo ln -s ../init.d/holaChau.sh K0holaChau.sh
tpcaso@ubuntu:/etc/rc0.d$
```

3.4. Ejecución de procesos en background

Antes de resolver esta sección instale los siguientes paquetes en la máquina virtual:

nano: editor de texto.

mc: manejador de archivos.

gcc: compilador de C.

libc6-dev: biblioteca estándar de C.

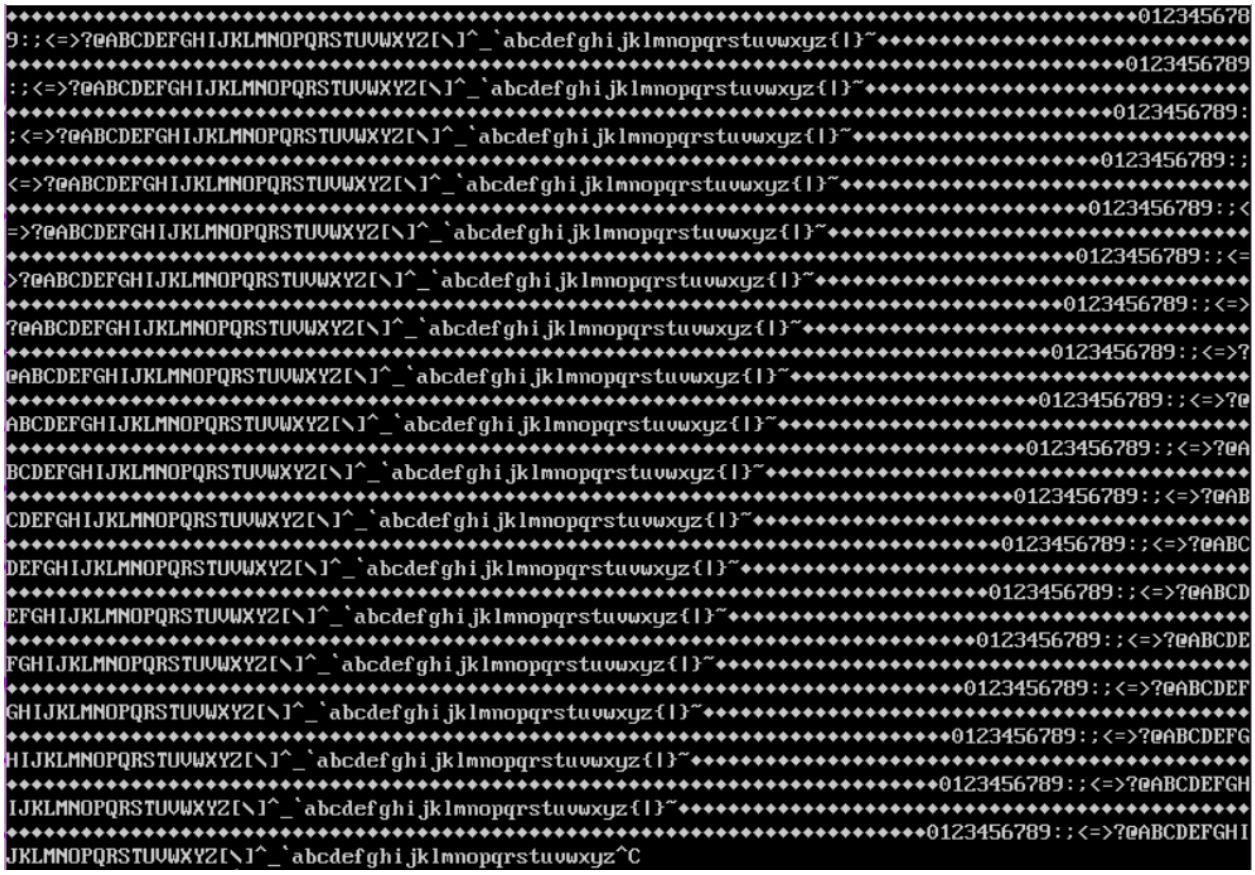
Cree el archivo */home/<usuario>/tp/loop.c*. Compílelo con **gcc**. El programa compilado debe llamarse **loop**.

3.4.1. loop.c

```c
#include <stdio.h>

#define IDGRUPO 200

int main() {
        int i, c;

        while (1) {
                c = 48+i;
                printf("%c",c);
                i++;
                i = i%IDGRUPO;
        }
        return 0;
}
```

1. Correrlo en foreground. Qué sucede? Mate el proceso con Ctrl-c.

```
tpcaso@ubuntu:~/tp$ cc loop.c
tpcaso@ubuntu:~/tp$ ./a.out
```



2. Ahora ejecútelo en background: */usr/src/loop >/dev/null &*. Mate el proceso con el comando *kill*.

```
tpcaso@ubuntu:~/tp$ gcc loop.c -o loop
tpcaso@ubuntu:~/tp$ /home/tpcaso/tp/loop > /dev/null &
[3] 27228
tpcaso@ubuntu:~/tp$ ps
  PID TTY          TIME CMD
 1139 tty1     00:00:00 bash
27202 tty1     00:00:00 nano
27203 tty1     00:00:00 nano
27228 tty1     00:00:11 loop
27229 tty1     00:00:00 ps
tpcaso@ubuntu:~/tp$ kill 27228
tpcaso@ubuntu:~/tp$ ps
  PID TTY          TIME CMD
 1139 tty1     00:00:00 bash
27202 tty1     00:00:00 nano
27203 tty1     00:00:00 nano
27230 tty1     00:00:00 ps
[3]   Terminado              /home/tpcaso/tp/loop > /dev/null
tpcaso@ubuntu:~/tp$ _
```

3.5. IPC y sincronización

3.5.1. Pipes

Muestre con un ejemplo en lenguaje C como realizar un productor consumidor entre dos procesos utilizando dos pipes.

Sugerencia Revise la ayuda de la llamada al sistema pipe para construir el pipe y de fork para crear nuevos procesos.

```
  GNU nano 2.5.3                    Archivo: pipe.c

#include <sys/types.h>
#include <sys/wait.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main(){
        pid_t child = fork(); //Crea el proceso

        switch(child){
                case -1: //Caso de error en el fork
                        perror("Fork error");
                        exit(EXIT_FAILURE);
                case 0: //Caso del proceso hijo
                        printf("Soy el hijo, PID: %d, padre es PID: %d\n", getpid(), getppid());
                        sleep(2); //Trabajo del hijo
                        printf("El hijo termina su trabajo\n");
                        exit(EXIT_SUCCESS);
                default: //Caso del proceso padre
                        printf("Soy el padre, PID: %d, hijo PID: %d\n", getpid(), child);
                        //Espera que el proceso hijo termine
                        if (wait(NULL) == -1){
                                perror("Error en el wait");
                                exit(EXIT_FAILURE);
                        }
                        printf("El hijo ha terminado\n");
        }
        return 0;
}


                              [ 29 líneas leídas ]
^G Ver ayuda  ^O Guardar    ^W Buscar     ^K Cortar Text^J Justificar ^C Posición   ^Y Pág. ant.
^X Salir      ^R Leer fich. ^\ Reemplazar ^U Pegar txt  ^T Ortografía ^_ Ir a línea ^V Pág. sig.
```

```
tpcaso@ubuntu:~/tp$ cc -o pipe pipe.c
tpcaso@ubuntu:~/tp$ ./pipe
Soy el padre, PID: 27257, hijo PID: 27258
Soy el hijo, PID: 27258, padre es PID: 27257
El hijo termina su trabajo
El hijo ha terminado
tpcaso@ubuntu:~/tp$
```

3.5.2. Threads

Antes de resolver este ejercicio instale el paquete *glibc-doc*.

Resuelva el problema de exclusión mutua utilizando threads.

Sugerencia **Revise la ayuda de pthreads, la implementación de threads en Linux, para conocer los mecanismos de creación y destrucción de threads. Además, pthreads provee mecanismos de sincronización que le ayudarán a resolver este ejercicio.**

```
  GNU nano 2.5.3                  Archivo: thread.c

#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#define NUM_THREADS 2
#define ITERACIONES 10

int valor = 0;
pthread_mutex_t mutex; //Mutex para exclusion mutua

void* decrementar(void* arg) {
        int threadID = *((int*)arg); //ID del hilo
        int i;
        printf("Se inicia hilo %d\n", threadID);
        for (i=0; i<ITERACIONES; i++) {
                pthread_mutex_lock(&mutex); //Bloquea el mutex
                valor--;
                printf("Hilo %d / Valor actual: %d\n", threadID, valor);
                pthread_mutex_unlock(&mutex); //Desbloquea el mutex
        }
        printf("Hilo %d finalizado\n", threadID);
        pthread_exit(NULL);
}
```

```
int main() {
        pthread_t threads[NUM_THREADS]; //Identificadores de  hilos
        int threadIDS[NUM_THREADS]; //IDs de hilos
        int rc, i;
        pthread_mutex_init(&mutex, NULL); //Inicia mutex
        for (i=0; i<NUM_THREADS; i++) {
                threadIDS[i] = i+1; //Asigna el ID a cada hilo
                rc = pthread_create(&threads[i], NULL, decrementar, (void*)&threadIDS[i]);
                if(rc) {
                        fprintf(stderr, "Error al crear el hilo %d\n", i+1);
                        exit(EXIT_FAILURE);
                }
        }
        //Espera que todos los hilos finalicen
        for (i=0; i<NUM_THREADS; i++) {
                rc = pthread_join(threads[i], NULL);
                if (rc) {
                        fprintf(stderr, "Error al esperar al hilo %d\n", i+1);
                        exit(EXIT_FAILURE);
                }
        }
        pthread_mutex_destroy(&mutex); //Destruye el mutex
        printf("Valor final del contador: %d\n", valor);
        return 0;
}


^G Ver ayuda  ^O Guardar     ^W Buscar      ^K Cortar Text^J Justificar ^C Posición   ^Y Pág. ant.
^X Salir      ^R Leer fich. ^\ Reemplazar  ^U Pegar txt ^T Ortografía  ^_ Ir a línea ^V Pág. sig.
```

```
tpcaso@ubuntu:~/tp$ cc -pthread -o thread thread.c
tpcaso@ubuntu:~/tp$ ./thread
Se inicia hilo 2
Hilo 2 / Valor actual: -1
Hilo 2 / Valor actual: -2
Hilo 2 / Valor actual: -3
Hilo 2 / Valor actual: -4
Hilo 2 / Valor actual: -5
Hilo 2 / Valor actual: -6
Hilo 2 / Valor actual: -7
Hilo 2 / Valor actual: -8
Hilo 2 / Valor actual: -9
Se inicia hilo 1
Hilo 1 / Valor actual: -10
Hilo 1 / Valor actual: -11
Hilo 1 / Valor actual: -12
Hilo 1 / Valor actual: -13
Hilo 1 / Valor actual: -14
Hilo 1 / Valor actual: -15
Hilo 1 / Valor actual: -16
Hilo 1 / Valor actual: -17
Hilo 1 / Valor actual: -18
Hilo 1 / Valor actual: -19
Hilo 1 finalizado
Hilo 2 / Valor actual: -20
Hilo 2 finalizado
Valor final del contador: -20
tpcaso@ubuntu:~/tp$
```

3.6. El kernel Linux

Antes de resolver esta sección instale los siguientes paquetes en la máquina virtual:

make: utilidad para mantener grupos de programas.

```
tpcaso@ubuntu:~$ sudo apt-get install make
[sudo] password for tpcaso:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Paquetes sugeridos:
  make-doc
Se instalarán los siguientes paquetes NUEVOS:
  make
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 153 kB de archivos.
Se utilizarán 369 kB de espacio de disco adicional después de esta operación.
Des:1 http://ar.archive.ubuntu.com/ubuntu xenial/main i386 make i386 4.1-6 [153 kB]
Descargados 153 kB en 5s (29,9 kB/s)
Seleccionando el paquete make previamente no seleccionado.
(Leyendo la base de datos ... 100270 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../archives/make_4.1-6_i386.deb ...
Desempaquetando make (4.1-6) ...
Procesando disparadores para man-db (2.7.5-1) ...
Configurando make (4.1-6) ...
tpcaso@ubuntu:~$
```

linux-headers-<version>: headers del kernel de Linux.

Sustituya <version> por el resultado del comando *uname -r*.

```
tpcaso@ubuntu:~$ uname -r
4.4.0-142-generic
tpcaso@ubuntu:~$ _
```

```
tpcaso@ubuntu:~$ sudo apt-get install linux-headers-4.4.0-142-generic
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
linux-headers-4.4.0-142-generic ya está en su versión más reciente (4.4.0-142.168).
fijado linux-headers-4.4.0-142-generic como instalado manualmente.
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
tpcaso@ubuntu:~$ _
```

3.6.1. Funcionamiento del kernel Linux

1. Describa la administración del procesador utilizada por omisión en el kernel Linux.

En un sistema con un solo procesador, solo un proceso puede usarlo a la vez. Una vez que un proceso ha terminado su turno, otro proceso toma su lugar. El kernel asigna pequeños intervalos de tiempo a cada proceso, lo que permite que varios procesos parezcan ejecutarse simultáneamente, logrando así la multitarea. Si un proceso no termina en su tiempo asignado, el kernel lo interrumpe, guarda su estado y selecciona otro proceso listo para ejecutarse según su prioridad.

En sistemas con múltiples procesadores o núcleos, el kernel distribuye los procesos de manera equilibrada entre los núcleos disponibles para maximizar el rendimiento y minimizar la inactividad. Esto puede implicar mover procesos entre núcleos, conocido como "migración de procesos", para mantener un balance de carga.

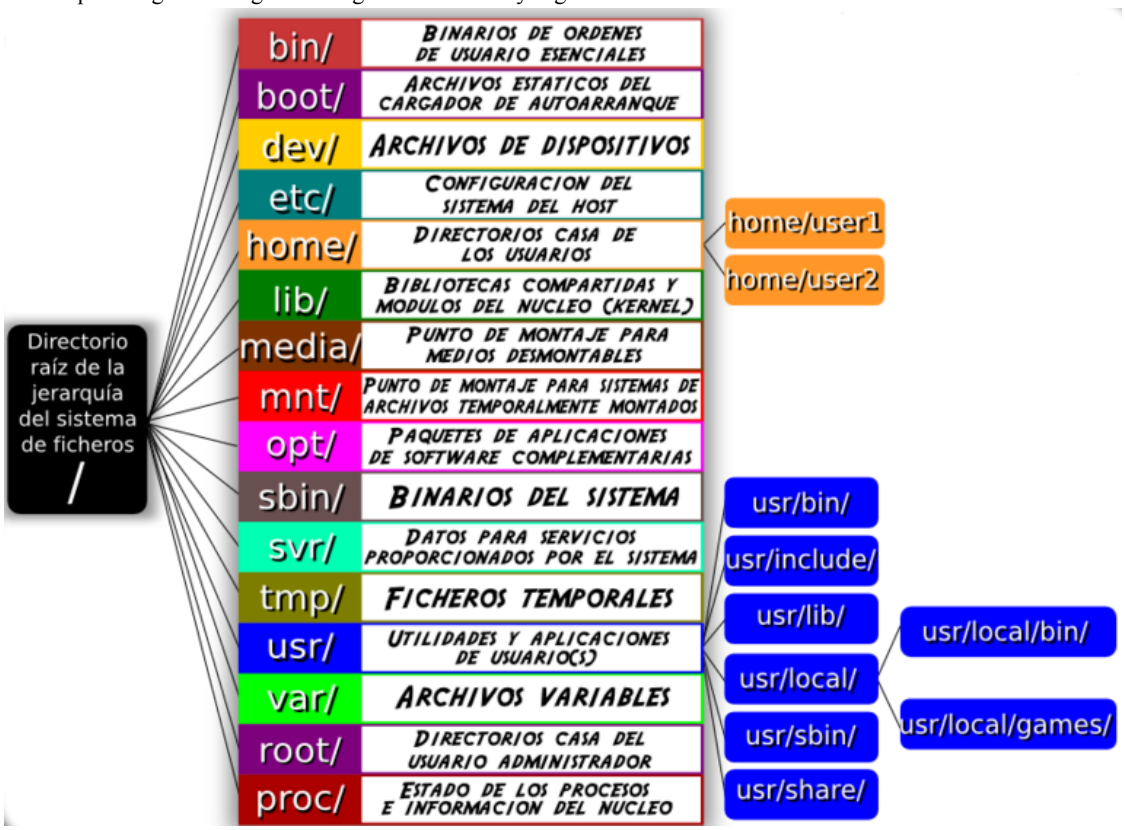
En resumen, la administración del procesador en el kernel de Linux es compleja y eficiente, asignando el tiempo de CPU a los procesos de manera equitativa y priorizando aquellos de alta importancia mediante algoritmos avanzados de planificación.

2. Describa la administración de memoria utilizada por omisión en el kernel Linux.
Linux utiliza Memoria Virtual, extendiendo la RAM con el uso del disco para aumentar la memoria disponible. El kernel escribe bloques de memoria inactivos al disco (área de swap) y los recupera cuando se necesitan, aunque esto es más lento que usar RAM. El swap puede ser un archivo o una partición, siendo la partición más rápida, pero el archivo más flexible para ajustar su tamaño.
El kernel de Linux emplea paginación, dividiendo la memoria en páginas y asignando a cada proceso su propio espacio de direcciones virtuales, traducidas a direcciones físicas mediante una tabla de páginas. Este sistema mejora la eficiencia y la protección de memoria entre procesos.
Para gestionar la memoria libre, Linux usa algoritmos que mantienen una lista de bloques de memoria disponibles y los asigna según sea necesario.
En resumen, la administración de memoria en el kernel de Linux es una combinación de técnicas avanzadas que permiten un uso eficiente de los recursos disponibles. La memoria virtual y el swap proporcionan una extensión flexible de la RAM física, mientras que la paginación y los algoritmos de gestión de memoria aseguran un rendimiento óptimo y protección entre procesos.

3. Describa el sistema de archivos utilizado en la distribución de Linux que instaló en la máquina virtual. Qué capas existen en el kernel Linux para soportar sistemas de archivos sobre dispositivos de bloques?
En Linux, todo se trata como un fichero. Los directorios, archivos y dispositivos se gestionan como ficheros, aunque a los dispositivos se les pueda llamar nodos. Estos elementos se organizan en una estructura jerárquica en forma de árbol, donde el nivel más alto es el directorio raíz, representado por " / ". Todos los demás archivos y directorios se encuentran bajo este directorio raíz.
Para soportar sistemas de archivos en dispositivos de bloques, el kernel de Linux utiliza varias capas:

- **VFS (Virtual File System):** Proporciona una interfaz uniforme para diferentes sistemas de archivos.
- **Sistemas de Archivos Específicos:** Como ext4, XFS y Btrfs, que gestionan la organización de datos y metadatos.
- **Page Cache:** Almacena en caché las páginas de archivos para acelerar el acceso a datos.
- **Buffer Cache:** Similar a Page Cache, pero específica para bloques de disco.
- **Device Drivers:** Permiten la comunicación entre el kernel y el hardware.
- **I/O Scheduler:** Optimiza el orden de las operaciones de entrada/salida.
- **Block Layer:** Coordina las operaciones a nivel de bloque y gestiona las colas de solicitudes de I/O.

Estas capas integradas aseguran una gestión eficiente y segura de los recursos de almacenamiento en Linux.



3.6.2. Módulos de kernel

El kernel de Linux permite ser ampliado en runtime con módulos. Los módulos son objetos de código compilado que pueden ser insertados en runtime al kernel, siendo linkeados contra el kernel al momento de ser insertados. De esta manera puede ampliarse la funcionalidad del kernel en runtime, sin tener que incluir todo el código en el binario original.

3.6.3. Compilando un módulo de kernel

A continuación compilaremos y probaremos un módulo de kernel muy simple. Este módulo simplemente escribe en la consola "*Hola kernel!*" al ser insertado y "*Chau, kernel.*" al ser removido.

Cree el siguiente módulo en el archivo */home/<usuario>/tp/hello.c*.

3.6.4. hello.c

```
  GNU nano 2.5.3                    Archivo: hello.c

//Hello.c
//"Hello world" usando modulos de kernel

//Headers para modulos de kernel
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/init.h>

//Prototipos de las funciones de inicializacion y destrucion
static int __init hello_init(void);
static void __exit hello_exit(void);

/*Informamos al kernel que inicialice el modulo usando hello_init
y que antes de quitarlo use hello_exit*/
module_init(hello_init);
module_exit(hello_exit);

//inicializacion
static int __init hello_init() {
        printk(KERN_ALERT "Hola kernel!\n");
        /*Si devolvemos un valor distinto de cero significa que
        hello_init fallo y el modulo no pudo ser cargado*/
        return 0;
}

//Destruccion
static void __exit hello_exit(){
        printk(KERN_ALERT "Chau, kernel.\n");
}
/*Declaramos que este codigo tiene licencia GPL. De esta manera
no estamos "manchando" el kernel con codigo propietario*/
MODULE_LICENSE("GPL");
                              [ 32 líneas leídas ]
^G Ver ayuda  ^O Guardar    ^W Buscar     ^K Cortar Text^J Justificar ^C Posición   ^Y Pág. ant.
^X Salir      ^R Leer fich. ^\ Reemplazar ^U Pegar txt  ^T Ortografía ^_ Ir a línea ^V Pág. sig.
```

Para compilar este código deberá construir un **Makefile**. Este archivo **Makefile** es utilizado luego por el comando *make* para compilar el módulo con las opciones correctas. En el mismo directorio donde se encuentra **hello.c** cree un archivo **Makefile** conteniendo:

3.6.5. Makefile

```
obj-m = hello.o
KVERSION = $(shell uname -r)
all:
	make -C /lib/modules/$(KVERSION)/build M=$(PWD) modules
clean:
	make -C /lib/modules/$(KVERSION)/build M=$(PWD) clean
```

Luego ejecute:

```
tpcaso@ubuntu:~/tp$ make
make -C /lib/modules/4.4.0-142-generic/build M=/home/tpcaso/tp modules
make[1]: se entra en el directorio '/usr/src/linux-headers-4.4.0-142-generic'
  CC [M]  /home/tpcaso/tp/hello.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/tpcaso/tp/hello.mod.o
  LD [M]  /home/tpcaso/tp/hello.ko
make[1]: se sale del directorio '/usr/src/linux-headers-4.4.0-142-generic'
tpcaso@ubuntu:~/tp$
```

Para compilar el módulo. Finalmente, pruebe insertar el módulo usando:

```
tpcaso@ubuntu:~/tp$ sudo insmod hello.ko
[sudo] password for tpcaso:
[15716.264513] Hola kernel!
tpcaso@ubuntu:~/tp$ sudo rmmod hello.ko
[15731.130661] Chau, kernel.
tpcaso@ubuntu:~/tp$
```

3.7. Un módulo propio —> mañana

Se pide implementar un módulo de kernel, que maneje un dispositivo tipo caracter "/dev/probabilidad", y que ante una lectura devuelva dos letras mayúsculas en forma aleatoria. Para tener un control sobre el módulo se pide también que genere un archivo en /proc para poder leer de él la cantidad de lecturas realizadas al dispositivo y escribirle una nueva semilla en caso de ser necesaria.

$ echo 5 > /proc/probabilidad **(Pone una nueva semilla)**
$ cat /proc/probabilidad **(Muestra cantidad de lecturas realizadas)**
0
$ cat /dev/probabilidad
AZ
$ cat /proc/probabilidad **(Muestra cantidad de lecturas realizadas)**
1

3.8. Temas de sistemas operativos

3.8.1. File system

En el ejercicio 3.1.12 se hace mención a los hardlinks como apuntadores a un mismo espacio de disco. Cómo se llama ese espacio de disco, que estructura tiene y por qué se pueden borrar los hardlinks sin borrar al archivo.

Un hardlink se puede borrar sin eliminar el archivo porque los datos en disco solo se eliminan cuando todos los enlaces que apuntan a esos datos han sido eliminados. Cada hardlink es una referencia al mismo conjunto de datos en el disco. Los hardlinks apuntan al mismo espacio de disco llamado inode, que contiene información sobre el archivo, como permisos, propietario y

direcciones de los bloques de datos. El archivo solo se elimina cuando todos los enlaces al inode han sido borrados, es decir, cuando el contador de enlaces del inode llega a cero. Los hardlinks son simplemente entradas de directorio que apuntan al mismo inodo, sin estructura propia.

3.8.2. Prioridades

Genere tres versiones del programa loop (3.4.1) (loop1, loop2 y loop3) y ejecútelos en background.
Logre que loop3 ejecute más rápido que los otros dos (prioridad). Obtenga los tiempos de ejecución de cada uno de ellos (uso de procesador) y sus estados. Explique detalladamente cómo logra obtener esta información.

1) Se crean los procesos loop1.c, loop2.c y loop3.c (Contienen el mismo código que loop.c).

```
tpcaso@ubuntu:~/tp$ ls
a.out     hello.ko     holaChau.sh  loop2.c      Makefile        pipe.c           script.sh
chau.sh   hello.mod.c  hola.sh      loop3.c      modules.order   salidaGrupo.txt  thread
config    hello.mod.o  loop         loop.c       Module.symvers  salida.txt       thread.c
hello.c   hello.o      loop1.c      loop.c.save  pipe            saludo.sh
tpcaso@ubuntu:~/tp$
```

2) Se compilan los tres procesos.

```
tpcaso@ubuntu:~/tp$ gcc loop1.c -o loop1
tpcaso@ubuntu:~/tp$ gcc loop2.c -o loop2
tpcaso@ubuntu:~/tp$ gcc loop3.c -o loop3
tpcaso@ubuntu:~/tp$ _
```

3) Se ejecutan los procesos en segundo plano.

```
tpcaso@ubuntu:~/tp$ /home/tpcaso/tp/loop1 > /dev/null &
[1] 1198
tpcaso@ubuntu:~/tp$ /home/tpcaso/tp/loop2 > /dev/null &
[2] 1199
tpcaso@ubuntu:~/tp$ /home/tpcaso/tp/loop3 > /dev/null &
[3] 1200
tpcaso@ubuntu:~/tp$
```

4) Se comprueba si efectivamente están siendo ejecutados.

```
tpcaso@ubuntu:~/tp$ ps
  PID TTY          TIME CMD
 1182 tty1     00:00:00 bash
 1198 tty1     00:00:23 loop1
 1199 tty1     00:00:12 loop2
 1200 tty1     00:00:08 loop3
 1201 tty1     00:00:00 ps
tpcaso@ubuntu:~/tp$ _
```

5) Se usa el comando top para obtener más información de los procesos. Como se puede observar el PR en los tres procesos es de 20 y el NI 0.

```
top - 15:56:10 up 2 min,  1 user,  load average: 2,10, 0,69, 0,25
Tareas: 103 total,   4 ejecutar,   99 hibernar,    0 detener,    0 zombie
%Cpu(s): 99,0 usuario,  1,0 sist,  0,0 adecuado,  0,0 inact,  0,0 en espera,  0,0 hardw int,  0,0 s
KiB Mem :  2062140 total,  1887448 free,    48916 used,   125776 buff/cache
KiB Swap:  1486672 total,  1486672 free,        0 used.  1806420 avail Mem

  PID USUARIO   PR  NI    VIRT    RES    SHR S %CPU %MEM     HORA+ ORDEN
 1198 tpcaso    20   0    2204    512    468 R 33,6  0,0   0:34.88 loop1
 1199 tpcaso    20   0    2204    536    492 R 33,2  0,0   0:24.44 loop2
 1200 tpcaso    20   0    2204    500    452 R 33,2  0,0   0:20.71 loop3
    1 root      20   0    6488   4884   3800 S  0,0  0,2   0:01.26 systemd
    2 root      20   0       0      0      0 S  0,0  0,0   0:00.00 kthreadd
    3 root      20   0       0      0      0 S  0,0  0,0   0:00.02 ksoftirqd/0
    4 root      20   0       0      0      0 S  0,0  0,0   0:00.00 kworker/0:0
    5 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 kworker/0:0H
    6 root      20   0       0      0      0 S  0,0  0,0   0:00.02 kworker/u2:0
    7 root      20   0       0      0      0 S  0,0  0,0   0:00.12 rcu_sched
    8 root      20   0       0      0      0 S  0,0  0,0   0:00.00 rcu_bh
    9 root      rt   0       0      0      0 S  0,0  0,0   0:00.00 migration/0
   10 root      rt   0       0      0      0 S  0,0  0,0   0:00.00 watchdog/0
   11 root      20   0       0      0      0 S  0,0  0,0   0:00.00 kdevtmpfs
   12 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 netns
   13 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 perf
   14 root      20   0       0      0      0 S  0,0  0,0   0:00.00 khungtaskd
   15 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 writeback
   16 root      25   5       0      0      0 S  0,0  0,0   0:00.00 ksmd
   17 root      39  19       0      0      0 S  0,0  0,0   0:00.00 khugepaged
   18 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 crypto
   19 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 kintegrityd
   20 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 bioset
   21 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 kblockd
   22 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 ata_sff
   23 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 md
   24 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 devfreq_wq
   25 root      20   0       0      0      0 S  0,0  0,0   0:00.36 kworker/u2:1
   26 root      20   0       0      0      0 S  0,0  0,0   0:00.03 kworker/0:1
   28 root      20   0       0      0      0 S  0,0  0,0   0:00.00 kswapd0
```

6) Se aplica el comando renice para asignarle una nueva prioridad, -n -10 especifica la nueva prioridad que se le asignará al proceso (En este caso -20 es la más alta posible). -p se utiliza para especificar el PID del proceso al que se le cambiará la prioridad (En este caso el 1200 que corresponde al loop 3).

```
tpcaso@ubuntu:~/tp$ sudo renice -n -20 -p 1200
[sudo] password for tpcaso:
1200 (process ID) prioridad antigua 0, prioridad nueva -20
tpcaso@ubuntu:~/tp$ _
```

7) Se comprueba que el proceso haya cambiado de prioridad utilizando el comando top. Ahora el proceso loop 3 posee el PR en 0 y el NI en -20, por lo que tiene la prioridad más alta.

```
top - 15:58:40 up 5 min,  1 user,  load average: 2,93, 1,61, 0,67
Tareas:  94 total,   4 ejecutar,   90 hibernar,    0 detener,    0 zombie
%Cpu(s): 99,7 usuario,  0,3 sist,  0,0 adecuado,  0,0 inact,  0,0 en espera,  0,0 hardw int,  0,0 s
KiB Mem :  2062140 total,  1886744 free,    48856 used,   126540 buff/cache
KiB Swap:  1486672 total,  1486672 free,        0 used.  1806096 avail Mem

  PID USUARIO   PR  NI    VIRT    RES    SHR S %CPU %MEM     HORA+ ORDEN
 1200 tpcaso     0 -20    2204    500    452 R 97,7  0,0   1:38.64 loop3
 1198 tpcaso    20   0    2204    512    468 R  1,0  0,0   1:10.71 loop1
 1199 tpcaso    20   0    2204    536    492 R  1,0  0,0   1:00.27 loop2
    1 root      20   0    6488   4884   3800 S  0,0  0,2   0:01.26 systemd
    2 root      20   0       0      0      0 S  0,0  0,0   0:00.00 kthreadd
    3 root      20   0       0      0      0 S  0,0  0,0   0:00.02 ksoftirqd/0
    4 root      20   0       0      0      0 S  0,0  0,0   0:00.00 kworker/0:0
```

3.8.3. Parámetros del Kernel

Determine la cantidad de memoria RAM que usa su sistema, ahora encuentre una manera para que use menor cantidad de memoria RAM. Explique detalladamente cómo logra obtener este cambio y cómo obtiene la información.

```
tpcaso@ubuntu:~/tp$ free -m
              total        used        free      shared  buff/cache   available
Memoria:       2013          51         165           3        1796        1698
Swap:           974           0         974
tpcaso@ubuntu:~/tp$ _
```

```
root@ubuntu:~# sync; echo 3 > /proc/sys/vm/drop_caches
root@ubuntu:~# free -m
              total        used        free      shared  buff/cache   available
Memoria:       2013          53        1911           3          49        1794
Swap:           974           0         974
root@ubuntu:~#
```

Linux tiene 3 opciones para limpiar el caché sin interrumpir ningún proceso o servicio. Todos poseen la estructura:
sync; echo X > /proc/sys/vm/drop_caches
donde X puede ser reemplazado por 1 (Solo limpia caché), 2 (Limpia dentries e inodos) y 3 (Limpia caché, dentries e inodos).
Sync limpia el buffer del sistema.

3.8.4. Administración de Memoria

Determine el tamaño de la partición swap que está utilizando. Amplíe el tamaño del swap por medio de un archivo en un FS. Hágalo persistente. Explique detalladamente cómo logra obtener este cambio.

1) Veo memoria disponible.

```
root@ubuntu:~# free -m
              total        used        free      shared  buff/cache   available
Memoria:       2013          53        1907           3          52        1792
Swap:           974           0         974
```

2) Reviso cuánta memoria hay disponible para usarla en el swap.

```
root@ubuntu:~# df -h
S.ficheros      Tamaño Usados  Disp Uso% Montado en
udev              988M       0  988M   0% /dev
tmpfs             202M   3,2M  199M   2% /run
/dev/sda1         3,0G   2,1G  775M  73% /
tmpfs            1007M       0 1007M   0% /dev/shm
tmpfs             5,0M       0  5,0M   0% /run/lock
tmpfs            1007M       0 1007M   0% /sys/fs/cgroup
tmpfs             202M       0  202M   0% /run/user/1000
root@ubuntu:~#
```

3) Se utiliza fallocate para crear un fichero del tamaño indicado de forma inmediata, se verifica su existencia y se le otorga solo permiso de lectura.

```
root@ubuntu:~# fallocate -l 500mb /swapfile
root@ubuntu:~# ls -lh /swapfile
-rw-r--r-- 1 root root 477M jun 24 18:51 /swapfile
root@ubuntu:~# chmod 600 /swapfile
root@ubuntu:~# _
```

4) Se indica al sistema que el archivo será utilizado como memoria swap mediante el comando mkswap. Se activa el swap con el swapon (-s para mostrar).

```
root@ubuntu:~# mkswap /swapfile
Setting up swapspace version 1, size = 476,9 MiB (499994624 bytes)
sin etiqueta, UUID=4a1b0b40-c76c-40cb-8b5b-113279af7858
root@ubuntu:~# swapon /swapfile
root@ubuntu:~# swapon -s
Filename                                Type            Size    Used    Priority
/dev/sda5                               partition       998396  0       -1
/swapfile                               file            488276  0       -2
```

5) Se comprueba que cambió el valor del swap.

```
root@ubuntu:~# free -m
              total        used        free      shared  buff/cache   available
Memoria:       2013          53        1904           3          56        1790
Swap:          1451           0        1451
root@ubuntu:~# _
```

6) Para que el sistema habilite de forma automática el archivo se debe modificar el fstab. Se edita realizando "nano /etc/fstab" (No se usa sudo porque somos root).

```
 GNU nano 2.5.3                  Archivo: /etc/fstab                       Modificado

# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/sda1 during installation
UUID=43f57774-60de-4a88-a302-b718c8674e22 /               ext4    errors=remount-ro 0       1
# swap was on /dev/sda5 during installation
UUID=0982d752-83a7-4515-9e00-900d32fe6d9b none            swap    sw              0       0
/swapfile none swap sw 0 0
```

7) Se comprueba que la memoria swap se mantiene apagando y volviendo a encender la MV.

```
root@ubuntu:~# free -m
              total        used        free      shared  buff/cache   available
Memoria:       2013          49        1799           3         164        1746
Swap:          1451           0        1451
root@ubuntu:~#
```

3.8.5 Administración del Procesador

Escriba un breve informe comparativo sobre la administración de memoria y "scheduling" de procesador del sistema operativo WINDOWS vs LINUX.

ADMINISTRACIÓN DE MEMORIA	
WINDOWS	**LINUX**
Usa conjuntos de trabajos por proceso con tamaño dinámico empleando el algoritmo de reloj.	Usa un único conjunto de trabajo bajo el algoritmo del reloj.
Permite definir el tamaño de memoria virtual que tiene disponible, aunque de ser necesario el SO puede ampliar este espacio.	Una porción de la RAM se asigna al kernel, el resto es memoria dinámica. Las políticas de asignación son por petición del kernel y por petición del usuario.
Se utiliza paginación segmentada y un liberador de memoria que actúa una vez por segundo.	Utiliza paginación LRU y el liberador de memoria se ejecuta solo cuando es necesario.
Posee 8 niveles de prioridad para la memoria virtual.	Memoria virtual sin prioridades.

ADMINISTRACIÓN DE MEMORIA	
WINDOWS	**LINUX**
Usa conjuntos de trabajos por proceso con tamaño dinámico empleando el algoritmo de reloj.	Usa un único conjunto de trabajo bajo el algoritmo del reloj.
Permite definir el tamaño de memoria virtual que tiene disponible, aunque de ser necesario el SO puede ampliar este espacio.	Una porción de la RAM se asigna al kernel, el resto es memoria dinámica. Las políticas de asignación son por petición del kernel y por petición del usuario.
Se utiliza paginación segmentada y un liberador de memoria que actúa una vez por segundo.	Utiliza paginación LRU y el liberador de memoria se ejecuta solo cuando es necesario.
El modelo de paginación es basado en una estructura de árbol, donde la raíz del árbol es un directorio de páginas, cada proceso dispone de un directorio con punteros a tablas y cada tabla posee 1024 entradas apuntando a páginas.	Usa un modelo de paginación similar al del i386.
SCHEDULING	
Usa una planificación multitarea cooperativa y la multitarea con derecho preferente.	Da prioridad a los procesos de tiempo real antes que a los demás procesos. Todos los procesos se encuentran organizados de manera jerárquica, entonces todos tienen un proceso padre que los ejecuta. Todos tienen un ID y son clasificados como “Procesos de usuario” (son usados por el usuario y no tienen acceso a todas las instrucciones) y “procesos demonio” (No interactúan con el usuario).

(The End ...)