Universidad de Huelva

Escuela Técnica Superior de Ingeniería

Universidad de Huelva

Memoria de Prácticas

ADMINISTRACIÓN DE SERVIDORES

Grado en Ingeniería Informática

Autor: Alejandro Gordillo Pedraza

6 de noviembre de 2023

Resumen

Esta práctica consiste en crear una Máquina Virtual (VM)en ProxMox e instalarle CentOs7.
Una vez tengamos instalada esta variante de Linux deberemos cambiar el kernel que nos viene de serie por uno de nuestra elección. En nuestro caso elegiremos unos un poco más actual (4.10).

Antes de elegir este kernel empezó probando uno de los mas actuales 6.* pero no conseguí configurarlo tras varios intentos y varias máquinas, así que probe con uno que fuera levemente mas actual que el que teníamos de default y conseguimos avanzar en nuestro aprendizaje.

Índice general

Contenido

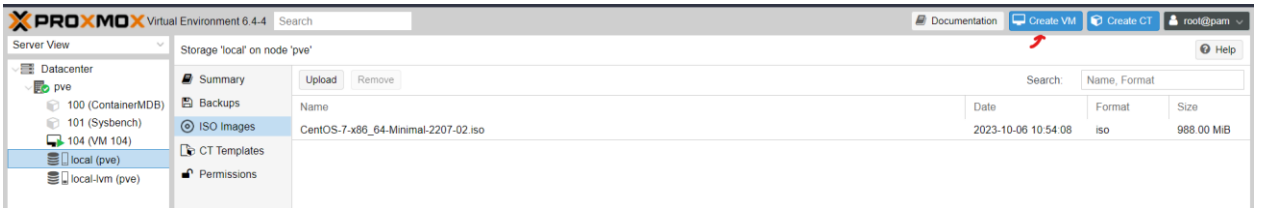
Capítulo 1

Instalación de CentOs 7

1.1 Creación el entorno de pruebas

Cree una máquina virtual completa, con las características antes indicadas, en la instancia de Proxmox creada para cada estudiante.

Antes de crear una máquina virtual tenemos que descargarnos la imagen ISO en internet (Yo me descargué la versión minimalista de 2009). Una vez se descarga por completo, nos vamos a Proxmox y vamos a las direcciones que están en azul y una vez ahí le damos a Upload y subimos la imagen



Una vez hecho esto, le damos a Create VM y la configuramos con 1 Gb de Ram, 2 CPU y 32 GB de disco duro.

1.2 Una vez creada la maquina virtual, instale CentOs 7

Una vez creada nuestra VM con las especificaciones que nos pedían, nos aparecerá un menú con 3 opciones, elegiremos la opción de **Test this media a install CentOs 7.**

Seguidamente nos realizará un test y nos abrirá una interfaz grafica de instalación, la cual se ira colocando automáticamente todo menos unos datos que deberemos poner manualmente.

1º Destino de instalación, aquí podremos seleccionar en que disco duro queremos realizar la instalación o como hacer las particiones de nuestro disco, en nuestro caso como solo tenemos un disco y queremos que las particiones las haga automáticas simplemente abrimos la opción y damos a listo.

2º Conexión Ethernet, debemos abrir esta opción y activar la conexión LAN, debido que si omitimos este paso cada vez que arranquemos nuestra maquina deberemos meter el comando **eth 0** para poder tener acceso a internet.

3º Una vez que corregimos esas dos opciones le daremos a siguiente o seguir con la instalación, y nos pedirá una contraseña para el root y un usuario (opcional). **IMPORTANTE** acordarse de la contraseña del root pues la necesitaremos cada vez que arranquemos nuestra máquina.

Finalmente, solo quedaría esperar a que termine la instalación de nuestro sistema. Una vez que termine se nos iniciara una línea de comandos para poder trabajar sobre ella.

1.3 Comprobar SSH

Compruebe que el servicio ssh se encuentra activo y configurado adecuadamente. En caso de no estarlo, siga lo indicado en "How to Enable, Install, & Configure SSH on CentOS 7 | PhoenixNAP KB."

Si hemos activado la conexión a ethernet en los pasos anteriores deberíamos tener el ssh activado por default.

Systemctl status sshd

```
[root@localhost ~]# systemctl status sshd
● sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enabled)
   Active: active (running) since lun 2023-10-30 09:50:22 CET; 24s ago
     Docs: man:sshd(8)
           man:sshd_config(5)
 Main PID: 1012 (sshd)
   CGroup: /system.slice/sshd.service
           └─1012 /usr/sbin/sshd -D

oct 30 09:50:22 localhost.localdomain systemd[1]: Starting OpenSSH server daemon...
oct 30 09:50:22 localhost.localdomain sshd[1012]: Server listening on 0.0.0.0 port 22.
oct 30 09:50:22 localhost.localdomain sshd[1012]: Server listening on :: port 22.
oct 30 09:50:22 localhost.localdomain systemd[1]: Started OpenSSH server daemon.
[root@localhost ~]#
```

En caso de no estar activado deberemos activarlo haciendo uso del siguiente comando:

systemctl start sshd

1.4 Instalando un núcleo personalizado

Descargar el código fuente de una versión del núcleo igual o superior a la que ahora mismo está ejecutando.

Para saber que núcleo tiene nuestro CentOs basta con poner:

uname -r

```
[root@localhost ~]# cd /usr/src
[root@localhost src]# uname -r
3.10.0-1160.71.1.el7.x86_64
[root@localhost src]# _
```

Con esto, sabemos que nuestra versión del núcleo es la 3.10.0 y descargamos otra con los siguientes pasos:

Primero nos vamos al directorio usr/src con el comando cd usr/src.

Luego descargamos el fichero con el código fuente y el fichero de firma con los comandos:

wget https://mirrors.edge.kernel.org/pub/linux/kernel/v4.x/linux-4.10.tar.xz

wget https://mirrors.edge.kernel.org/pub/linux/kernel/v4.x/linux-4.10.tar.sign

En el caso de no tener instalada la funcionalidad wget, debemos usar el comando **yum install wget**

1.5 Verifican la identidad e integridad del núcleo descargado antes de proceder al desempaquetado.

Para verificar que el núcleo es fiable confirmamos la firma del autor, para ello utilizaremos el siguiente comando:

gpg --verify linux-4.10.tar.sign

Donde en el comando Linux-4.10.tar.sign es nuestro kernel seleccionado y descargado.

Si nos falla el comando es porque la clave no esta disponible y necesitaremos comprobarlo con la clave pública:

gpg --recv-keys <ID_de_la_clave>

1.6 Desempaque el nuevo núcleo en el directorio.

Para realizar este paso hay que ver que tipo de empaquetado tiene nuestro kernel, en nuestro caso tenemos .tar.xz así que los descomprimiremos con el siguiente comando:

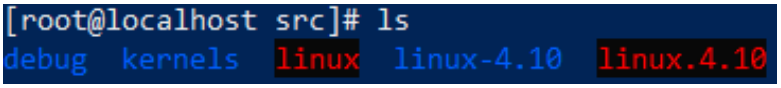
tar xvf Linux-4.10.tar.xz -C /usr/src/

Este comando nos permite descomprimir los tipos de empaquetado con un solo comando y nos lo extrae en el directorio /usr/src, que es donde queremos tener nuestro kernel para el posterior compilado e instalación.

Si quisiéramos ir haciéndolo de a poco, porque no tengamos los dos empaquetados serian estos comandos:

xz -d para él .xz

tar xf para él .tar



1.7 Prepare el fichero de configuración tomando como referencia alguno de los ya instalados.

Aquí se nos presentan dos opciones de hacerlo.

Una seria usando el archivo config de nuestro kernel actual que sabemos que ya funciona, el cual deberíamos poder encontrar haciendo un filtrado ls.

ls /boot/config*

Y luego lo copiariamos a la carpeta de nuestro nuevo kernel.

cp /boot/config-3.10.0-1160.el7.x86_64 /usr/src/linux-4.10/.config

La segunda opción, que es la que he utilizado yo, consiste en usar una de la opciones de los comandos del paquete de instrucciones de **make**.

make menuconfig

Con este comando lo que vamos a obtener es una interfaz grafica para crear nuestro propio config para el kernel que queremos instalar, en mi caso deje todo de serie. Para ello simplemente ejecutamos el comando y en la interfaz gráfica **Save** y **Exit**.

1.8 Compile el núcleo y sus módulos.

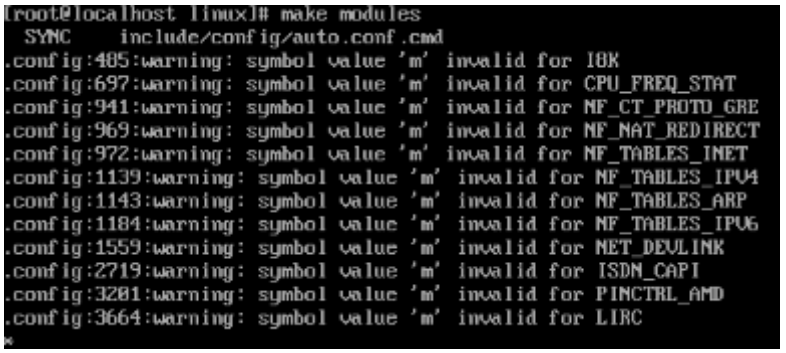
Antes de comenzar con este paso, e incluso antes de usar el comando de make menuconfig deberemos instalar las herramientas entre las que se encuentrar los make.

Para esto usaremos el siguiente comando:

yum group install "Development Tools"

Para compilar nuestro nuevo kernel deberemos estar en el directorio /usr/src y na vez en el ejecutaremos el comando **make**

El siguiente paso compilar los módulos del kernel y para ello haremos uso del comando **make modules**



Veremos que nos da diferentes warning, pero mientras ninguno de ellos sea algo en concreto que necesitemos de alguna forma en específico, todo debería estar okey. Estos datos se solucionan en el config antes de la compilación del núcleo y de los módulos.

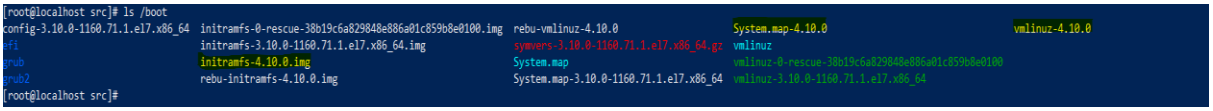
1.9 Instale el núcleo y sus módulos.

Después de hacer el make en el apartado anterior, hay que ejecutar el comando:

make bzImage

Esto nos permitirá generar el Kernel main file, en algunos kernel mas actuales este puede ser generado por el make, sin necesidad de usar este comando. En nuestro caso necesitamos hacer uso de el para poder generarlo.

Ahora cuando acudamos con un ls a nuestro directio boot deberíamos tener en los diferentes archivos necesarios para seguir con nuestra instalación.



Para instalar los módulos simplemente deberemos usar el comando:

make modules_install

Esto instalará los módulos de nuestro nuevo kernel, y deberá aparecernos como última línea depmod y sin errores.

Es una práctica común y considera como buena, una vez instalados los módulos de nuestro nuevo kernel hacer uso del comando

depmod <versión del kernel> (depmod 4.10.0 en nuestro caso)

Esto asegurara que los módulos se actualicen y ejecuten adecuadamente en nuestro nuevo sistema.

Importante: Todos estos pasos deben realizarse desde el directorio del kernel que queremos instalar, es decir /usr/src/Linux-4.10

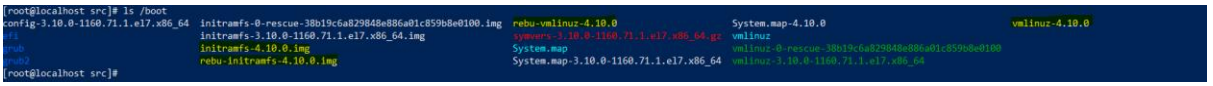
1.10 Haga una copia del núcleo y RAM Disk actuales. Los nuevos ficheros se llamarán como los antiguos añadiendo un prefijo que identifique al alumno.

Nos debemos dirigir al directorio /boot y hacer una copia de estos dos archivos:

cp vmlinuz-4.10.0 rebu-vmlinuz-4.10.0

cp initramfs-4.10.0.img rebu-initramfs-4.10.0.img

El ultimo archivo que hemos copiado sería el Ramdisk y el primero el generado por bzImage.



1.11 Creación del RAM Disk

Usando la utilidad mkinitrd cree una RAM Disk para el núcleo que actualmente está en ejecución.

Para esto debemos acceder al directorio /boot y usar las funcionalidades de **mknitrd**

mknitrd -f -v rebu-initramfs-4.10.0.img rebu-vmlinuz-4.10.0

Compruebe los contenidos del RAM Disk creado, indique los módulos que contiene.

Para ello utilizo el comando:

lsinitrd /boot/rebu-initramfs-4.10.0.img

Los módulos son los siguientes (Los que empiezan por etc)

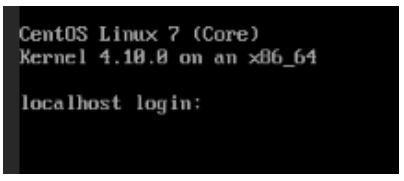
```
crw-r--r--   1 root     root       1,  11 Nov  2 04:34 dev/kmsg
crw-r--r--   1 root     root       1,   3 Nov  2 04:34 dev/null
lrwxrwxrwx   1 root     root           7 Nov  2 04:34 bin -> usr/bin
drwxr-xr-x   2 root     root           0 Nov  2 04:34 dev
drwxr-xr-x  12 root     root           0 Nov  2 04:34 etc
drwxr-xr-x   2 root     root           0 Nov  2 04:34 etc/cmdline.d
drwxr-xr-x   2 root     root           0 Nov  2 04:34 etc/conf.d
-rw-r--r--   1 root     root         124 Nov  2 04:34 etc/conf.d/systemd.conf
-rw-r--r--   1 root     root         262 Sep 30  2020 etc/dhclient.conf
-rw-r--r--   1 root     root           0 Nov  2 04:34 etc/fstab.empty
-rw-r--r--   1 root     root         172 Nov  2 04:34 etc/group
-rw-r--r--   1 root     root          22 Oct 31 08:07 etc/hostname
-rw-r--r--   1 root     root         170 Nov  2 04:34 etc/initrd-release
-rw-r--r--   1 root     root        8043 Nov  2 04:34 etc/ld.so.cache
-rw-r--r--   1 root     root          28 Feb 27  2013 etc/ld.so.conf
drwxr-xr-x   2 root     root           0 Nov  2 04:34 etc/ld.so.conf.d
-rw-r--r--   1 root     root          26 Feb 23  2022 etc/ld.so.conf.d/bind-export-x86_64.conf
-rw-r--r--   1 root     root          19 Aug  9  2019 etc/ld.so.conf.d/dyninst-x86_64.conf
-r--r--r--   1 root     root          63 Jun 28  2022 etc/ld.so.conf.d/kernel-3.10.0-1160.71.1.el7.x86_64.conf
-rw-r--r--   1 root     root          17 Oct  1  2020 etc/ld.so.conf.d/mariadb-x86_64.conf
drwxr-xr-x   2 root     root           0 Nov  2 04:34 etc/libnl
-rw-r--r--   1 root     root        1130 Aug  3  2017 etc/libnl/classid
-rw-r--r--   1 root     root          19 Oct 31 08:07 etc/locale.conf
drwxr-xr-x   2 root     root           0 Nov  2 04:34 etc/lvm
-rw-r--r--   1 root     root       95859 Nov  2 04:34 etc/lvm/lvm.conf
-r--r--r--   1 root     root          33 Oct 31 08:00 etc/machine-id
drwxr-xr-x   2 root     root           0 Nov  2 04:34 etc/modprobe.d
-rw-r--r--   1 root     root         215 Jun 28  2022 etc/modprobe.d/dccp-blacklist.conf
-rw-r--r--   1 root     root         166 Apr 28  2021 etc/modprobe.d/firewalld-sysctls.conf
-rw-r--r--   1 root     root         674 Mar 21  2019 etc/modprobe.d/tuned.conf
lrwxrwxrwx   1 root     root          17 Nov  2 04:34 etc/mtab -> /proc/self/mounts
lrwxrwxrwx   1 root     root          14 Nov  2 04:34 etc/os-release -> initrd-release
-rw-r--r--   1 root     root         101 Nov  2 04:34 etc/passwd
drwxr-xr-x   2 root     root           0 Nov  2 04:34 etc/plymouth
-rw-r--r--   1 root     root          72 Oct  1  2020 etc/plymouth/plymouthd.conf
-rw-r--r--   1 root     root         449 Nov 16  2020 etc/sysctl.conf
drwxr-xr-x   2 root     root           0 Nov  2 04:34 etc/sysctl.d
lrwxrwxrwx   1 root     root          14 Nov  2 04:34 etc/sysctl.d/99-sysctl.conf -> ../sysctl.conf
drwxr-xr-x   2 root     root           0 Nov  2 04:34 etc/systemd
-rw-r--r--   1 root     root        1047 Nov  2 04:34 etc/systemd/journald.conf
-rw-r--r--   1 root     root        1552 Jan 13  2022 etc/systemd/system.conf
drwxr-xr-x   3 root     root           0 Nov  2 04:34 etc/udev
drwxr-xr-x   2 root     root           0 Nov  2 04:34 etc/udev/rules.d
-rw-r--r--   1 root     root         142 Sep 12  2013 etc/udev/rules.d/11-dm.rules
-rw-r--r--   1 root     root         681 Nov  2 04:34 etc/udev/rules.d/59-persistent-storage-dm.rules
-rw-r--r--   1 root     root         298 Nov  2 04:34 etc/udev/rules.d/59-persistent-storage.rules
-rw-r--r--   1 root     root        1031 Nov  2 04:34 etc/udev/rules.d/61-persistent-storage.rules
-rw-r--r--   1 root     root         776 Sep 12  2013 etc/udev/rules.d/64-lvm.rules
-rw-r--r--   1 root     root          49 Jan 13  2022 etc/udev/udev.conf
-rw-r--r--   1 root     root          37 Oct 31 08:07 etc/vconsole.conf
-rw-r--r--   1 root     root        1982 Dec 15  2020 etc/virc
lrwxrwxrwx   1 root     root          23 Nov  2 04:34 init -> usr/lib/systemd/systemd
```

Compruebe que puede arrancar con el nuevo núcleo creado

En mi caso no tuve que modificar el archivo del boot loader porque una vez que ya reinicié el sistema me arrancaba sin problemas en el nuevo kernel.

Por lo tanto, seleccione el nuevo kernel al reiniciar:

El kernel justo arriba del seleccionado en la imagen y una vez que inicia nos confirma que ha cargado correctamente el 4.10.0

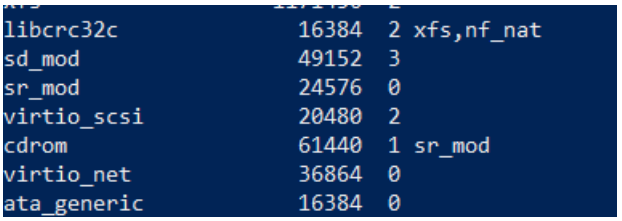


1.12 Gestión de módulos.

Localice el módulo que nos permite tener acceso a la unidad de cdrom.

Para encontrar el módulo que gestiona el cdrom nos vamos a /lib/modules y usamos el comando

lsmod

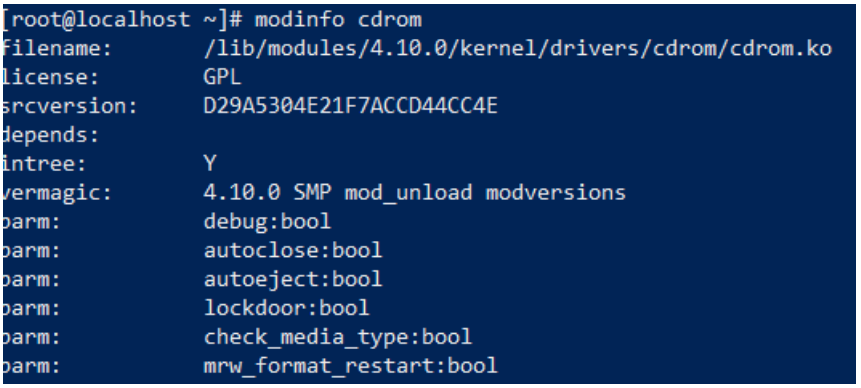


En la foto encontramos diferentes módulos, pero a nosotros nos interesa el cdrom y el sr_mod que es el que usa al cdrom.

Indique los parámetros que admite dicho módulo

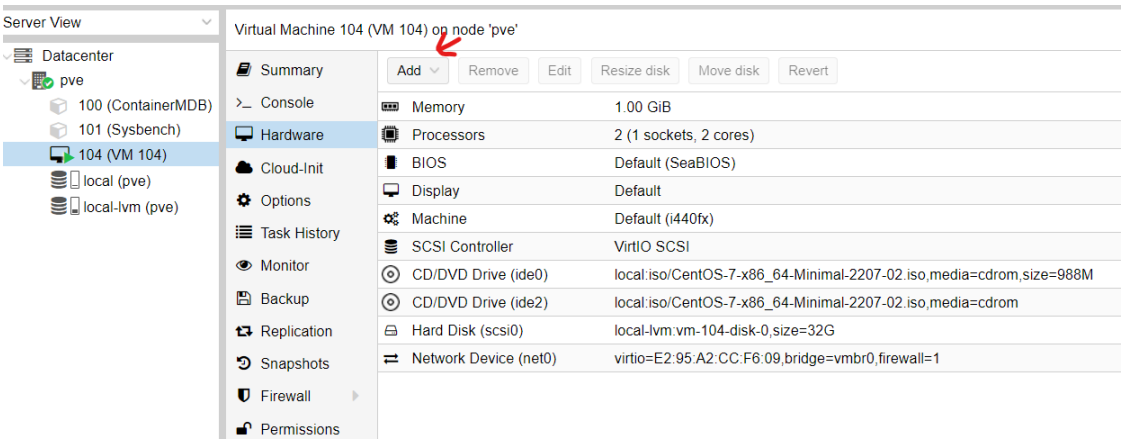
Para esto basta con poner el siguiente comando que nos dará unos parm:

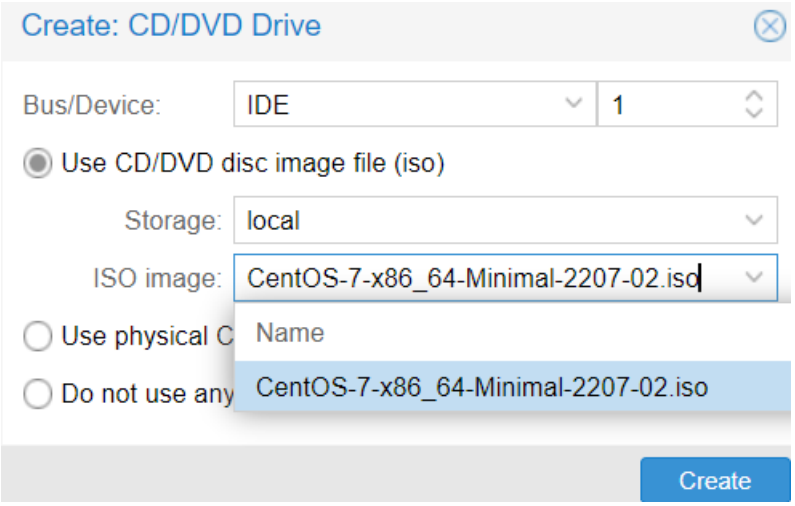
modinfo cdrom



Vincule un fichero ISO como unidad de cdrom de nuestra máquina.

Para este apartado tenemos que irnos al servidor en proxmox. Seleccionamos nuestra máquina virtual, nos vamos a la parte de hardware y le damos a Add como se ve en la imagen.





Pruebe que funciona dicho módulo ejecutando el comando mount /dev/cdrom /mnt y comprobando que el contenido cdrom lo podemos consultar accediendo al directorio.

Como dice el enunciado primero ponemos el comando y luego accedemos con ls. Como podemos ver en la foto, nos ha dejado acceder al directorio por lo que lo hemos montado bien

```
[root@localhost ~]# mount /dev/cdrom /mnt
mount: /dev/sr0 is write-protected, mounting read-only
[root@localhost ~]# ls /mnt
CentOS_BuildTag  EULA  images    LiveOS    repodata              RPM-GPG-KEY-CentOS-Testing-7
EFI              GPL   isolinux  Packages  RPM-GPG-KEY-CentOS-7  TRANS.TBL
[root@localhost ~]#
```

Descargue el módulo que da acceso a la unidad de cdrom.

Para este paso primero deberemos desmontar el cdrom para que deje de estar en uso los módulos.

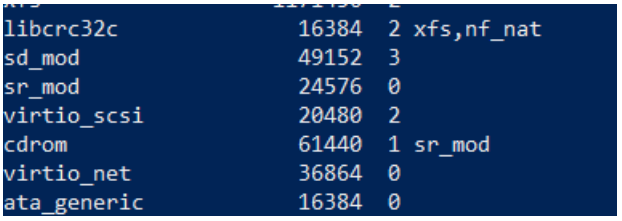
umount /mnt

Para posteriormente desinstalar los módulos intalados y volver a instalarlos después.

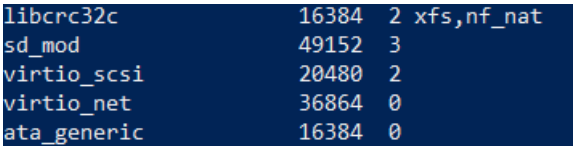
1º **rmmod sr_mod** (Porque es el modulo que usa cdrom)
2º **rmmod cdrom**

Tras hacer usar estos dos comandos tendremos desisntalados los módulos que nos daban acceso a los montajes del cdrom.

Aquí podemos ver cómo están instalados **antes** de usar los comandos:



Y aquí podremos ver como no están una vez usamos los comandos:



Para obtener ambas imágenes usamos **lsmod** como hicimos al principio de este apartado.

Ahora cuando intentamos montar otra vez el cdrom nos dira que no se ha podido, por lo que deberemos volver a instalar dichos módulos

Para esto seguiremos estos dos sencillos pasos:

1º **modprobe cdrom**
2º **modprobe sr_rom**

Ahora si nos dejara montar nuestro cdrom y acceder a él. Una vez hechos estos pasos nos aparecerán los módulos listados al usar un **lsmod** en las primeras líneas de los módulos en vez de en su posición anterior.

1.13 Compilación de módulos de terceros (optativa)

Compile este módulo haciendo uso de la utilidad DKMS

Para empezar, deberemos descargar o clonar el código fuente desde el repositorio de github

git clone https://github.com/umlaeute/v412loopback.git

Después accedemos al repositorio local de nuestra VM

cd v412loopback

Para proseguir con nuestra compilación necesitaremos instalar la utilidad DKMS

1º **yum install --enablerepo=extras epel-release**

2º **yum remove ipa-common ipa-common-client ipa-client**

3º **yum install kernel-debug-devel dkms**

Una vez tengamos la utilidad instalada deberemos hacer uso del **make** nuevamente para la instalación

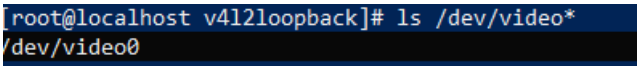
make dkms

Esto nos permitirá que se instale el módulo en el directorio de DKMS para que se compile e instale automáticamente cada vez que se actualice el kernel.

Después procederemos a cargar el módulo:

modprobe v412loopback

Depués ejecutaremos un **ls** para saber si se ha instalado y deberemos ver una dependencia de video



Seguido deberíamos hacer uso de los ejemplos de prueba que vienen en el repositorio que clonamos, pero al probarlos se queda colgada la VM y no he conseguido avanzar más.

1.14 Conclusión

Como conclusión lo que he podido sacar en claro desde primera hora, es que debes saber manejarte con Linux para que este tipo de practica se te haga medianamente llevadera por que debes descargar bastantes utilidades y repositorios.

Una vez que te adaptas a esas pequeñas cosas ya la cosa empieza a fluir un poco mejor, pero es entonces cuando empiezas a tener los odiosos problemas de compilado. En mi caso cuando quise instalar un kernel actual tuve que crear 4 maquinas distintas haciendo pruebas, hasta que di con el make menuconfig que fue el que me pudo solucionar los problemas. Una vez que di con esa utilidad todo fue mucho más fluido y empecé a comprender más de lo que hacía.

En general me parece una practica que hay que dedicarle bastantes horas si no sabes lo que estas haciendo, pero que una vez que consigues cogerle el hilo resulta interesante y se acortan bastante los tiempos en el proceso.