

HowTo ASIX Virtualization and installations

Curs 2021-2022

Virtualization	5
Conceptes clau	5
Instal·lació de les eines de virtualització Kvm/Qemu/Libvirt	6
Requeriments	6
Instal·lació del software	6
Configurar libvirtd per ser executat com un usuari no privilegiat	6
Laboratori de pràctiques	9
Per instal·lar via HTTP	9
Per instal·lar usant netinst	9
Per instal·lar usant un DVD complet	9
Imatges Cloud	10
Altres imatges	10
Live Imatges	11
Tricks: users & passwords	11
Eines virt (1)	13
Virt-manager	13
Crear una nova màquina virtual:	14
Opcions de creació d'una nova VM	14
Passos de creació de la VM:	14
Eliminar una màquina virtual	17
Configuració hardware d'una VM	18
Pràctica-1 Virt-manager	18
Virt-install	20
Descripció	20
Opcions de configuració	20
Exemples del man	22
Exemples d'exercicis antics	23
Exemples de funcionament	24
Pràctica-2 Virt-install	28
Virt-shell	29
Quadre d'ordres principals	29
Funcionalitat general bàsica	30
Obtenir informació general del domini	32

Afegir/eliminar elements a una VM (attach/detach)	34
Snapshots d'una VM	37
Pràctica-3 Virsh	39
qemu	39
qemu / qemu-system-x86_64	40
qemu-img	40
Pràctica-4 qemu / qemu-img	42
Configuracions de xarxa	44
Default network (NAT)	45
Virtual Network: Default	45
Característiques de la xarxa default	45
Descripció de les interfícies de xarxa en els guests i el host	47
Afegir una nova interfície de xarxa	48
Create a virtual private network	51
Característiques en la creació d'una nova xarxa virtual	51
Procés de creació d'una xarxa privada virtual net-hisix	51
Private	53
Isolated	54
Bridge	54
Crear / Configurar bridge en Debian	55
Crear / Configurar bridge en Fedora	56
Gestió d'imatges de disc i pools d'emmagatzemament <pendent>	58
virt-convert	58
virt format	58
virt-make-fs	58
Virt-resize	58
virt-sparsify	58
qemu-img	58
virsh vol-[command]	58
virsh pool-[command]	58
guestfishfs	58
Eines virt (2)	59
Utilitats generals virt-[command]	59
Crear / Clonar / Preparar màquines virtuals <pendent>	66
virt-builder	66
virt-clone	67
virt-sysprep	68
virt-customize	68
Clients d'escriptoris remots <pendent>	70
virt-viewer	70
spice	70
vncviewer	70
remmina	70

Descripció XML de les VM <pendent>	71
virt-xml	71
virt-xml-validate	71
virsh [command]	71
Instal·lacions	72
Fedora-32	73
Layout per fer particions:	73
Instal·lació Fedora-32	73
Opcions de particionat de Fedora-32: automàtic	74
Opcions de particionat de Fedora-32: Custom	75
Exemple amb LVM + Create them automatically	75
Observar les opcions de configuració manual	77
Exemple amb Standard Partition + Create them automatically	77
Exemple de creació de particions manualment (gràfic)	78
Exemple de creació de particions Btrfs	79
Exemple d'espai insuficient per a la instal·lació	80
Debian-11	80
Ubuntu 20 <pendent>	120

Virt-Manager	kvm / qemu / libvirt
Vagrant	
Packer	
Ansible	
Cloud-init	

Docker	
Kubernetes	
Cloud Providers: AWS / Google	

- ☐ <https://wiki.debian.org/QEMU>
 - ☐ <https://wiki.debian.org/QEMU>
 - ☐ <https://www.linux-kvm.org/page/HOWTO>
 - ☐ <http://wiki.virtualsquare.org/#!tutorials/vdebasics.md>
 - ☐ <https://wiki.debian.org/libvirt>
 - ☐ <https://libvirt.org/manpages/virsh.html>
 - ☐ https://cloud-init.io/?_ga=2.115595270.1671114692.1642099236-687519757.1617810142
 - ☐ <https://cloudinit.readthedocs.io/en/latest/>
 - ☐ Ansible: <https://elpuig.xeill.net/Members/vcarceler/asix-m06/a7>
-

Virtualization

Conceptes clau

- Para-virtualization.
- Full virtualization.
- Hypervisor
- Host / Guest

- Kvm
- Qemu
- Xen
- libvirt

Podeu consultar entre altres les següents fonts d'informació per conèixer el significat d'aquests conceptes clau:

- ☐ Wiki [Virtualization](#), [Hypervisor](#), [Kvm](#), [Qemu](#), [libvirt](#) ans [Xen](#).
- ☐ Publicació: Teoria_Virtualizacio_Contenedores.pdf, José Antonio Carrasco Díaz, I.E.S. Foco. Romero Vargas.

Conceptes clau:

- host: l'equip on està treballant l'usuari (si hi ha usuari), és a dir, el de l'alumne. En el host és on executem les ordres de virtualització i dins d'ell engegarem una a més màquines virtuals.
- guest: és la màquina virtual. El guest és la finestra (si es visualitza gràficament) on simulem tenir un ordinador executant un sistema operatiu.
- Tecla-sortida: usualment cal prémer ctr+alt per sortir de la finestra del guest i poder tornar a la del host. Una altra combinació usual és el control de la dreta.

Tipus d'imatges:

- Imatges iso.
- Imatges en pens USB.
- Imatges en particions físiques.
- Imatges .img fetes amb dd d'un device. De fet es poden muntar al loop i es tractaria del mateix que el cas anterior.
- Imatges residents en fitxers de imatges de disc tipus qcow2, ova, etc

Instal·lació de les eines de virtualització Kvm/Qemu/Libvirt

Requeriments

Verificar que la CPU té les extensions necessàries per fer la virtualització:

- Intel processor with the Intel VT and the Intel 64 extensions.
- AMD processor with the AMD-V and the AMD64 extensions.
- [svm](#) / [vmx](#)

```
$ grep -E "svm|vmx" /proc/cpuinfo  
$ lscpu | grep -E "svm|vmx"
```

Instal·lació del software

1) Instal·lar el software de:

- @Virtualization
- libvirt
- qemu

Verificar que es disposa de virt-manager, qemu, libvirt

```
[fedora]$ rpm -qa | grep -E "qemu|libvirt|virt"  
[debian]$ dpkg -l | grep -E "qemu|virt"
```

```
$ sudo apt install qemu-system libvirt-daemon-system libvirt-clients
```

2) Activar el servei libvirt permanentment

```
$ sudo systemctl enable libvirt  
$ sudo systemctl start libvirt  
$ sudo systemctl is-enabled libvirt  
$ sudo is-active libvirt  
$ sudo status libvirt
```

Configurar libvirt per ser executat com un usuari no privilegiat

Per tal de poder executar les eines de virtualització amb un usuari no privilegiat es configura libvirt fent:

- Cal assignar l'usuari al grup libvirt (creat en instal·lar el paquet)

- Modificar la configuració de libvirt per permetre l'accés a través dels permisos de grup (/etc/libvirt/libvirtd.conf).
 - Cal descomentar la línia de la directiva: `unix_sock_group = "libvirt"`
 - Verificar la directiva: `unix_sock_rw_perms = "0770"`

Verificar l'existència del grup:

```
$ grep "libvirt" /etc/group
libvirt:x:984:

$ getent group libvirt
```

Assignar l'usuari al grup

```
$ sudo systemctl stop libvirtd

$ usermod -a -G libvirt $(whoami)
```

Modificar el fitxer de configuració `/etc/libvirt/libvirtd.conf`

```
unix_sock_group = "libvirt"

unix_sock_rw_perms = "0770"
```

Reiniciar el servei, pot ser que faci falta tancar la sessió gràfica i iniciar-la de nou per tal de que la pertinença al grup per part de l'usuari sigui efectiva:

```
$ systemctl start libvirtd

$ systemctl status libvirtd
```

Ara ja es pot iniciar virt-manager com un usuari no privilegiat:

```
$ virt-manager
```

Repàs de la configuració:

```
$ uname -a
Linux d02 5.10.0-9-amd64 #1 SMP Debian 5.10.70-1 (2021-09-30) x86_64 GNU/Linux

$ cat /etc/os-release
PRETTY_NAME="Debian GNU/Linux 11 (bullseye)"
NAME="Debian GNU/Linux"
VERSION_ID="11"
VERSION="11 (bullseye)"
VERSION_CODENAME=bullseye
ID=debian
HOME_URL="https://www.debian.org/"
SUPPORT_URL="https://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"
```

```
$ dpkg -l virt*
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Inst/Conf-files/Unpacked/halF-conf/Half-inst/trig-aWait/Trig-pend
|/ Err?=(none)/Reinst-required (Status,Err: uppercase=bad)
||/ Name                               Version                               Architecture
Description
+++-=====
```

ii	virt-manager	1:3.2.0-3	amd64	desktop application for managing virtual machines
ii	virt-viewer	7.0-2	amd64	Displaying the graphical console of a virtual machine
ii	virtinst	1:3.2.0-3	amd64	utilities to create and edit virtual machines

```
$ dpkg -l | grep virt
```

```
ii gir1.2-libvirt-glib-1.0:amd64 3.0.0-1 amd64 GObject introspection files for the libvirt-glib library
ii gvfs:amd64 1.46.2-1 amd64 userspace virtual filesystem - GIO module
ii gvfs-backends 1.46.2-1 amd64 userspace virtual filesystem - backends
ii gvfs-common 1.46.2-1 all userspace virtual filesystem - common data files
ii gvfs-daemons 1.46.2-1 amd64 userspace virtual filesystem - servers
ii gvfs-fuse 1.46.2-1 amd64 userspace virtual filesystem - fuse server
ii gvfs-libs:amd64 1.46.2-1 amd64 userspace virtual filesystem - private libraries
ii libgovirt-common 0.3.7-2 all GObject-based library to access oVirt REST API (common files)
ii libgovirt2:amd64 0.3.7-2 amd64 GObject-based library to access oVirt REST API
ii libvirglrenderer1:amd64 0.8.2-5 amd64 Virtual GPU for KVM virtualization
ii libvirt-clients 7.0.0-3 amd64 Programs for the libvirt library
ii libvirt-daemon 7.0.0-3 amd64 Virtualization daemon
ii libvirt-daemon-config-network 7.0.0-3 all Libvirt daemon configuration files (default network)
ii libvirt-daemon-config-nwfilter 7.0.0-3 all Libvirt daemon configuration files (default network filters)
ii libvirt-daemon-driver-lxc 7.0.0-3 amd64 Virtualization daemon LXC connection driver
ii libvirt-daemon-driver-qemu 7.0.0-3 amd64 Virtualization daemon QEMU connection driver
ii libvirt-daemon-driver-vbox 7.0.0-3 amd64 Virtualization daemon VirtualBox connection driver
ii libvirt-daemon-driver-xen 7.0.0-3 amd64 Virtualization daemon Xen connection driver
ii libvirt-daemon-system 7.0.0-3 amd64 Libvirt daemon configuration files
ii libvirt-daemon-system-systemd 7.0.0-3 all Libvirt daemon configuration files (systemd)
ii libvirt-glib-1.0-0:amd64 3.0.0-1 amd64 libvirt GLib and GObject mapping library
ii libvirt0:amd64 7.0.0-3 amd64 library for interfacing with different virtualization systems
ii ovmf 2020.11-2 all UEFI firmware for 64-bit x86 virtual machines
ii python3-libvirt 7.0.0-2 amd64 libvirt Python 3 bindings
ii virt-manager 1:3.2.0-3 all desktop application for managing virtual machines
ii virt-viewer 7.0-2 amd64 Displaying the graphical console of a virtual machine
ii virtinst 1:3.2.0-3 all utilities to create and edit virtual machines
```

```
$ dpkg -l | grep qemu
```

```
ii ipxe-qemu 1.0.0+git-20190125.36a4c85-5.1 all PXE boot firmware - ROM images for qemu
ii libvirt-daemon-driver-qemu 7.0.0-3 amd64 Virtualization daemon QEMU connection driver
ii qemu 1:5.2+dfsg-11+deb11u1 amd64 fast processor emulator, dummy package
ii qemu-system-common 1:5.2+dfsg-11+deb11u1 amd64 QEMU full system emulation binaries (common files)
ii qemu-system-data 1:5.2+dfsg-11+deb11u1 all QEMU full system emulation (data files)
ii qemu-system-gui:amd64 1:5.2+dfsg-11+deb11u1 amd64 QEMU full system emulation binaries (user interface and audio support)
ii qemu-system-x86 1:5.2+dfsg-11+deb11u1 amd64 QEMU full system emulation binaries (x86)
ii qemu-utils 1:5.2+dfsg-11+deb11u1 amd64 QEMU utilities
```

Laboratori de pràctiques

Per poder realitzar les pràctiques i exemples convé disposar localment i a través de xarxa d'imatges d'instal·lació, màquines virtuals del Clou i altres màquines.

Per instal·lar via HTTP

- ☐ Fedora32
https://dl.fedoraproject.org/pub/fedora/linux/releases/32/Everything/x86_64/os/
- ☐ Debian11
<http://ftp.us.debian.org/debian/dists/Debian11.2/main/installer-amd64/>
- ☐ Ubuntu 18 Bionic
<http://archive.ubuntu.com/ubuntu/dists/bionic/main/installer-amd64/>

Per instal·lar usant netinst

Es tracta d'imatges ISO d'instal·lació dels sistemes operatius que no contenen tot el software d'instal·lació i per tant caben en un CD ocupant al voltant de 500MB. El procediment d'instal·lació va a buscar el software als repositoris d'internet.

- ☐ Fedora 32 netinst 579M
https://archives.fedoraproject.org/pub/archive/fedora/linux/releases/30/Workstation/x86_64/iso/Fedora-Workstation-netinst-x86_64-30-1.2.iso
- ☐ Debian 11 Bullseye netinst amd64 396M
<https://cdimage.debian.org/debian-cd/current/amd64/iso-cd/debian-11.1.0-amd64-netinst.iso>
- ☐ Centos 8.4 boot 723M
http://mirror.arenanetworks.es/CentOS/8.4.2105/isos/x86_64/CentOS-8.4.2105-x86_64-boot.iso

Per instal·lar usant un DVD complert

Algunes versions de les distribucions encara es lliuren en format DVD complert que conté tot el software a instal·lar. Es tracta, però, del software en el moment de generar el DVD i

per tant queda desfasat respecte les actualitzacions. Un cop instal·lat el sistema és recomanable fer l'actualització (per això és més pràctic usar les instal·lacions netinst).

- ☐ Ubuntu 20

<https://ubuntu.com/download/desktop/thank-you?version=20.04.3&architecture=amd64>

Imatges Cloud

Màquines virtuals fetes per les pròpies distribucions, són petites i lleugeres per poder-les posar en funcionament directament. Poden proporcionar-se en varis formats com per exemple: qcow2 (qemu), raw (en cru), vagrant, per a vmware, per a virtualbox, etc.

- ☐ Fedora Cloud 32 Repo: /pub/fedora/linux/releases/32/Cloud/x86_64/images
https://dl.fedoraproject.org/pub/fedora/linux/releases/32/Cloud/x86_64/images/Fedora-Cloud-Base-32-1.6.x86_64.qcow2

Fedora images have no root/user access. Image should be modified by virt-sysprep to set the root password: virt-sysprep belongs to the libguestfs-tools package:

```
$ sudo virt-sysprep -a Fedora-name-image.qcow2 --root-password password:newpasswd
```

- ☐ Debian 11 Bullseye nocloud 297M
<https://cloud.debian.org/images/cloud/bullseye/latest/debian-11-nocloud-amd64.qcow2>
- ☐ Ubuntu Cloud Images 20.10 (usb image .IMG 549M)
<https://cloud-images.ubuntu.com/releases/groovy/release/ubuntu-20.10-server-cloudimg-amd64.img>

Altres imatges

- ☐ Alpine
<https://alpinelinux.org/downloads/>
- ☐ DSL Damn Small Linux
Wiki en https://en.wikipedia.org/wiki/Damn_Small_Linux
http://distro.ibiblio.org/damnsmall/release_candidate/dsl-4.11.rc1.iso
- ☐ Gparted
<https://downloads.sourceforge.net/gparted/gparted-live-1.3.1-1-amd64.iso>
- ☐ Super Grub Disk

https://sourceforge.net/projects/supergrub2/files/2.04s1/super_grub2_disk_2.04s1/super_grub2_disk_hybrid_2.04s1.iso/download

Live Images

- ☐ Fedora 30 Live Workstation

https://archives.fedoraproject.org/pub/archive/fedora/linux/releases/30/Workstation/x86_64/iso/Fedora-Workstation-Live-x86_64-30-1.2.iso

- ☐ Debian Live Images

<https://www.debian.org/CD/live/>
<https://cdimage.debian.org/debian-cd/current-live/amd64/iso-hybrid/debian-live-11.2.0-amd64-standard.iso>

Tricks: users & passwords

Debian:

- ☐ Debian nocloud image qcow2: User **root**, no password.
- ☐ Debian Live: **user / live**
- ☐ Debian AWS EC2:

Fedora:

- ☐ Fedora Cloud images: usar virt-sysprep per assignar password.
- ☐ Fedora Live:
- ☐ Fedora AWS EC2

Ubuntu:

- ☐ Ubuntu Cloud images:
- ☐ Ubuntu Live:
- ☐ Ubuntu AWS EC2

AWS EC2 images

- ☐ Debian:
- ☐ Fedora:
- ☐ Centos:
- ☐ AWS AMI2: **ec2-user**
- ☐ Windows:

Centos

- ☐
- ☐

Windows

- ☐
- ☐

Eines virt (1)

- ☐ Virt-manager
- ☐ Virt-install
- ☐ Virt-shell
- ☐ qemu-xxxx
- ☐ Virt-viewer

virt-alignment-scan	virt-df	virt-index-validate	virt-make-fs
virt-gemu-run	virt-tar-in		
virt-builder	virt-diff	virt-inspector	virt-manager
virt-rescue	virt-tar-out		
virt-builder-repository	virt-edit	virt-install	virtnetworkd
virt-resize	virtualbox		
virt-cat	virt-filesystems	virtinterfaced	virtnodedevd
virtsecret	virtualboxvm		
virt-clone	virt-format	virtlockd	virtnwfilterd
virt-sparsify	virt-viewer		
virt-copy-in	virtfs-proxy-helper	virt-log	
virt-pki-validate	virtstoraged	virt-xml	
virt-copy-out	virt-get-kernel	virtlogd	virtproxyd
virt-sysprep	virt-xml-validate		
virt-customize	virt-host-validate	virt-ls	virtqemud
virt-tail			

qemu-edid	qemu-img	qemu-keymap	qemu-nbd
qemu-system-i386	qemu-trace-stap	qemu-ga	qemu-io
qemu-kvm	qemu-pr-helper	qemu-system-x86_64	

Virt-manager

- ☐ Creació d'una nova màquina virtual
- ☐ Eliminar una màquina virtual

☐ Configuració hardware

Crear una nova màquina virtual:

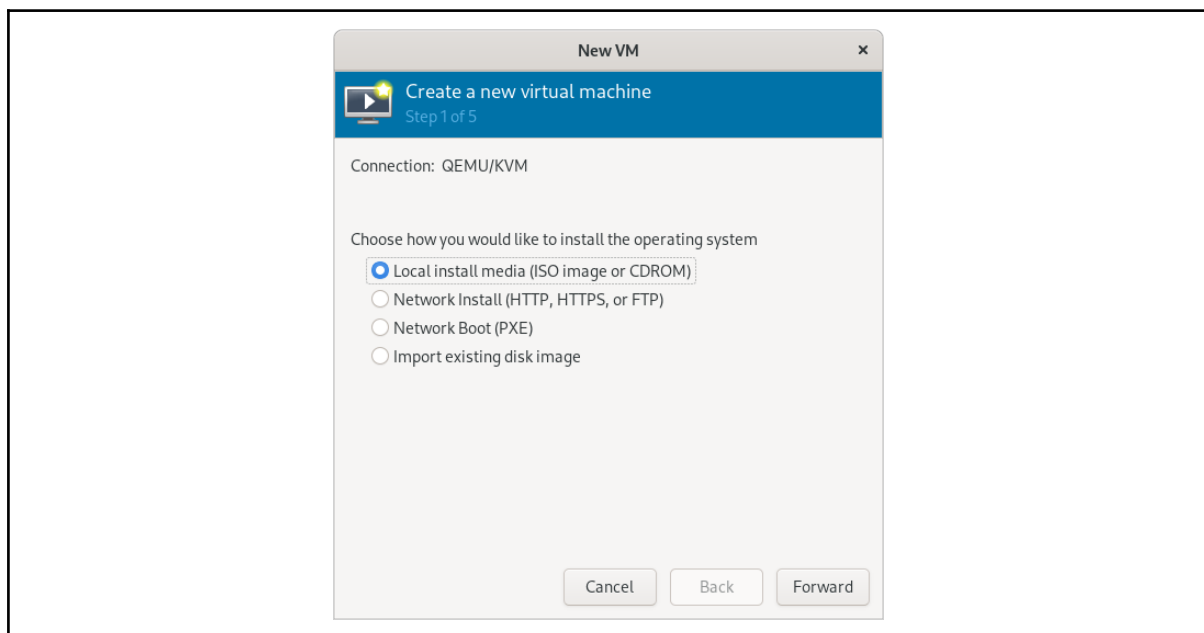
Opcions de creació d'una nova VM

Opcions (1)

- Instal·lar d'un CDROM
- Instal·lar de xarxa via HTTP/HTTPS/FTP
- Per xarxa via PXE
- Importar una imatge de disk existent

Opcions(2)

- Instal·lar d'un CDROM
- Instal·lar de xarxa via HTTP/HTTPS/FTP
- Importar una imatge de disk existent
- Manual Install

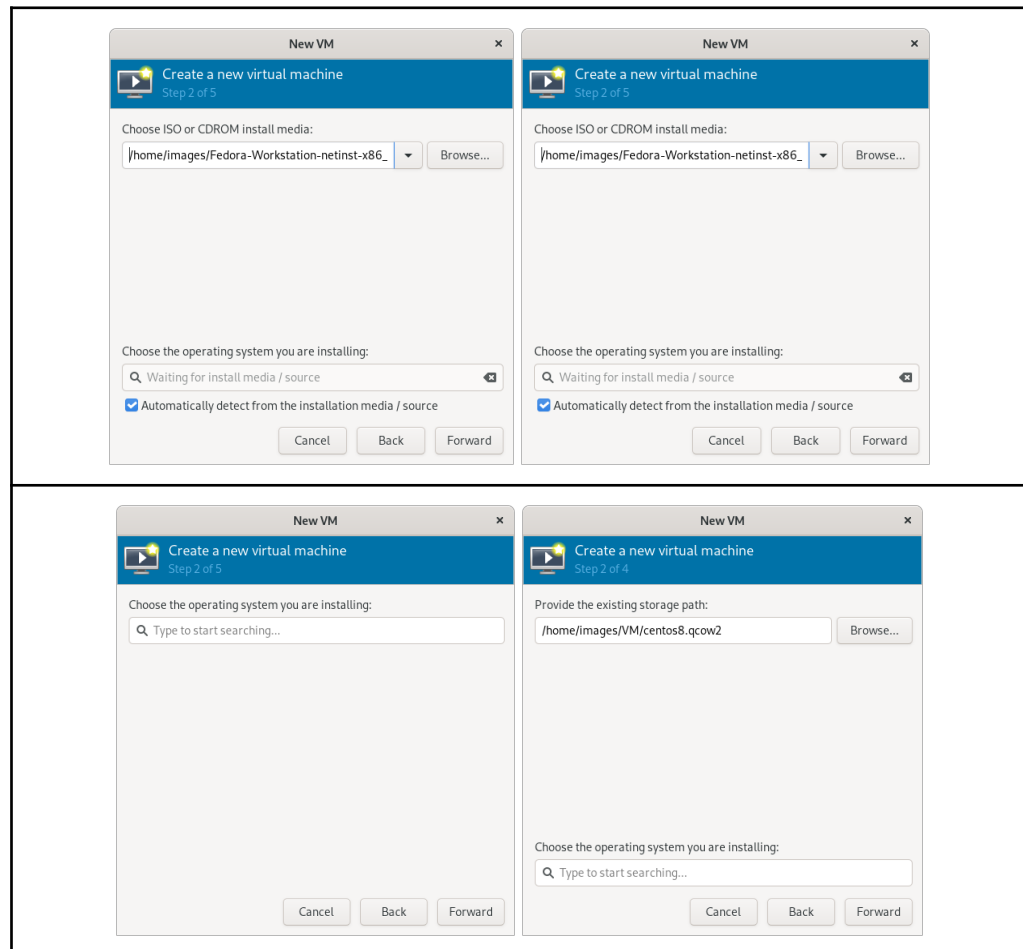


Passos de creació de la VM:

1. New VM

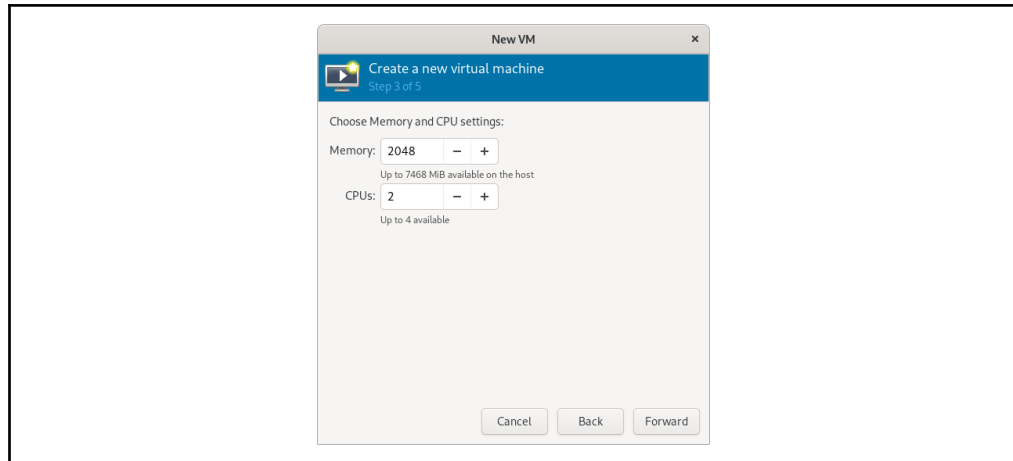
Segons el mètode d'instal·lació seleccionat cal indicar la iso a utilitzar o la URL, etc. Si no es detecta automàticament el tipus de sistema operatiu es pot indicar manualment seleccionant-ne un del desplegable.

- Si s'ha escollit instal·lar via ISO virt-manager lliga automàticament la ISO com a dispositiu CD/DVD únicament per a la primera arracada.
- <pendent> Network boot
- Si s'utilitza local install caldrà que sigui l'usuari qui configuri el mecanisme d'instal·lació, si és usant un CD/DVD caldrà que en fer el reboot pensi a modificar el Boot Order, també si ha triat instal·lar via PXE.



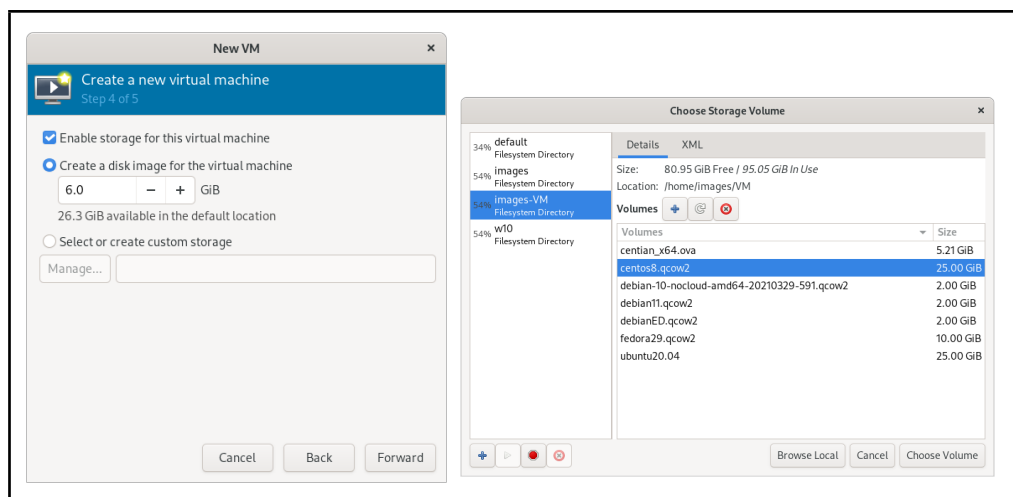
2. Memory and CPU settings

- Determinar la quantitat de memòria que tindrà la VM i quantes CPUs. No usar totes les disponibles o la màquina host es quedarà sense recursos.



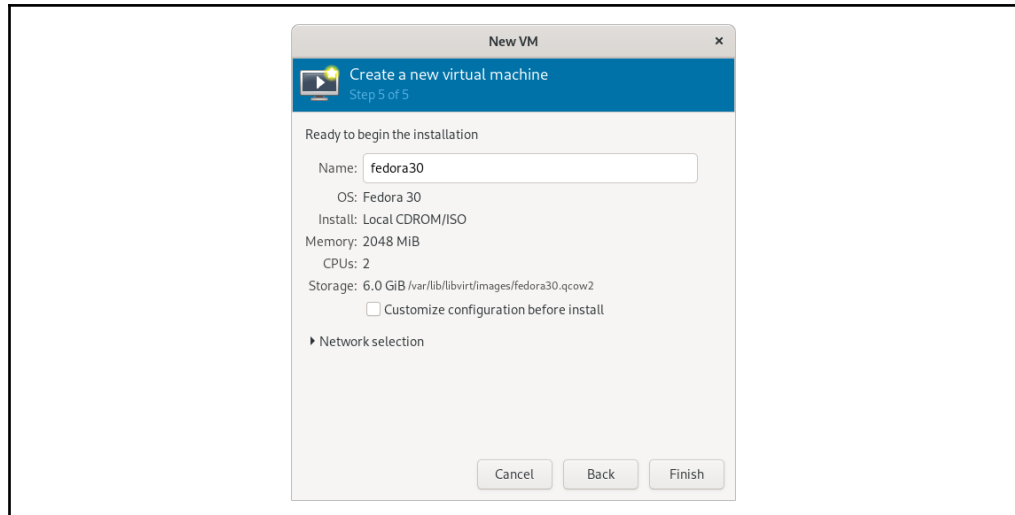
3. Storage

- Determinar si cal o no emmagatzemament per a la VM. Per exemple si es tracta d'una màquina que executa un sistema Live no requereix de HD.
- Si cal crear un (o més) HD d'emmagatzemament es pot:
 - Crear-ne un de nou indicant la mida en GiB.
 - Seleccionar una imatge o partició d'emmagatzemament ja existent.
- **qcow2**: aquest és el format per defecte que s'utilitza, un format Copy On Write (per capes) dinàmic (creix a mida que es necessita).
- **Atenció**: observar clarament on es genera el fitxer de la imatge (qcow2) per eliminar-lo quan s'elimini la VM, és important fer neteja o ràpidament el disc queda ple!. Usualment: `/var/lib/libvirt/images/nom.qcow2`.



4. Ready to begin

- Un cop seleccionades totes les opcions clicant a Finish es genera la VM.
- Observar la ruta on es genera el disc de emmagatzemament: `/var/lib/libvirt/images/nom.qcow2`.
- Observar l'opció de *"Custom configuration before install"* que permet configurar el hardware de la VM abans de crear-la / engegar-la.

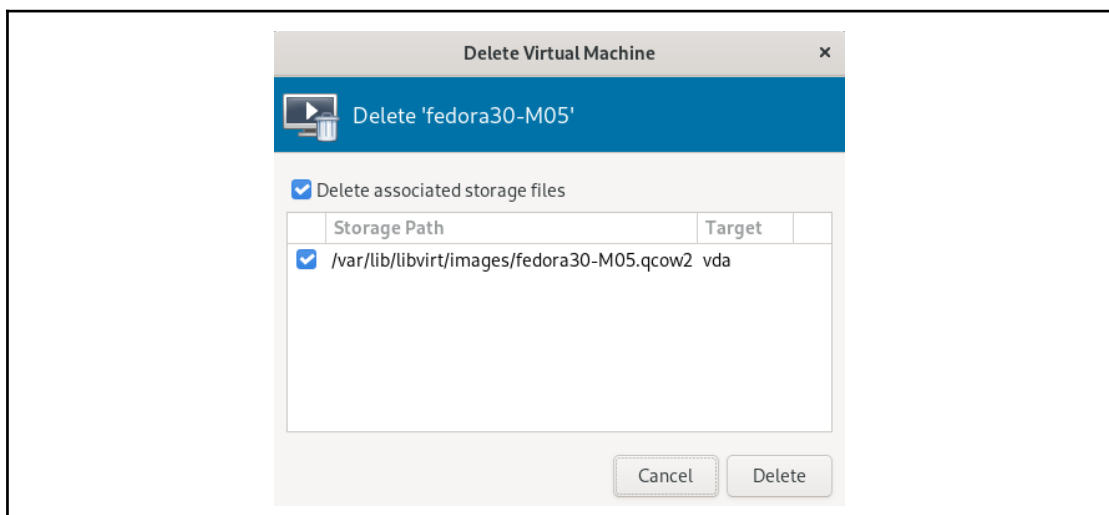


5. Executar la màquina virtual, engegar-la

- Un cop feta la creació s'engega automàticament.
- A vegades pot sol·licitar si es vol activar la xarxa virtual indicant que actualment està desactivada (yes).
- Si la màquina està parada es pot engegar amb el botó de play.
- Es pot configurar el hardware de la màquina amb la màquina parada prement el botó de configuració.

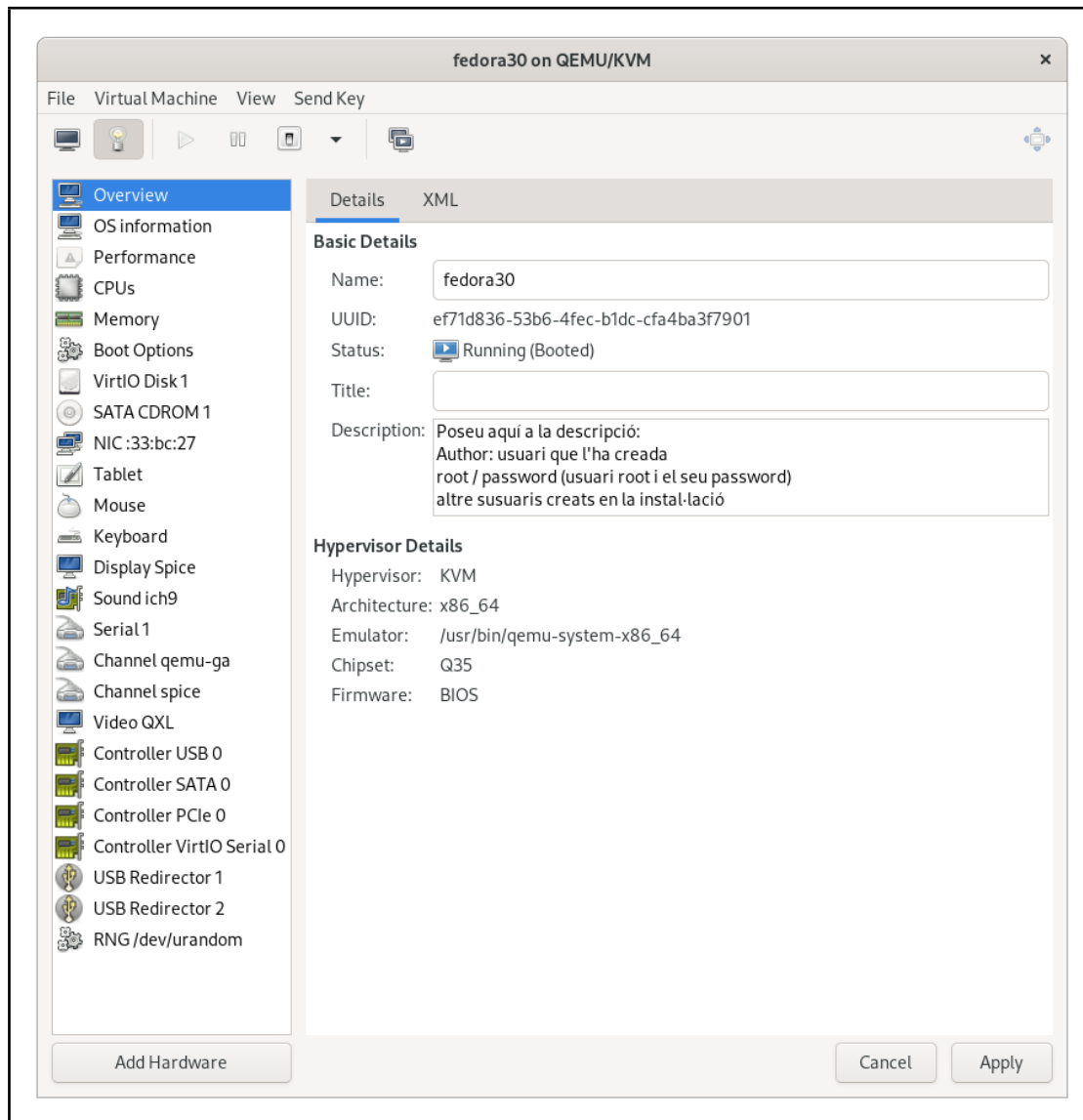
Eliminar una màquina virtual

- Per eliminar una màquina virtual simplement seleccionar-la del llistat de màquines i escollir Delete.
- **Atenció:** decidir si es vol eliminar el storage o no. Per defecte a l'aula es recomana que si sinó es van deixant discs virtuals que omplen el disc dir i impedeixen de funcionar correctament el sistema operatiu.



Configuració hardware d'una VM

Es pot modificar la configuració hardware d'una VM simulant la modificació que es faria en un ordinador si se li treuen, afageixen o modifiquen components, com per exemple targetes de xarxa, unitats de disc, de CD/DVD, etc.



Pràctica-1 Virt-manager

- 01 Crear una VM amb un disc en blanc i una iso netinst de Debian i realitzar una instal·lació minimal.

- 02 Crear una VM amb un disc en blanc i instal·lar Fedora32 usant una url de HTTP com a mecanisme d'instal·lació.
- 03 Crear una VM amb un disc en blanc i instal·lar un Ubuntu complet partint de la url HTTP.
- 04 Crear una VM partint d'una imatge ja existent: debian qcow2. Assignar-li memòria, cpu, nom, descripció.
- 05 Crear una VM partint d'una imatge ja existent tipus DSL com per exemple un alpine. Assignar-li una unitat de disc extra.
- 06 Crear una VM que engega una Live i té assignat espai d'emmagatzemament amb un disc extra.
- 07 Crear una VM basada en la imatge Cloud de Fedora32. Modificar-la amb virt-sysprep per assignar-li a root el password habitual.

Virt-install

Descripció

Amb l'eina virt-install es poden **crear màquines virtuals** des de l'entorn de comandes, sense usar l'eina gràfica virt-manager o usant-la. Crear la màquina tant pot voler dir crear una màquina partint d'una imatge ja existent com crear una màquina fent tot el procés d'instal·lació (interactivament o d manera desatesa).

Característiques:

- Permet crear màquines virtuals de manera desatesa, sense menús, des d'una sola comanda de text.
- La màquina virtual pot disposar d'un o més discs virtuals, devices usb i pci del host amfitrió, audio i interfícies de xarxa.
- La instal·lació del sistema operatiu en el guest es pot fer localment via cd, dvd, imatges iso o bé per xarxa via http, ftp, nfs, PXE, etc.
- Si el guest és una instal·lació desatesa (kickstart) permet crear el guest tot de manera desatesa.
- El guest s'executa com un procés en el host. No cal visualitzar-lo
- gràficament per tenir-lo engegat. Existeixen diversos mecanismes per 'visualitzar' una màquina virtual. Es pot fer via:
 - Hipervisors: com qemu i xen.
 - VNC (Virtual Network Computing), un visualitzador d'escriptoris remots.
- Les configuracions XML de les VM que es creen, dels domains, es poden trobar a </etc/libvirt>.

Opcions de configuració

Virt-install té multitud d'opcions de configuració que permeten configurar fins al mínim detall. Es recomana consultar la seva ajuda i els exemples de la pàgina man.

```
$ virt-install --help
```

Exemple d'opcions de configuració:

Opcions generals:

- `--name=NAME` permet posar un nom identificador que ha de ser únic al guest.
- `--ram=MEM` permet indicar la quantitat de memòria en Mbytes del guest.
- `--vcpus=CPUS` permet indicar la quantitat de cpus del guest.
- `--os-type=OS-TYPE` permet indicar el tipus de sistema operatiu del guest.

- `--os-variant=OS-VARIANT` permet indicar el tipus concret de sistema operatiu.
- `--host-device=HOST_DEV` permet adjuntar un device físic del host en el guest.

Opcions referents al mètode d'instal·lació del sistema operatiu en el guest:

- `--cdrom=CDROM` indica que cal usar el device indicat (ex: `/dev/sr0`) o el fitxer indicat com a font de la instal·lació del sistema operatiu en el guest.
- `--location` permet indicar una URL d'on obtenir la font de la instal·lació.
- `--pxe` permet fer la instal·lació via PXE (requereix bridge de xarxa).
- `--import` obliga a saltar-se el pas de la instal·lació i utilitza directament el device o fitxer indicat com a virtual-disk (opcions `--disk` o `--file`) com a sistema operatiu fet.
- `--livecd` indica que el dispositiu d'instal·lació és un livecd.

• Opcions d'emmagatzemament:

- `--disk=DISKOPTIONS` permet descriure tot allò referent a un device de disc virtual que el guest ha de tenir.
 - El virtual-disk pot ser un fitxer o un dispositiu real. També pot ser local o compartit per xarxa.
 - Si el disc ja existeix s'utilitza. Si no existeix es crea de nou. En aquest cas és obligatori indicar la mida.
 - Els fitxers d'imatges virtuals de disc (virtual-disk) poden ser de diferents formats. Els identificats actualment són: raw, qcow2, qcow, vdi, vmsdk, etc.

Opcions de configuració de xarxa:

- `--network=NETWORK,opcions` permet descriure el o els dispositius de xarxa a usar per el guest. És a dir, descriu com serà la xarxa que lliga el guest amb el host i descriu el tipus d'interfície de xarxa virtual del guest.
- `NETWORK` pot prendre un dels tres valors següents que descriuen el 'tipus de xarxa' que s'estableix:
 - `bridge=BRIDGE` indica que el guest usarà un bridge conjuntament amb el host. Això significa que el guest té visibilitat a la xarxa externa del host, té una cara a la interfície de xarxa externa. Només es pot usar aquesta opció si el host té connexió de xarxa amb cable. Aquesta opció permet que els guest siguin servidors de xarxa i que s'hi accedeixi des de l'exterior. Per usar un bridge cal configurar-lo anteriorment en el host.
 - `network=NAME` crea una xarxa virtual de nom NAME. Les xarxes virtuals són xarxes internes entre el host i les màquines virtuals. Vàries màquines virtuals poden pertànyer a la mateixa xarxa interna (usar el mateix NAME).
 - Si no s'indica un nom de xarxa la xarxa usada s'anomena 'default'. El host pot fer NAT per proporcionar sortida a l'exterior dels guest de la xarxa interna. Els guest no són accessibles des de l'exterior. No poden fer de servidors. Cal usar aquesta opció si el host usa NetworkManager o usa wireless.
 - `user` permet usar opcions de xarxa molt limitades, s'utilitza amb qemu quan l'executa un usuari no privilegiat.
- Es poden indicar opcions que permetin descriure el tipus d'interfície de xarxa virtual a usar. ◦ Si el host té un bridge amb alguna interfície física adjunta s'utilitza el bridge. Si no s'utilitza la xarxa virtual 'default'.

Opcions de configuració gràfiques:

- `--vnc` activa en el guest un servidor vnc de manera que es pot visualitzar el guest a través d'una connexió vnc client des de qualsevol equip, sigui el host o un altre.
- `--vnc-port=xx` permet indicar el port per on escolta el servei vnc del guest.
- `--sdl` genera una consola virtual visible en el host. Si es tanca es tanca el guest.

Opcions específiques només per a Full-Virtualization:

- `--sound` permet disposar en el guest del device de so del host.

Tipus de virtualització:

- `--hvm` indica que cal usar Full-Virtualization.
- `--paravirt` indica que cal usar paravirtualització en lloc de full-virtualization.

Indicar el hipervisor a usar: --connect=HIPERVISOR (només si volem usar un diferent del visor per defecte):

- xen
- qemu:///system
- qemu:///session

Exemples del man

Els següents exemples són extrets de la documentació de *man virt-install*.

```
#1
The simplest invocation to interactively install a Fedora 29 KVM VM with recommended
defaults. virt-viewer(1) will be launched to graphically interact with the VM install

# sudo virt-install --install fedora29
```

```
#2
Similar, but use libosinfo's unattended install support, which will perform the fedora29
install automatically without user intervention:

# sudo virt-install --install fedora29 --unattended
```

```
#3
Install a Windows 10 VM, using 40GiB storage in the default location and 4096MiB of ram,
and ensure we are connecting to the system libvirtd instance:

# virt-install \
    --connect qemu:///system \
    --name my-win10-vm \
    --memory 4096 \
    --disk size=40 \
    --os-variant win10 \
    --cdrom /path/to/my/win10.iso
```

```
#4
Install a CentOS 7 KVM from a URL, with recommended device defaults and default required
storage but specifically request VNC graphics instead of the default SPICE, and request
8 virtual CPUs and 8192 MiB of memory:

# virt-install \
    --connect qemu:///system \
    --memory 8192 \
    --vcpus 8 \
    --graphics vnc \
    --os-variant centos7.0 \
    --location http://mirror.centos.org/centos-7/7/os/x86_64/
```

```
#5
Create a VM around an existing debian9 disk image:

# virt-install \
    --import \
    --memory 512 \
    --disk /home/user/VMs/my-debian9.img \
    --os-variant debian9
```

```
#6
Start serial QEMU ARM VM, which requires specifying a manual kernel.

# virt-install \
    --name armtest \
    --memory 1024 \
    --arch armv7l --machine vexpress-a9 \
    --disk /home/user/VMs/myarmdisk.img \
    --boot
    --kernel=/tmp/my-arm-kernel,initrd=/tmp/my-arm-initrd,dtb=/tmp/my-arm-dtb,kernel_args="console=ttyAMA0 rw root=/dev/mmcblk0p3" \
    --graphics none
```

```
#7
Start an SEV launch security VM with 4GB RAM, 4GB+256MiB of hard_limit, with a couple of virtio devices:
Note: The IOMMU flag needs to be turned on with driver.iommu for virtio devices. Usage of --memtune is currently required
because of SEV limitations, refer to libvirt docs for a detailed explanation.
```

```
# virt-install \
    --name foo \
    --memory 4096 \
    --boot uefi \
    --machine q35 \
    --memtune hard_limit=4563402 \
    --disk size=15,target.bus=scsi \
    --import \
    --controller type=scsi,model=virtio-scsi,driver.iommu=on \
    --controller type=virtio-serial,driver.iommu=on \
    --network network=default,model=virtio,driver.iommu=on \
    --rng driver,iommu=on \
    --memballoon driver.iommu=on \
    --launchSecurity sev
```

Exemples d'exercicis antics

```
#1
# Crear una màquina virtual d'un LiveCD de Fedora: imatge .iso
$ virt-install --name=Live01 --ram=512 --nodisks --livedcd --cdrom
/fc9/vms/Fedora-14-i686-Live-Desktop.iso

#2
# Crear una màquina virtual d'un LiveCD de Fedora: imatge d'un device USB
$ virt-install --name=Live03 --ram=512 --import --disk path=/dev/sdb1 &

#3
# Crear una màquina virtual partint d'un virtual-device ja existent
$ virt-install --name=win01 --ram=512 --import --disk path=win2003s.img &
$ virt-install --name=lin01 --ram=512 --import --disk path=minifl3-pxe-vi.img &

#4
# Crear una màquina virtual d'una imatge dd d'una partició
$ virt-install --name=lin02 --ram=512 --import --disk path=mini-linux.fl3.img &

#5
# Crear una imatge virtual a partir d'un device
$ virt-install --name=dev01 --ram=512 --import --disk path=/dev/sda2 &
$ virt-install --name=dev02 --ram=512 --import --disk path=/dev/sda &
```

```
#6
# Crear un virtual-disk com a emmagatzemament per a un d'un Live CD
# virt-install --name=Live01 --ram=512 --livedcd --cdrom Fedora-14-i686-Live-Desktop.iso
--disk path=free.img,size=1 &
```

```
#7
# Crear un virtual-disk nou on fer una instal·lació a partir d'una imatge Live
# virt-install --name=Live01 --ram=512 --livecd --cdrom Fedora-14-i686-Live-Desktop.iso
--disk path=fedora.img,size=2 &

#8
# Crear una màquina virtual amb un virtual-disk en blanc per instal·lar-hi un Fedora via
xarxa.
# virt-install --name Linux01 --ram 512 --disk path=fedora3g.img,size=3 --location
http://download.fedora.redhat.com/pub/fedora/linux/releases/14/Fedora/i386/os/ &

#9
# Crear una màquina virtual usant un virtual-disk ja existent
# virt-install --name Linux02 --ram 512 --disk path=fedora.img --import
```

Exemples de funcionament

```
#1
$ virt-install --name=alpine --ram=2048 --nodisks --livecd
--cdrom=alpine-standard-3.15.0-x86_64.iso

$ virsh dominfo alpine
Id: 1
Nombre: alpine
UUID: 04a655aa-31c0-4fbf-8243-75be095fce23
Tipo de sistema operativo: hvm
Estado: ejecutando
CPU(s): 1
Hora de la CPU: 5,3s
Memoria máxima: 2097152 KiB
Memoria utilizada: 2097152 KiB
Persistente: si
Autoinicio: desactivar
Guardar administrado: no
Modelo de seguridad: none
DOI de seguridad: 0

$ virsh list --all
Id Nombre Estado
-----
1 alpine ejecutando
```

```
#2
$ virt-install --name debian11 --import --memory 2048 --vcpus 2 --os-variant debian10
--disk debian-11-nocloud-amd64-20220121-894.qcow2 &

$ virsh list
Id Nombre Estado
-----
1 alpine ejecutando
2 debian11 ejecutando

$ virsh dominfo debian10
Id: 2
Nombre: debian11
UUID: fdc0562d-0ab1-493a-8a11-8ac5e2d4aaf7
Tipo de sistema operativo: hvm
Estado: ejecutando
CPU(s): 2
Hora de la CPU: 61,3s
Memoria máxima: 2097152 KiB
Memoria utilizada: 2097152 KiB
Persistente: si
Autoinicio: desactivar
Guardar administrado: no
Modelo de seguridad: none
DOI de seguridad: 0
```



```
#3
$ virt-install --name debian-live --memory 2048 --disk size=4 --vcpus 2 --os-type GNU/Linux
--os-variant debian10 --livecd --cdrom debian-live-10.9.0-amd64-gnome.iso &
[3] 7746
$
Starting install...
Allocating 'debian-live.qcow2' 4.0 GB 00:00:00

$ virsh list
  Id    Name           State
  ----  -
  1     alpine         running
  2     debian10       running
  3     debian-live    running

$ virsh domaininfo debian-live
Id:          3
Name:        debian-live
UUID:        dbf39a97-276f-438b-8504-881a12dc758a
OS Type:     hvm
State:       running
CPU(s):      2
CPU time:    3.1s
Max memory:  2097152 KiB
Used memory: 2097152 KiB
Persistent:  yes
Autostart:   disable
Managed save: no
Security model: selinux
Security DOI: 0
Security label: unconfined_u:unconfined_r:svirt_t:s0:c28,c640 (permissive)

# find / -name debian-live.qcow2 -ls 2> /dev/null
  6685027      840 -rw-----  1 ecanet  ecanet   4295884800 Mar  2 17:50
/home/ecanet/.local/share/libvirt/images/debian-live.qcow2

$ ls -lh ~/.local/share/libvirt/images/debian-live.qcow2
-rw----- 1 ecanet ecanet 4.1G Mar  2 17:50
/home/ecanet/.local/share/libvirt/images/debian-live.qcow2

$ tree ~/.local/share/libvirt
/home/ecanet/.local/share/libvirt
├── images
│   └── debian-live.qcow2
```

```
#4
$ virsh shutdown debian-live
Domain debian-live is being shutdown

$ virsh shutdown debian10
Domain debian10 is being shutdown

$ virsh shutdown alpine
Domain alpine is being shutdown

$ virsh list --all
  Id    Name           State
  ----  -
  1     alpine         running
  2     debian10       running
  -     debian-live    shut off

$ virsh destroy alpine
Domain alpine destroyed

$ virsh destroy debian10
Domain debian10 destroyed

$ virsh list --all
```

```

Id      Name      State
-----
-       alpine      shut off
-       debian-live shut off
-       debian10   shut off

$ virsh start alpine
Domain alpine started

$ virsh list
Id      Name      State
-----
4       alpine      running

$ virsh domdisplay alpine
spice://127.0.0.1:5900

$ virt-viewer alpine

$ virsh destroy alpine
Domain alpine destroyed

$ virsh undefine alpine
Domain alpine has been undefined

$ virsh undefine debian10
Domain debian10 has been undefined

$ virsh undefine debian-live
Domain debian-live has been undefined

$ virsh list --all
Id      Name      State
-----

$ ls ~/.local/share/libvirt/images/debian-live.qcow2
/home/ecanet/.local/share/libvirt/images/debian-live.qcow2

$ rm ~/.local/share/libvirt/images/debian-live.qcow2

```

```

#5
virt-install --name fedora-netinst --memory 2048 --vcpus 2 --os-type GNU/Linux --os-variant
fedora30 --disk size=4,path=/var/tmp/fedora.qcow2 --cdrom
Fedora-Workstation-netinst-x86_64-30-1.2.iso
Starting install...
Allocating 'fedora.qcow2' | 4.0 GB  00:00:00

# ls -lh /var/tmp/fedora.qcow2
-rw-----. 1 ecanet ecanet 4.1G Mar  2 18:20 /var/tmp/fedora.qcow2

$ virsh list --all
Id      Name      State
-----
1       fedora-netinst running

$ virsh shutdown fedora-netinst
Domain fedora-netinst is being shutdown

$ virsh list --all
Id      Name      State
-----
-       fedora-netinst shut off

$ virsh undefine fedora-netinst
Domain fedora-netinst has been undefined

```

```

#5
$ virt-install --name gparted --memory 2048 --disk /var/tmp/fedora.qcow2 --vcpus 2
--os-type GNU/Linux --os-variant debian10 --livecd --cdrom gparted-live-1.2.0-1-amd64.iso
--boot cdrom &

```

```
$ virsh destroy gparted
Domain gparted destroyed

$ virsh undefine gparted
Domain gparted has been undefined
```

```
#6
$ virt-install --name fedora-desatesa --memory 2048 --vcpus 2 --os-type GNU/Linux
--os-variant fedora32 --disk /var/tmp/fedora.qcow2 --install fedora32 --unattended &

[1] 11214
Using fedora32 --location
https://download.fedoraproject.org/pub/fedora/linux/releases/32/Server/x86\_64/os

Starting install...
WARNING Using unattended profile 'desktop'
Retrieving file vmlinuz... | 10 MB
00:00:01
Retrieving file initrd.img... | 74 MB
00:00:09

$ virsh list
  Id   Name               State
-----
  1    fedora-desatesa    running

$ virsh destroy fedora-desatesa
Domain fedora-desatesa destroyed

$ virsh undefine fedora-desatesa
Domain fedora-desatesa has been undefined
```

```
#7
$ osinfo-query os

$ osinfo-query os | grep ubuntu

$ virt-install --name ubuntu-http --memory 2048 --vcpus 2 --os-type GNU/Linux --os-variant
ubuntu20.04 --disk /var/tmp/fedora.qcow2 --location
http://archive.ubuntu.com/ubuntu/dists/bionic/main/installer-amd64/ &

[1] 12028

Starting install...
Retrieving file linux... | 7.9 MB
00:00:00
Retrieving file initrd.gz... | 45 MB
00:00:00

$ virsh list
  Id   Name               State
-----
  1    ubuntu-http        running

$ virsh destroy ubuntu-http
Domain ubuntu-http destroyed

$ virsh undefine ubuntu-http
Domain ubuntu-http has been undefined
```

```
#8
$ virt-install --name win10 --memory 2048 --vcpus 2 --os-type windows --os-variant win10
--disk /var/tmp/fedora.qcow2 --cdrom Win10_21H2_EnglishInternational_x64.iso

Starting install...

$ virsh destroy win10
Domain win10 destroyed

$ virsh undefine win10
```

```
Domain win10 has been undefined
```

Pràctica-2 Virt-install

- 01 Crear una VM partint d'una imatge ja existent: debian qcow2. Assignar-li memòria, cpu, nom, descripció.
- 02 Crear una VM partint d'una imatge ja existent: debian o DSL o Alpine. Assignar-li també una unitat de disc extra.
- 03 Crear una VM que engega una Live i té assignat espai d'emmagatzemament amb un disc extra.
- 04 Crear una VM amb un disc en blanc i una iso netinst de Debian i realitzar una instal·lació minimal.
- 04 Crear una VM amb un disc en blanc i realitzar una instal·lació desatesa de Fedora32.
- 06 Generar un USB d'instal·lació de Debian planxant la iso d'instal·lació al USB. Crear una VM amb un disc en blanc que utilitzi el USB real com a font de la instal·lació.
- 07 Crear una VM basada en la imatge Cloud de Fedora32. Modificar-la prèviament amb virt-sysprep per assignar-li a root el password habitual.
- 08 Crear una VM Windows partint de les màquines prefabricades de prova que proporciona el propi Windows.

Virt-shell

Quadre d'ordres principals

```
$ virsh list [--all]
$ virsh dominfo vm01
$ virsh domstate vm01
$ virsh start vm01
$ virsh reboot vm01
$ virsh shutdown vm01
$ virsh destroy vm01
$ virsh undefine vm01
$ virsh autostart
```

```
$ virsh suspend vm01
$ virsh resume vm01
$ virsh screenshot
$ virsh vcpuinfo
$ virsh vncdisplay
$ virsh domdisplay
```

```
$ virsh create xml.file
$ virsh define xml.file
$ virsh undefine vm01
$ virsh dumpxml xml.file
$ virsh save vm01
$ virsh restore vm01
```

```
$ virsh attach-device
$ virsh attach-disk
$ virsh attach-interface
$ virsh net-info
$ virsh net-list
$ virsh iface-list
$ virsh iface-bridge
```

```
$ virsh snapshot create
$ virsh snapshot list
$ virsh snapshot-revert
$ virsh snapshot-delete
```

Funcionalitat general bàsica

Amb virsh es pot governar el funcionament de les VM, la seva configuració els snapshots i molts altres aspectes. En especial permet també editar fàcilment el XML de la definició de les VM.

Les VM no tenen perquè visualitzar-se, poden estar engegades per funcionar com a servidors *'headless'*. Es pot accedir a la consola / escriptori amb eines d'accés remot com *spice* o *vncviewer* a través de les utilitats *virt-viewer* i la informació de *virtsh domdisplay*.

Les configuracions XML de les VM que es creen, dels domains, es poden trobar a [/etc/libvirt](#).

- ☐ [14. Managing guest virtual machines with virsh](#) (Red Hat Documentation)
- ☐ Computing for geeks [virsh sheet](#)

list [--all]

listar

start

engegar una VM que estava aturada.

reboot

Fer un reboot de la VM.

shutdown

aturar la VM (no sempre es deixa).

destroy

aturar la VM obligatòriament. La VM però continua existint i no se'n pot crear una de nova amb el mateix nom. Amb start es pot tornar engegar. És equivalent a fer stop de la màquina.

undefine

eliminar definitivament la VM, eliminant la seva configuració (el XML). No s'eliminen els seus dispositius de disc.

autostart

engega la VM automàticament en iniciar-se el servei del libvirtd, en general en engegar el host amfitrió.

```
#1
$ virt-install --name=alpine --ram=2048 --nodisks --livecd
--cdrom=alpine-standard-3.15.0-x86_64.iso
```

```

$ virsh dominfo alpine
Id: 1
Nombre: alpine
UUID: 04a655aa-31c0-4fbf-8243-75be095fce23
Tipo de sistema operativo: hvm
Estado: ejecutando
CPU(s): 1
Hora de la CPU: 5,3s
Memoria máxima: 2097152 KiB
Memoria utilizada: 2097152 KiB
Persistente: si
Autoinicio: desactivar
Guardar administrado: no
Modelo de seguridad: none
DOI de seguridad: 0

$ virsh list --all
  Id   Nombre   Estado
-----
  1     alpine   ejecutando

$ virsh destroy alpine
Domain alpine destroyed

$ virsh list
  Id   Name   State
-----

$ virsh dominfo alpine
Id: -
Name: alpine
UUID: 5e2843f3-2a59-4c7b-a10e-cc07a25f869b
OS Type: hvm
State: shut off
CPU(s): 1
Max memory: 2097152 KiB
Used memory: 2097152 KiB
Persistent: yes
Autostart: disable
Managed save: no
Security model: selinux
Security DOI: 0

```

Per visualitzar VM que estan engegades s'utilitzen clients d'escriptori remot i ordres com:

virt-viewer

obre un client d'escriptori remot per accedir a la consola / escriptori de la VM. Tancar el virt-viewer no tanca la màquina, només el visar (quan tanques la tele no deixen de fer les notícies!).

virsh domdisplay

mostra la informació d'accés d'escriptori remot de la VM.

virsh dominfo

Mostra informació de la màquina virtual, una descripció general del domini.

```

#2
$ virsh list --all
  Id   Name   State
-----
  -     alpine   shut off

$ virsh start alpine

```

```
Domain alpine started

$ virsh list
  Id   Name      State
-----
  2    alpine    running

$ virsh domdisplay alpine
spice://127.0.0.1:5900

$ virt-viewer alpine
<tancar el visor>

$ virsh destroy alpine
Domain alpine destroyed

$ virsh domstate alpine
shut off

$ virsh undefine alpine
Domain alpine has been undefined
```

Obtenir informació general del domini

Les màquines virtuals que es creen s'anomenen dominis i la seva informació s'emmagatzema en format XML. Aquestes són algunes de les ordres per obtenir informació:

[dominfo](#)

Mostra informació general descriptiva del domini

[domstat](#)

Indica l'estat de la VM (running, paused, shut-down)

[domstats](#)

Mostra estadístiques del domini, de la VM.

[domuuid](#)

Mostra el UUID identificador únic del domini (de la VM).

[domdisplay](#)

Mostra la informació necessària per accedir a la VM amb un client d'escriptori remot, la url i port.

[vcpu](#)

Mostra informació de les vcpu.

[vcpuinfo](#)

Mostra informació més detallada de les cpu.

[vcpupin](#)

Informació de les cpu.

[dumpxml](#)

Fa un volcat per stdout complet de la configuració XML de la màquina virtual.

[screenshot](#)

Desa una imatge (una foto, un screenshot o captura de pantalla) del que mostra l'escriptori de la VM.

[suspend](#)

Posa la VM en pausa.

resume

Reanuda l'execució d'una VM que s'havia posat en pausa anteriorment.

```
#3
$ virt-install --name=alpine --ram=2048 --nodisks --livecd
--cdrom=alpine-standard-3.15.0-x86_64.iso

$ virsh dominfo alpine
Id: 1
Nombre: alpine
UUID: 04a655aa-31c0-4fbf-8243-75be095fce23
Tipo de sistema operativo: hvm
Estado: ejecutando
CPU(s): 1
Hora de la CPU: 5,3s
Memoria máxima: 2097152 KiB
Memoria utilizada: 2097152 KiB
Persistente: si
Autoinicio: desactivar
Guardar administrado: no
Modelo de seguridad: none
DOI de seguridad: 0

$ virsh domstate alpine
running

$ virsh domuuid alpine
e3d35363-4bf6-46c1-bba5-24b250fd449d

$ virsh domstats alpine
Domain: 'alpine'
  state.state=1
  state.reason=1
  cpu.cache.monitor.count=0
  balloon.current=2097152
  balloon.maximum=2097152
  balloon.swap_in=0
  balloon.swap_out=0
  balloon.major_fault=0
  balloon.minor_fault=4185
  balloon.unused=1946048
  balloon.available=2029900
  balloon.usable=1875112
  balloon.last-update=1646247217
  balloon.disk_caches=20988
  balloon.hugetlb_pgalloc=0
  balloon.hugetlb_pgfail=0
  balloon.rss=365192
  vcpu.current=1
  vcpu.maximum=1
  vcpu.0.state=1
  vcpu.0.time=4880000000
  vcpu.0.wait=0
  net.count=1
  block.count=1
  block.0.name=hda
  block.0.path=/home/images/alpine-extended-3.15.0-x86_64.iso
  block.0.backingIndex=1
  block.0.rd.reqs=14496
  block.0.rd.bytes=158083294
  block.0.rd.times=382876239
  block.0.wr.reqs=0
  block.0.wr.bytes=0
  block.0.wr.times=0
  block.0.fl.reqs=0
  block.0.fl.times=0
  block.0.allocation=0
  block.0.capacity=689963008
  block.0.physical=689967104

$ virsh domdisplay alpine
spice://127.0.0.1:5900
```

```
#4
$ virsh vcpu alpine
vcpucount vcpuinfo vcpupin
$ virsh vcpucount alpine
maximum    config      1
maximum    live         1
current    config      1
current    live         1

$ virsh vcpuinfo alpine
VCPU:      0
CPU:       3
State:     running
CPU time:   5.5s
CPU Affinity:  yyyy

$ virsh vcpupin alpine
VCPU  CPU Affinity
-----
0      0-3
```

```
#5
$ virsh screenshot alpine
Screenshot saved to alpine-2022-03-02-20:00:25.ppm, with type of
image/x-portable-pixmap

$ virsh domstate alpine
running

$ virsh suspend alpine
Domain alpine suspended

$ virsh domstate alpine
paused

$ virsh resume alpine
Domain alpine resumed

$ virsh domstate alpine
running
```

```
#6
$ virsh dumpxml alpine | head
<domain type='kvm' id='1'>
  <name>alpine</name>
  <uuid>e3d35363-4bf6-46c1-bba5-24b250fd449d</uuid>
  <memory unit='KiB'>2097152</memory>
  <currentMemory unit='KiB'>2097152</currentMemory>
  <vcpu placement='static'>1</vcpu>
  <os>
    <type arch='x86_64' machine='pc-i440fx-4.2'>hvm</type>
    <boot dev='cdrom'>/>
  </os>
```

Afegir/eliminar elements a una VM (attach/detach)

- Devices
- Disk
- Interfaces

Comandes virsh attach i detach:

```
$ virsh attach-
attach-device      attach-disk      attach-interface

$ virsh detach-
detach-device      detach-device-alias  detach-disk      detach-interface
```

```
#6
$ virsh attach-device

$ virsh attach-disk

$ virsh attach-interface
```

```
$ virsh detach-device

$ virsh detach-disk

$ virsh detach-interface
```

```
#7
<disk type="block" device="disk">
  <driver name="qemu" type="raw" cache="none" io="native"
    ithread="2" iommu="on"/>
```

```
# virsh attach-device --config Guest1 ~/MyDisk.xml
```

```
<disk type='file' device='disk'>
  <driver name='qemu' type='raw' cache='none' />
  <source file='/var/lib/libvirt/images/FileName.img' />
  <target dev='vdb' />
</disk>
```

```
<disk type='file' device='cdrom'>
  <driver name='qemu' type='raw' cache='none' />
  <source file='/var/lib/libvirt/images/FileName.img' />
  <readonly />
  <target dev='hdc' />
</disk >
```

```
# virsh attach-device --config Guest1 ~/NewStorage.xml
```

```
#8
# virsh attach-disk {vm-name} \
--source /var/lib/libvirt/images/{img-name-here} \
--target vdb \
--persistent
```

```
#9
$ virsh attach-disk alpine --source /var/tmp/fedora.qcow2 --target vdb --persistent
Disk attached successfully
```

```
$ virsh domstats alpine
Domain: 'alpine'
state.state=1
state.reason=1
cpu.cache.monitor.count=0
balloon.current=2097152
balloon.maximum=2097152
balloon.swap_in=0
balloon.swap_out=0
balloon.major_fault=0
balloon.minor_fault=4708
```

```

balloon.unused=1950364
balloon.available=2029868
balloon.usable=1879456
balloon.last-update=1646249978
balloon.disk_caches=20988
balloon.hugetlb_pgalloc=0
balloon.hugetlb_pgfail=0
balloon.rss=356236
vcpu.current=1
vcpu.maximum=1
vcpu.0.state=1
vcpu.0.time=4520000000
vcpu.0.wait=0
net.count=1
block.count=2
block.0.name=hda
block.0.path=/home/images/alpine-extended-3.15.0-x86_64.iso
block.0.backingIndex=2
block.0.rd.reqs=14496
block.0.rd.bytes=158083294
block.0.rd.times=245459947
block.0.wr.reqs=0
block.0.wr.bytes=0
block.0.wr.times=0
block.0.fl.reqs=0
block.0.fl.times=0
block.0.allocation=0
block.0.capacity=689963008
block.0.physical=689967104
block.1.name=vdb
block.1.path=/var/tmp/fedora.qcow2
block.1.backingIndex=1
block.1.rd.reqs=147
block.1.rd.bytes=3772928
block.1.rd.times=3135642
block.1.wr.reqs=0
block.1.wr.bytes=0
block.1.wr.times=0
block.1.fl.reqs=0
block.1.fl.times=0
block.1.allocation=0
block.1.capacity=4295884800
block.1.physical=864256

$ virsh detach-disk alpine --target vdb
Disk detached successfully

```

```

#10
$ cat disc.xml
<disk type='file' device='disk'>
  <driver name='qemu' type='qcow2' cache='none' />
  <source file='/var/tmp/fedora.qcow2' />
  <target dev='hdb' />
</disk>

$ virsh destroy alpine
Domain alpine destroyed

$ virsh attach-device alpine --config disc.xml
Device attached successfully

$ virsh attach-device alpine --config disc.xml
Device attached successfully

$ virsh domstats alpine
Domain: 'alpine'
state.state=5
state.reason=2
balloon.current=2097152
balloon.maximum=2097152
vcpu.current=1
vcpu.maximum=1

```

```

block.count=2
block.0.name=hda
block.0.path=/home/images/alpine-extended-3.15.0-x86_64.iso
block.0.allocation=689967104
block.0.capacity=689963008
block.0.physical=689963008
block.1.name=hdb
block.1.path=/var/tmp/fedora.qcow2
block.1.allocation=864256
block.1.capacity=4294967296
block.1.physical=4295884800

$ virsh undefine alpine
Domain alpine has been undefined

```

Snapshots d'una VM

Comandes virsh per gestionar snapshots:

```

# virsh snapshot-
snapshot-create      snapshot-current      snapshot-dumpxml      snapshot-info
snapshot-parent
snapshot-create-as   snapshot-delete      snapshot-edit          snapshot-list
snapshot-revert

```

```

#11
$ virsh snapshot create

$ virsh snapshot list

$ virsh snapshot-revert

$ virsh snapshot-delete

```

```

#12
$ virsh list
 Id   Name      State
-----
 1    debian10  running

$ virsh snapshot-
snapshot-create      snapshot-current      snapshot-dumpxml      snapshot-info      snapshot-parent
snapshot-create-as   snapshot-delete      snapshot-edit          snapshot-list      snapshot-revert

$ virsh snapshot-create-as --domain test --
--atomic              --disk-only          --halt                --memspec
--no-metadata         --quiesce
--description         --diskspec          --live                --name
--print-xml           --reuse-external

$ virsh snapshot-create-as --domain debian10 --name "debian10-snap01" --description
"my first snapshot"
Domain snapshot debian10-snap01 created

$ virsh snapshot-list debian10
 Name                  Creation Time              State
-----
 debian10-snap01      2022-03-02 21:13:43 +0100  running

$ virsh snapshot-info --domain debian10 --snapshotname debian10-snap01
Name:      debian10-snap01
Domain:    debian10

```

```

Current:      yes
State:        running
Location:     internal
Parent:       -
Children:     0
Descendants:   0
Metadata:     yes

$ virsh snapshot-create-as --domain debian10 --name "debian10-snap02" --description
"my second snapshot"
Domain snapshot debian10-snap02 created

$ virsh snapshot-info --domain debian10 --snapshotname debian10-snap02
Name:         debian10-snap02
Domain:       debian10
Current:      yes
State:        running
Location:     internal
Parent:       debian10-snap01
Children:     0
Descendants:   0
Metadata:     yes

$ virsh snapshot-list debian10

```

Name	Creation Time	State
-----	-----	-----
debian10-snap01	2022-03-02 21:13:43 +0100	running
debian10-snap02	2022-03-02 21:15:37 +0100	running

```

#13
$ virsh snapshot-info --domain debian10 --current
Name:         debian10-snap02
Domain:       debian10
Current:      yes
State:        running
Location:     internal
Parent:       debian10-snap01
Children:     0
Descendants:   0
Metadata:     yes

$ virsh snapshot-revert --domain debian10 --snapshotname debian10-snap01 --running

$ virsh snapshot-info --domain debian10 --current
Name:         debian10-snap01
Domain:       debian10
Current:      yes
State:        running
Location:     internal
Parent:       -
Children:     1
Descendants:   1
Metadata:     yes

```

```

#14
$ virsh snapshot-delete --domain debian10 --snapshotname debian10-snap02
Domain snapshot debian10-snap02 deleted

$ virsh snapshot-list debian10

```

Name	Creation Time	State
-----	-----	-----
debian10-snap01	2022-03-02 21:13:43 +0100	running

```

$ virsh snapshot-info --domain debian10 --current
Name:         debian10-snap01
Domain:       debian10
Current:      yes
State:        running
Location:     internal
Parent:       -
Children:     0

```

```
Descendants:      0
Metadata:       yes

$ virsh snapshot-delete --domain debian10 --snapshotname debian10-snap01
Domain snapshot debian10-snap01 deleted

$ virsh snapshot-list debian10
Name      Creation Time    State
-----
$ virsh snapshot-info --domain debian10 --current
error: Domain snapshot not found: the domain does not have a current snapshot
```

Pràctica-3 Virsh

- 01 Proveu cada una de les ordres de les taules anteriors i anoteu quin és el seu significat.
- 02 Escriu el conjunt d'ordres necessaries per:
 - Engegar una VM ja existent
 - Llistar la informació de la màquina
 - Llistar la configuració XML de la màquina
 - Mostrar un screenshot de la seva pantalla
 - Accedir a la imatge usant un client d'escriptori remot
 - Fer una pausa de l'execució de la màquina
 - Reanudar la màquina i fer-ne un restart
 - Llistar la informació de xarxa de la màquina.
 - Aturar la màquina
 - Destruir-la.
- 03 Engegar una VM i treballar amb snapshots:
 - Crear un snapshot
 - Fer modificacions a la màquina
 - Crear un segon snapshoot.
 - Fer modificacions fatals a la màquina.
 - Revertir la màquina a l'estat corresponent al primer snapshot.

qemu

Amb les eines de qemu es poden visualitzar les VM de manera ràpida sense engegar virt-manager.

qemu-edid	qemu-img	qemu-keymap	qemu-nbd
qemu-system-i386	qemu-trace-stap		
qemu-ga	qemu-io	qemu-kvm	qemu-pr-helper
qemu-system-x86_64			

qemu / qemu-system-x86_64

Les options de configuració més usuals són:

- Memòria
- Numero de cpus
- Dispositius de disc i de cdrom
- Dispositius de xarxa
- Ordre de precedència de l'arrancada.

Exemples antics:

```
#1
$ qemu -cdrom brutalix_3.0_Beta.iso -vga std -m 512 &

#2
$ qemu -hda /dev/sdb1 -vga std -m 512 &

#3
$ qemu -hda /dev/sda6 -vga std -m 512 &

#4
$ qemu -hda /dev/sda -vga std -m 512 &

#5
$ qemu -hda mini-linux.img -vga std -m 512 &

#6
$ qemu -cdrom winXP_desatesa.iso -hda zero.img -vga std -m 512 &

#7
$ qemu -cdrom brutalix_3.0_Beta.iso -hda zero.img -hdb /dev/sda6 -vga std -m 512 &

#8
$ qemu -cdrom brutalix_3.0_Beta.iso -hda rawDisc1.img -hdb qcow2Disc1.img -vga std -m
```

qemu-img

Eina per crear i gestionar imatges, permet entre altres:

- Crear imatges de disk.
- Modificar imatges.

- Convertir formats.
- Xequeig d'imatges.
- Mostrar informació de les imatges.
- Snapshots. Instantànies de les imatges.

Exemples antics:

```
#1
$ qemu-img create -f raw rawDisc1.img 500M

#2
$ qemu-img info rawDisc1.img

#3
$ qemu-img create -f qcow2 qcow2Disc1.img 500M
```

Documentació:

- Qemu Read the docs: [qemu-img](#)
- Man page ([man](#))
- Qemu-img for [windows](#)

qemu-img allows you to create, convert and modify images offline. It can handle all image formats supported by QEMU.

Warning: Never use qemu-img to modify images in use by a running virtual machine or any other process; this may destroy the image. Also, be aware that querying an image that is being modified by another process may encounter inconsistent state.

Commands:

Amend

Amends the image format specific options for the image file filename. Not all file formats support this operation.

Bench

Run a simple sequential I/O benchmark on the specified image. If "-w" is specified, a write test is performed, otherwise a read test is performed.

Check

Perform a consistency check on the disk image filename. The command can output in the format ofmt which is either "human" or "json". The JSON output is an object of QAPI type "ImageCheck".

Commit

Commit the changes recorded in filename in its base image or backing file. If the backing file is smaller than the snapshot, then the backing file will be resized to be the same size as the snapshot. If the snapshot is smaller than the backing file, the backing file will not be truncated. If you want the backing file to match the size of the smaller snapshot, you can safely truncate it yourself once the commit operation successfully completes.

Compare

Check if two images have the same content. You can compare images with different format or settings.

Convert

Convert the disk image filename or a snapshot snapshot_param to disk image output_filename using format output_fmt. It can be optionally

compressed ("-c" option) or use any format specific options like encryption ("-o" option).

Create

Create the new disk image filename of size size and format fmt. Depending on the file format, you can add one or more options that enable additional features of this format.

dd

dd copies from input file to output file converting it from fmt format to output_fmt format.

Info

Give information about the disk image filename. Use it in particular to know the size reserved on disk which can be different from the displayed size. If VM snapshots are stored in the disk image, they are displayed too.

Map

Dump the metadata of image filename and its backing file chain. In particular, this commands dumps the allocation state of every sector of filename, together with the topmost file that allocates it in the backing file chain.

Measure

Calculate the file size required for a new image. This information can be used to size logical volumes or SAN LUNs appropriately for the image that will be placed in them. The values reported are guaranteed to be large enough to fit the image. The command can output in the format ofmt which is either "human" or "json". The JSON output is an object of QAPI type "BlockMeasureInfo".

Snapshot

List, apply, create or delete snapshots in image filename.

Rebase

Changes the backing file of an image. Only the formats "qcow2" and "qed" support changing the backing file.

Resize

Change the disk image as if it had been created with size.

Before using this command to shrink a disk image, you MUST use file system and partitioning tools inside the VM to reduce allocated file systems and partition sizes accordingly. Failure to do so will result in data loss!

After using this command to grow a disk image, you must use file system and partitioning tools inside the VM to actually begin using the new space on the device.

Pràctica-4 qemu / qemu-img

- 01 Engega usant qemu un DSL linux com per exemple alpine.
- 02 Engega usant qmeu el gparted live acompanyat d'una imatge qcow2 de disc, com per exemple el debian11.
- 03 Crear un disc raw de 2G anomenat disc01.raw. Mostrar amb info les seves característiques.

- 04 Convertir un disc raw a un altre format, per exemple al format qcow2. De fet no es converteix sinó que se'n genera un de nou en aquest format.

Configuracions de xarxa

☐ Default network (NAT)

☐ Private (named)

☐ Isolated

☐ Bridge

- **Libvirt Networking Handbook**

<https://jamielinux.com/docs/libvirt-networking-handbook/>

- **Libvirt VirtualNetworking**

<https://wiki.libvirt.org/page/VirtualNetworking>

- Libvirt Networking

<https://wiki.libvirt.org/page/Networking>

- KVM Networking

[Documentation](#)

```
$ virsh iface-  
iface-begin      iface-define      iface-edit      iface-name      iface-unbridge  
iface-bridge     iface-destroy    iface-list      iface-rollback  iface-undefine  
iface-commit     iface-dumpxml    iface-mac       iface-start
```

```
$ virsh net-  
net-autostart    net-dhcp-leases  net-info        net-port-delete  net-undefine  
net-create       net-dumpxml      net-list        net-port-dumpxml net-update  
net-define       net-edit         net-name        net-port-list    net-uuid  
net-destroy      net-event        net-port-create net-start
```

Default network (NAT)

Virtual Network: Default

- Realitza NAT.
- Els guest interns reben una adreça per DHCP del propi libvirt, del tipus [192.168.122.x/24](#).
- Des del guest s'accedeix a l'exterior (internet) a través del NAT, el host fa de gateway.
- Els guests que estan en aquesta xarxa es veuen entre ells, tenen connectivitat entre ells i connectivitat a l'exterior.
- El seu gateway és l'adreça IP [192.168.122.1](#) que correspon al host.
- En el host amfitrió es crea una interfície bridge virtual anomenada [virbr0](#) que té l'adreça [192.168.122.1/24](#).
- Aquesta interfície virbr0 del host amfitrió no ha de tenir assignades interfícies físiques reals dins el bridge., utilitza NAT i forwarding per accedir a l'exterior.
- Des de l'exterior (internet) no es pot accedir als guest que formen part d'aquesta xarxa interna.

Característiques de la xarxa default

- Podem observar les característiques de la xarxa [default](#) amb l'ordre `virsh` i les utilitats `virsh net`.
- A vegades la xarxa default no està activa per defecte en iniciar virt-manager, es pot activar des de l'applet gràfic o en mode comanda amb `virsh net-autostart`.
- En l'applet gràfic de virt-manager, edit, connection details, virtual networks es pot observar: nom, device, state, autostart i la configuració IPv4 (adreça IP, rang DHCP i NAT).
- També des de l'applet gràfic es pot observar la descripció XML de la xarxa default.

```
#1
$ virsh net-list
  Name          State    Autostart   Persistent
-----
  default       active   yes         yes

$ virsh net-info default
Name:          default
UUID:          b2b6b65b-7995-4f49-8d97-6b676f315f4c
Active:        yes
Persistent:    yes
Autostart:     yes
Bridge:        virbr0
```

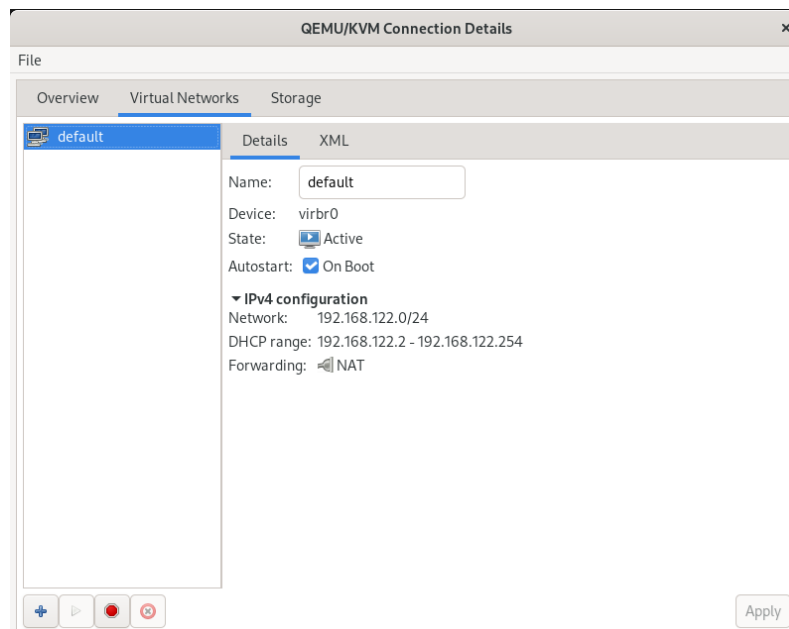
```
#2
$ virsh net-list --all
Name      State    Autostart Persistent
-----
default   active  yes      yes

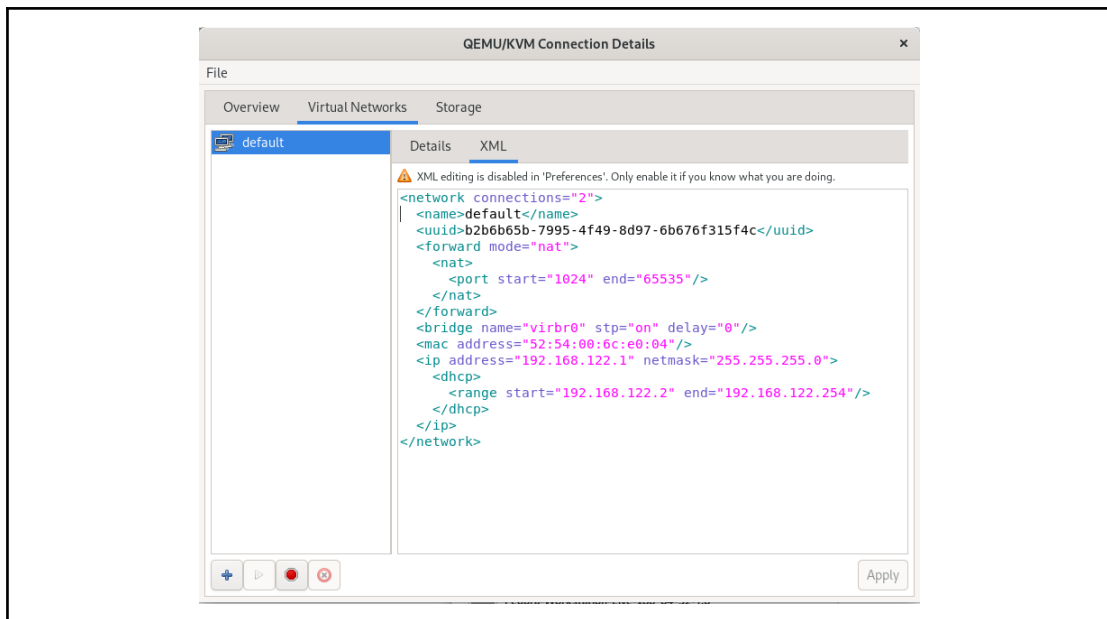
$ virsh net-autostart --disable --network default
Network default unmarked as autostarted

$ virsh net-list --all
Name      State    Autostart Persistent
-----
default   active  no       yes

$ virsh net-autostart --network default
Network default marked as autostarted

$ virsh net-list --all
Name      State    Autostart Persistent
-----
default   active  yes      yes
```





Descripció de les interfícies de xarxa en els guests i el host

- Tenim un host amfitrió 192.168.122.1
- Un guest debian10 192.168.122.176/24
- Un guest debian11 192.168.122.85/24

En el host amfitrió

```
$ ip a s virbr0
4: virbr0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
    group default qlen 1000
    link/ether 52:54:00:6c:e0:04 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0
        valid_lft forever preferred_lft forever

$ ip a s virbr0-nic
5: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc fq_codel master virbr0 state
DOWN group default qlen 1000
    link/ether 52:54:00:6c:e0:04 brd ff:ff:ff:ff:ff:ff

$ brctl show
bridge name      bridge id        STP enabled      interfaces
docker0          8000.02425a35b341 no                 virbr0
virbr0           8000.5254006ce004 yes                virbr0-nic
```

En el guest debian10

```
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 52:54:00:6c:6f:a4 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.176/24 brd 192.168.122.255 scope global dynamic enp1s0
        valid_lft 2849sec preferred_lft 2849sec
    inet6 fe80::5054:ff:fe6c:6fa4/64 scope link
        valid_lft forever preferred_lft forever
root@debian:~#
```

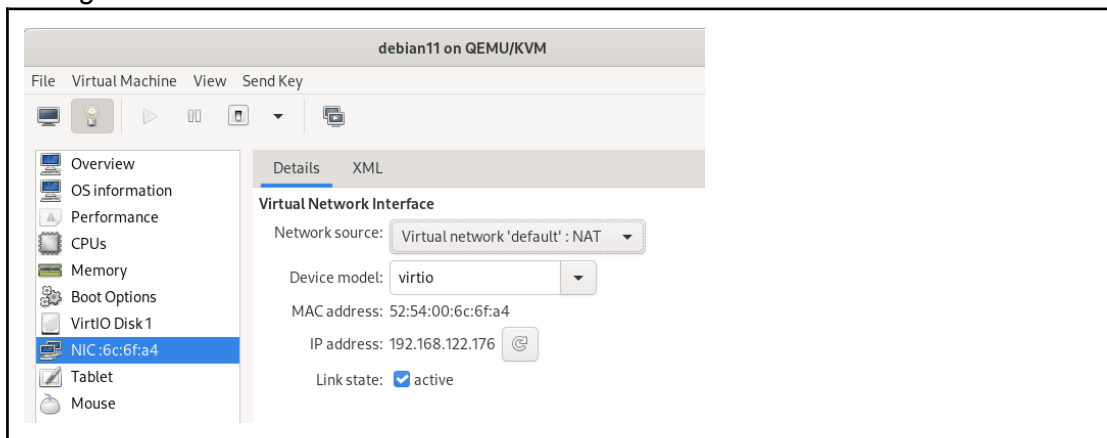
En el guest debian11

```

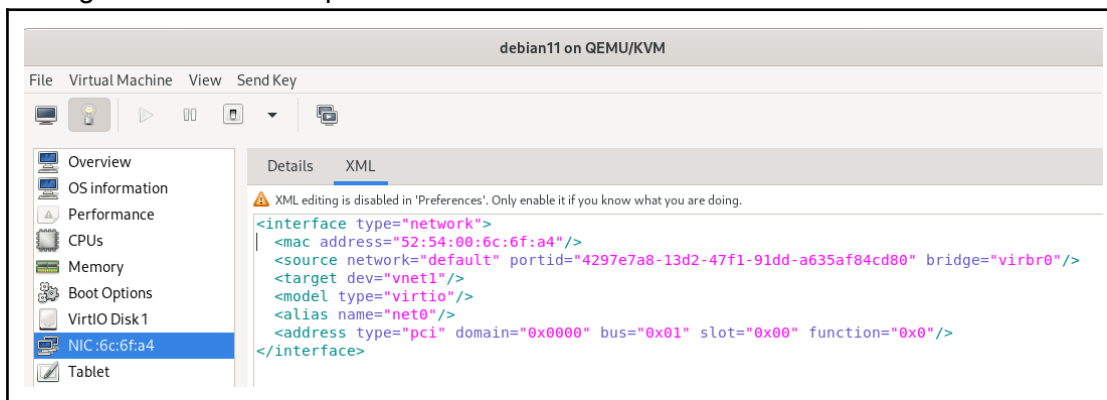
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 52:54:00:fc:14:b4 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.85/24 brd 192.168.122.255 scope global dynamic enp1s0
        valid_lft 3043sec preferred_lft 3043sec
    inet6 fe80::5054:ff:fefc:14b4/64 scope link
        valid_lft forever preferred_lft forever
root@debian:~# ip r
default via 192.168.122.1 dev enp1s0
192.168.122.0/24 dev enp1s0 proto kernel scope link src 192.168.122.85
root@debian:~#
root@debian:~# ping 192.168.122.176
PING 192.168.122.176 (192.168.122.176) 56(84) bytes of data.
64 bytes from 192.168.122.176: icmp_seq=1 ttl=64 time=0.608 ms
64 bytes from 192.168.122.176: icmp_seq=2 ttl=64 time=0.644 ms
^C
--- 192.168.122.176 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 10ms
rtt min/avg/max/mdev = 0.608/0.626/0.644/0.018 ms
root@debian:~#

```

Vista gràfica de la interfície de xarxa del debian 11:



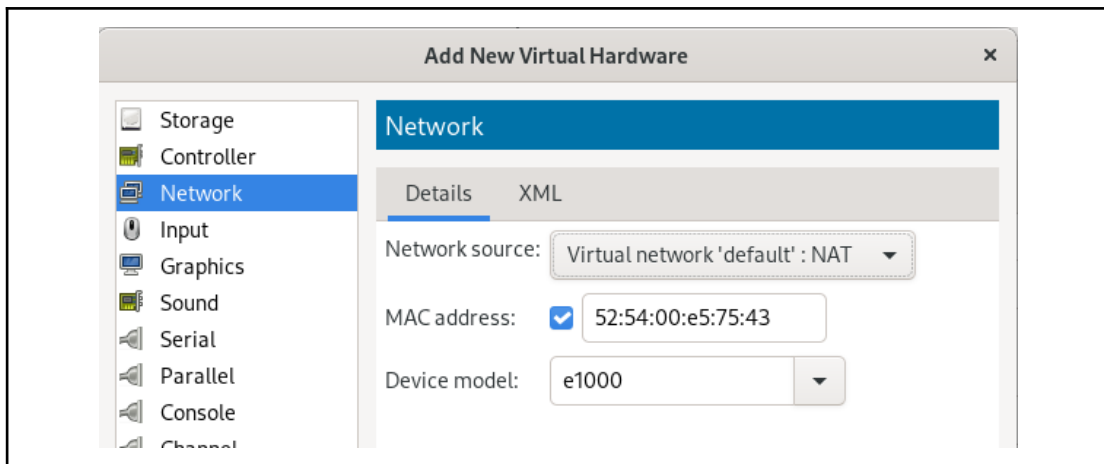
Vista gràfica de la descripció XML de la interfície:



Afegir una nova interfície de xarxa

- Als guests es poden afegir interfícies de xarxa i definir de quin tipus han de ser.

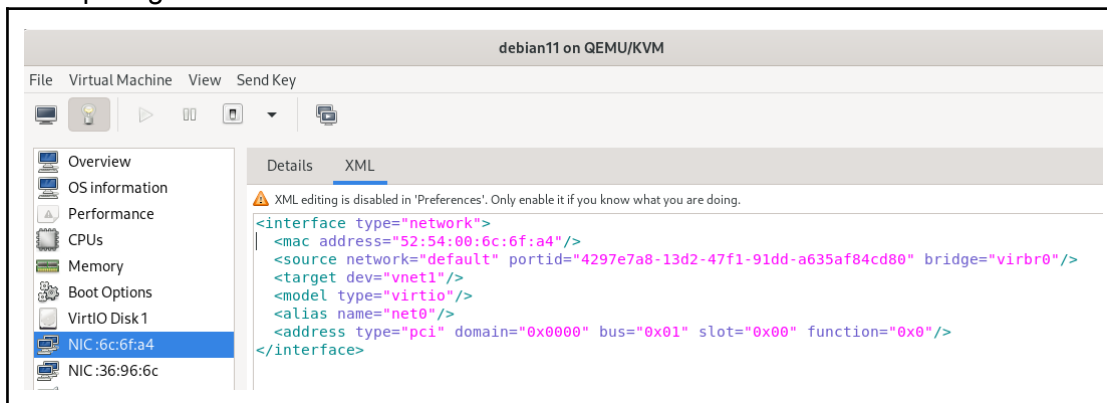
Exemple afegir una nova network de tipus default NAT:



Descripció XML de la interfície nova que s'afegirà:

```
<interface type="network">
  <source network="default"/>
  <mac address="52:54:00:e5:75:43"/>
  <model type="e1000"/>
</interface>
```

Un cop afegida i activada:



Des de dins de la VM observe aquesta segona interfície:

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 52:54:00:6c:6f:a4 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.176/24 brd 192.168.122.255 scope global dynamic enp1s0
        valid_lft 2447sec preferred_lft 2447sec
    inet6 fe80::5054:ff:fe6c:6fa4/64 scope link
        valid_lft forever preferred_lft forever
3: enp7s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 52:54:00:36:96:6c brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.100/24 brd 192.168.122.255 scope global dynamic enp7s0
        valid_lft 3547sec preferred_lft 3547sec
    inet6 fe80::5054:ff:fe36:966c/64 scope link
        valid_lft forever preferred_lft forever
root@debian:~#
```

Create a virtual private network

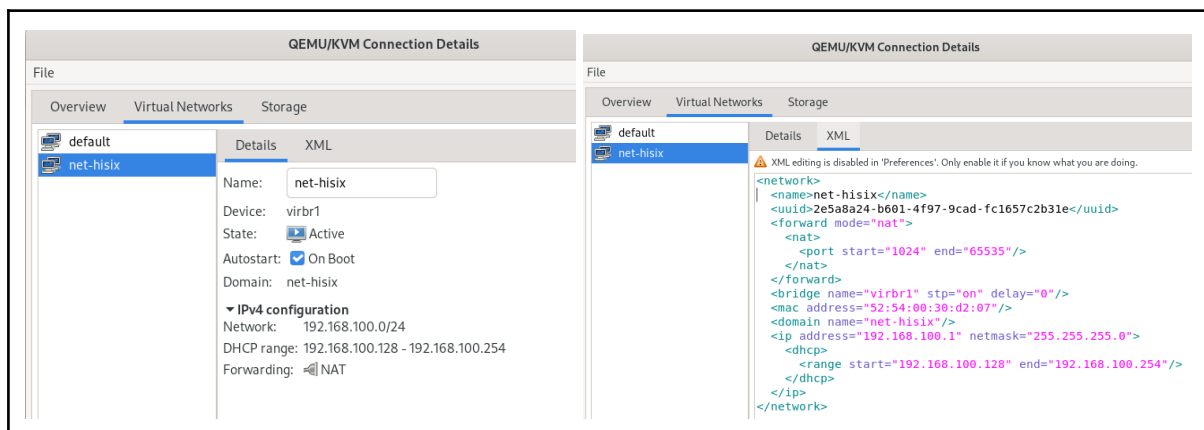
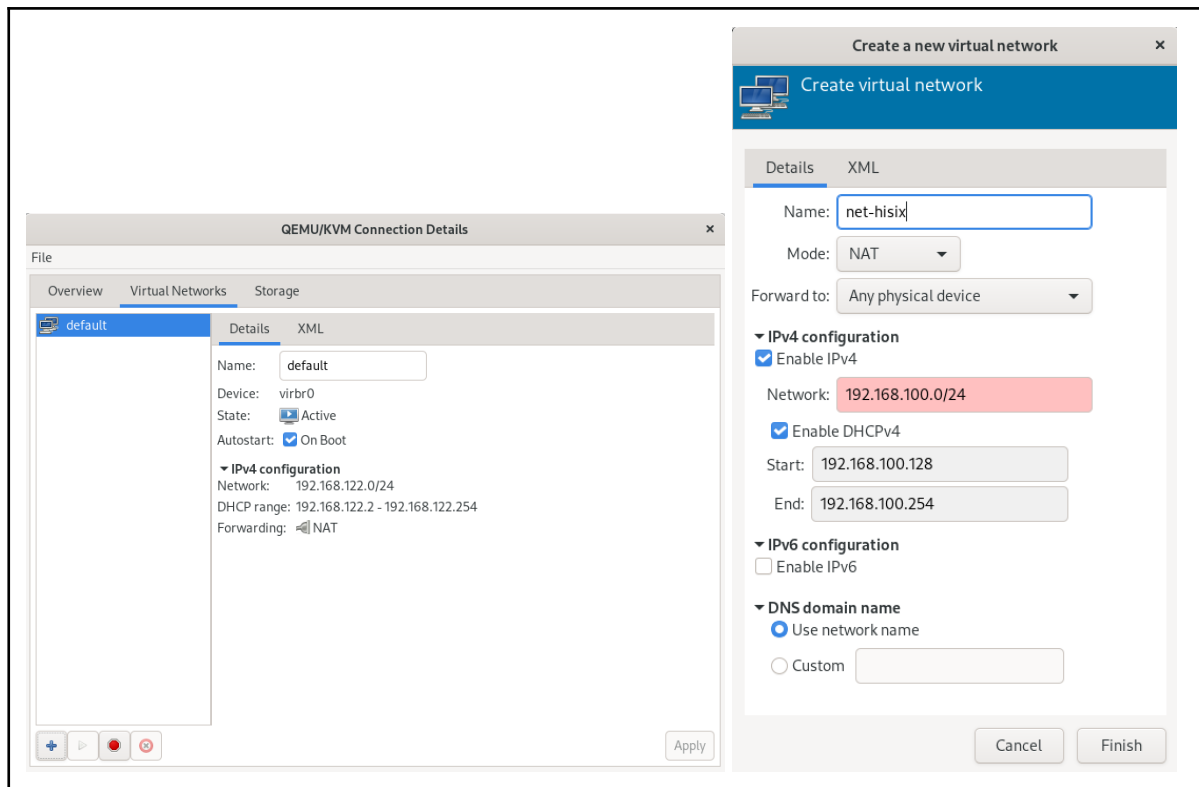
Característiques en la creació d'una nova xarxa virtual

- Nom
- Mode:
 - NAT
 - Routed
 - Open
 - Isolated
 - SR-IOV pool
- Forward to
 - Any physical device
 - List of devices...
- IPV4 configuration
 - Enable IPv4
 - Network (adreça de la xarxa)
 - Enable DHCPV4 (start address / end ddress)
- IPV6 configuration
 - Same configuration as IPV4
- DNS domain name
 - Use network name
 - Custom

<https://wiki.libvirt.org/page/TaskNATSetupVirtManager>

Procés de creació d'una xarxa privada virtual net-hisix

- Seleccionar connection details, virtual netowks i botó afegir.
- Definir el nom i el tipus de xarxa a usar
 - net-hisix
 - NAT
 - Activar IPV4



```
$ virsh net-list
Name          State    Autostart  Persistent
-----
default       active  yes        yes
net-hisix     active  yes        yes

$ virsh net-info net-hisix
Name:          net-hisix
UUID:          2e5a8a24-b601-4f97-9cad-fc1657c2b31e
Active:        yes
Persistent:    yes
Autostart:     yes
Bridge:        virbr1

$ virsh net-dumpxml --network net-hisix
<network connections='1'>
  <name>net-hisix</name>
  <uuid>2e5a8a24-b601-4f97-9cad-fc1657c2b31e</uuid>
  <forward mode='nat'>
    <nat>
```

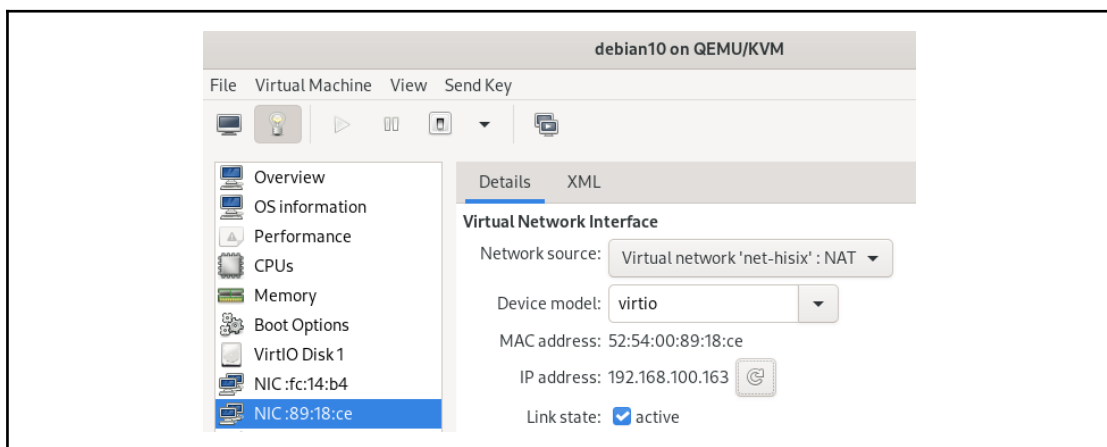
```

    <port start='1024' end='65535' />
  </nat>
</forward>
<bridge name='virbr1' stp='on' delay='0' />
<mac address='52:54:00:30:d2:07' />
<domain name='net-hisix' />
<ip address='192.168.100.1' netmask='255.255.255.0'>
  <dhcp>
    <range start='192.168.100.128' end='192.168.100.254' />
  </dhcp>
</ip>
</network>

```

Aplicar a un guest una interfície de xarxa en aquesta xarxa:

- Observem que el debian10 ara té una interfície nova.
- L'adreça IP correspon al nou rang d'adreces de la xarxa net-hisix.



```

root@debian:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 52:54:00:fc:14:b4 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.85/24 brd 192.168.122.255 scope global dynamic enp1s0
        valid_lft 3419sec preferred_lft 3419sec
    inet6 fe80::5054:ff:fefc:14b4/64 scope link
        valid_lft forever preferred_lft forever
4: enp7s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 52:54:00:89:18:ce brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.163/24 brd 192.168.100.255 scope global dynamic enp7s0
        valid_lft 3596sec preferred_lft 3596sec
    inet6 fe80::5054:ff:fe89:18ce/64 scope link
        valid_lft forever preferred_lft forever
root@debian:~#

```

```
$ virsh net-dhcp-leases --network net-hisix
```

Expiry Time	MAC address	Protocol	IP address	Hostname	Client ID or DUID
2022-03-04 22:31:42	52:54:00:89:18:ce	ipv4	192.168.100.163/24	debian	ff:00:89:18:ce:00:01:00:01:29:b5:32:92:52:54:00:61:5a:e6

Private

Isolated

<https://wiki.libvirt.org/page/TaskIsolatedNetworkSetupVirtManager>

Bridge

□ <https://wiki.debian.org/BridgeNetworkConnections>

Funcionament d'un bridge:

- Crear una interfície bridge
- Afegir una (o més) interfície física al bridge
- Configurar el bridge, en especial perquè NetworkManager no fastigui!

Configurar un bridge implica fer canvis en la configuració de xarxa del host. Cal crear una nova interfície de tipus bridge que és qui tindrà realment l'adreça IP. Aquesta configuració IP del bridge pot ser estàtica o dinàmica via DHCP. Crear la interfície bridge és fa diferent segons siguin sistemes d'una distribució o d'una altra.

Cal configurar el bridge perquè s'engegi automàticament en iniciar el sistema (si es vol així) i per incloure la interfície de xarxa ethernet o les interfícies que es requereixin. Això implica que ara la interfície de xarxa ethernet ja no té assignada una adreça IP i no n'ha de tenir, perquè és part del bridge.

Documentació:

- How to use bridged networking with libvirt and KVM
<https://linuxconfig.org/how-to-use-bridged-networking-with-libvirt-and-kvm>
- Create and Configure Bridge Networking For KVM in Linux ([computingforgeeks](#))
- [KVM Networking](#)

Crear / Configurar bridge en Debian

Procediment:

- Crear la interfície bridge br0
- Afegir la interfície eth0 (eno1) al bridge. En aquest punt pot perdre la ip i perdre la configuració.
- Editar /etc/network/interfaces
 - Modificar l'entrada de eth0 (eno1) per que sigui configurada manualment i no a través de network manager (`iface eno1 inet manual`).
 - Afegir l'entrada del bridge, que s'activi automàticament en iniciar el sistema (`auto br0`).
 - També que rebí la configuració via dhcp (`iface br0 inet dhcp`)
 - I finalment indicar que el bridge està format per la interfície eno1 (`bridge_ports eno1`). Reiniciar el servei de xarxa
- Verificar l'estat del bridge
- Reiniciar el host per verificar que en successives arrancades el sistema utilitza el bridge. **Atenció** que això generarà que a l'escriptori a la zona de notifikacions (dalt a

la dreta) mostri que la xarxa està unmanaged, perquè el networkManager no controla la eno1.

```
apt-get install bridge-utils

ip link add br0 type bridge
ip link set eno1 master br0
ip link show master br0

cp /etc/network/interfaces /etc/network/myinterfaces.bk

cat /etc/network/interfaces
auto lo
iface lo inet loopback

iface eno1 inet manual

auto br0
iface br0 inet dhcp
    bridge_ports eno1
    bridge_stp off

systemctl restart networking
systemctl status networking

brctl show
```

Si es vol configurar el bridge (o una interfície) amb valors estàtics en debian es fa configurant el fitxer `/etc/network/interfaces` amb els valors concrets:

```
# /etc/network/interfaces
# exemple interface eno1 estàtica

iface eno1 static
    address 192.168.200.2
    broadcast 192.168.200.255
    netmask 255.255.255.0
    gateway 192.168.200.1

# /etc/network/interfaces
# exemple interface bridge br0 estàtica

iface br0 static
    bridge_ports eno1
    address 192.168.200.2
    broadcast 192.168.200.255
    netmask 255.255.255.0
    gateway 192.168.200.1
```

Crear / Configurar bridge en Fedora

Procediment:

- Directori de configuració de xarxa: `/etc/sysconfig/network-scripts`
- Verificar que es disposa del paquet `network-scripts`.
- Crear la interfície del bridge creant un fitxer `ifcfg-br0` amb la seva configuració.
- Modificar la contiguració de la interfície ethernet per assignar-la al bridge.
- Reiniciar la xarxa


```
dnf install network scripts

# Exemple amb bridge amb adreça estàtica
cat /etc/sysconfig/network-scripts/ifcfg-br0
DEVICE=br0
TYPE=Bridge
BOOTPROTO=none
IPADDR=192.168.0.90
GATEWAY=192.168.0.1
NETMASK=255.255.255.0
ONBOOT=yes
DELAY=0
NM_CONTROLLED=0

# Exemple amb bridge amb adreça dinàmica
DEVICE=br0
TYPE=Bridge
BOOTPROTO=dhcp
ONBOOT=yes
DELAY=0
NM_CONTROLLED=0

cat /etc/sysconfig/network-scripts/ifcfg-en01
TYPE=ethernet
BOOTPROTO=none
NAME=en01
DEVICE=enp0s29u1u1
ONBOOT=yes
BRIDGE=br0
DELAY=0
NM_CONTROLLED=0

systemctl enable --now network
```

Gestió d'imatges de disc i pools d'emmagatzemament <pendent>

- ☐ virt-convert
- ☐ virt format
- ☐ virt-make-fs
- ☐ Virt-resize
- ☐ virt-sparsify
- ☐ qemu-img
- ☐ virsh vol-[command]
- ☐ virsh pool-[command]
- ☐ guestfishfs

Eines virt (2)

virt-alignment-scan	virt-df	virt-index-validate	virt-make-fs
virt-qemu-run	virt-tar-in		
virt-builder	virt-diff	virt-inspector	virt-manager
virt-rescue	virt-tar-out		
virt-builder-repository	virt-edit	virt-install	virtnetworkd
virt-resize	virtualbox		
virt-cat	virt-filesystems	virtinterfaced	virtnodedevd
virtsecret	virtualboxvm		
virt-clone	virt-format	virtlockd	virtnwfilterd
virt-sparsify	virt-viewer		
virt-copy-in	virtfs-proxy-helper	virt-log	
virt-pki-validate	virtstoraged	virt-xml	
virt-copy-out	virt-get-kernel	virtlogd	virtproxyd
virt-sysprep	virt-xml-validate		
virt-customize	virt-host-validate	virt-ls	virtqemud
virt-tail			

Utilitats generals virt-[command]

```
$ virt
virt-alignment-scan      virt-format              virt-make-fs            virt-sparsify
virt-builder            virtfs-proxy-helper     virt-manager            virtstoraged
virt-builder-repository virt-get-kernel          virtnetworkd            virt-sysprep
virt-cat                virt-host-validate      virtnodetdevd           virt-tail
virt-clone              virt-index-validate     virtnwfilterd           virt-tar-in
virt-copy-in            virt-inspector           virt-pki-validate       virt-tar-out
virt-copy-out           virt-install            virtproxyd              virtualbox
virt-customize           virtinterfaced          virtqemu                virtualboxvm
virt-df                 virtlockd               virt-qemu-run           virt-viewer
virt-diff               virt-log                 virt-rescue              virt-xml
virt-edit               virtlogd                 virt-resize
virt-xml-validate
virt-filestystems       virt-ls                  virtsecret
```

Utilitats Virt:

- [virt-cat](#) show file content inside an VM or disk
- [virt-copy-in](#) copy from localdisk to an VM or disk
- [virt-copy-out](#) copy from an VM or disk to localdir
- [virt-df](#) show disk free information of an VM or disk
- [virt-diff](#) compare two disks
- [virt-edit](#) edit a text file inside an VM ord disk
- [virt-format](#) format a disk (create a blank disk)
- [virt-get-kernel](#) show the kernel files inside an VM or disk
- [virt-inspector](#) show information about an VM or disk
- [virt-ls](#) list a directory inside an VM or disk
- [virt-make-fs](#) create a filesystem from a tar or from a dir (like mkisofs)
- [virt-rescue](#) open a rescue shell in the VM or disc
- [virt-resize](#) resizes (grow and shrink) the disk images of a VM
- [virt-sparsify](#) create from a raw disk on sparsified disk (thin provisioning)
- [virt-tail](#) 'follow' a file inside the VM (like tail -f)
- [virt-tar-in](#) Unpack a tarball into a virtual machine disk image
- [virt-tar-out](#) Pack a virtual machine disk image directory into a tarball
- [virt-top](#) utility like top for VM

```
#1
$ virt-install --name=alpine --ram=2048 --nodisks --livecd
--cdrom=alpine-extended-3.15.0-x86_64.iso &

$ virt-install --name debian11 --import --memory 2048 --vcpus 2 --os-variant debian10
--disk debian-11-nocloud-amd64.qcow2 &

$ virsh list
 Id   Name      State
-----
 1    alpine    running
 2    debian11  running

$ export LIBGUESTFS_BACKEND=direct
```

```
#2
$ virt-cat -d alpine /etc/fstab

$ virt-cat -d debian11 /etc/fstab
# /etc/fstab: static file system information
UUID=05392894-ef9f-4e2c-878e-1dea2e7da980 / ext4
rw,discard,errors=remount-ro,x-systemd.growfs 0 1
UUID=DE5D-9DD8 /boot/efi vfat defaults 0 0

$ virt-cat -a Fedora-Cloud-Base-27-1.6.x86_64.qcow2 /etc/fstab
#
# /etc/fstab
# Created by anaconda on Sun Nov  5 07:22:11 2017
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
UUID=d020d197-657b-4030-bde5-0db8417bc39c / ext4          defaults          1 1

$ virt-cat -a debian-10-genericcloud-amd64-20210329-591.qcow2 /etc/fstab
# /etc/fstab: static file system information
UUID=2a32edc5-aef2-4dcc-93ee-9cd605341279 / ext4
rw,discard,errors=remount-ro,x-systemd.growfs 0 1
UUID=A2EB-3AB7 /boot/efi vfat defaults 0 0
```

```
#3
$ virt-df -d alpine

$ virt-df -a Fedora-Cloud-Base-27-1.6.x86_64.qcow2
Filesystem              1K-blocks      Used    Available   Use%
Fedora-Cloud-Base-27-1.6.x86_64.qcow2:/dev/sda1
                        4061888        589784     3246056     15%

$ virt-df -a gparted-live-1.2.0-1-amd64.iso
Filesystem              1K-blocks      Used    Available   Use%
gparted-live-1.2.0-1-amd64.iso:/dev/sda1
                        396256         396256         0     100%
```

```
#4

$ echo "that's the content!" > /tmp/dades.md

$ virt-copy-in -a debian-10-genericcloud-amd64-20210329-591.qcow2 /tmp/dades.md /opt

$ virt-cat -a debian-10-genericcloud-amd64-20210329-591.qcow2 /opt/dades.md
that's the content!

$ virt-copy-out -a debian-10-genericcloud-amd64-20210329-591.qcow2 /etc/services /tmp

$ ls /tmp/services
/tmp/services

$ virt-edit -a Fedora-Cloud-Base-27-1.6.x86_64.qcow2 /etc/os-release

$ virt-cat -a Fedora-Cloud-Base-27-1.6.x86_64.qcow2 /etc/os-release
NAME=Fedora
VERSION="27 (Cloud Edition)"
ID=fedora
VERSION_ID=27
PRETTY_NAME="Fedora 27 (Cloud Edition)"
ANSI_COLOR="0;34"
CPE_NAME="cpe:/o:fedoraproject:fedora:27"
HOME_URL="https://fedoraproject.org/"
SUPPORT_URL="https://fedoraproject.org/wiki/Communicating_and_getting_help"
BUG_REPORT_URL="https://bugzilla.redhat.com/"
REDHAT_BUGZILLA_PRODUCT="Fedora"
REDHAT_BUGZILLA_PRODUCT_VERSION=27
REDHAT_SUPPORT_PRODUCT="Fedora"
REDHAT_SUPPORT_PRODUCT_VERSION=27
PRIVACY_POLICY_URL="https://fedoraproject.org/wiki/Legal:PrivacyPolicy"
```

```
VARIANT="Cloud Edition"
VARIANT_ID=cloud
# Comment added
```

```
#5
$ virt-filesystems -d debian11
/dev/sda1
/dev/sda15

$ virt-filesystems -a debian-10-genericcloud-amd64-20210329-591.qcow2
/dev/sda1
/dev/sda15

$ virt-filesystems -a Fedora-Cloud-Base-27-1.6.x86_64.qcow2
/dev/sda1

$ virt-filesystems -a gparted-live-1.2.0-1-amd64.iso
/dev/sda1

$ virt-ls -a Fedora-Cloud-Base-27-1.6.x86_64.qcow2 /tmp
.ICE-unix
.Test-unix
.X11-unix
.XIM-unix
.font-unix
ks-script-jnovkove
ks-script-nllqch6t
systemd-private-33e4e968a6fb461fac4d3a9cc34202d1-chrond.service-tuPsUr
```

```
#6
$ virsh vol-create-as images disk01.qcow2 2G
Vol disk01.qcow2 created

$ ls -l disk01.qcow2
-rw-----. 1 root root 2147483648 Mar  4 17:40 disk01.qcow2

$ virt-format -a disk01.qcow2
```

```
#7
$ virt-get-kernel -d debian11
download: /boot/vmlinuz-5.10.0-9-amd64 -> ./vmlinuz-5.10.0-9-amd64
download: /boot/initrd.img-5.10.0-9-amd64 -> ./initrd.img-5.10.0-9-amd64

$ virt-get-kernel -a Fedora-Cloud-Base-27-1.6.x86_64.qcow2
download: /boot/vmlinuz-4.13.9-300.fc27.x86_64 -> ./vmlinuz-4.13.9-300.fc27.x86_64
download: /boot/initramfs-4.13.9-300.fc27.x86_64.img ->
./initramfs-4.13.9-300.fc27.x86_64.img

$ virt-get-kernel -a debian-10-genericcloud-amd64-20210329-591.qcow2
download: /boot/vmlinuz-4.19.0-16-cloud-amd64 -> ./vmlinuz-4.19.0-16-cloud-amd64
download: /boot/initrd.img-4.19.0-16-cloud-amd64 -> ./initrd.img-4.19.0-16-cloud-amd64

$ virt-inspector -a debian-10-genericcloud-amd64-20210329-591.qcow2 | head
<?xml version="1.0"?>
<operatingsystems>
  <operatingsystem>
    <root>/dev/sda1</root>
    <name>linux</name>
    <arch>x86_64</arch>
    <distro>debian</distro>
    <product_name>10.9</product_name>
    <major_version>10</major_version>
    <minor_version>9</minor_version>

$ virt-log -d alpine

$ virt-log -d debian11
```

```
#8
$ virt-make-fs /etc/ etc.img

$ ls -lh etc.img
-rw-r--r--. 1 root root 32M Mar  4 18:01 etc.img

$ file etc.img
etc.img: Linux rev 1.0 ext2 filesystem data, UUID=02b41180-a4d0-434d-be56-0ab0f3c41eaa
(large files)

$ qemu-img info etc.img
image: etc.img
file format: raw
virtual size: 31.6 MiB (33115648 bytes)
disk size: 29.6 MiB

$ virsh attach-disk debian11 --source /home/images/etc.img --target vdb --persistent
Disk attached successfully

$ virt-filesystems -d debian11
/dev/sdb
/dev/sda1
/dev/sda15

$ virsh detach-disk debian11 --target vdb --persistent
Disk detached successfully
```

```
#9
$ virt-rescue --ro -d alpine
The virt-rescue escape key is '^j'. Type '^j h' for help.

-----

Welcome to virt-rescue, the libguestfs rescue shell.

Note: The contents of / (root) are the rescue appliance.
You have to mount the guest's partitions under /sysroot
before you can examine them.

><rescue>
```

```
#10
# EXAMPLES virt-resize

1. This example takes "olddisk" and resizes it into "newdisk", extending one of the
guest's partitions to fill the extra 5GB of space:

    virt-filesystems --long -h --all -a olddisk

    truncate -r olddisk newdisk
    truncate -s +5G newdisk

    # Note "/dev/sda2" is a partition inside the "olddisk" file.
    virt-resize --expand /dev/sda2 olddisk newdisk

2. As above, but make the /boot partition 200MB bigger, while giving the remaining
space to /dev/sda2:

    virt-resize --resize /dev/sda1=+200M --expand /dev/sda2 \
    olddisk newdisk

3. As in the first example, but expand a logical volume as the final step. This is
what you would typically use for Linux guests that use LVM:

    virt-resize --expand /dev/sda2 --LV-expand /dev/vg_guest/lv_root \
    olddisk newdisk

4. As in the first example, but the output format will be qcow2 instead of a raw disk:
```

```
qemu-img create -f qcow2 -o preallocation=metadata newdisk.qcow2 15G
virt-resize --expand /dev/sda2 olddisk newdisk.qcow2
```

```
#11
# Virt-sparsify is a tool which can make a virtual machine disk (or any disk image)
sparse a.k.a. Thin-provisioned

$ dd if=/dev/zero of=disk01.raw bs=1k count=2M status=progress
2085735424 bytes (2.1 GB, 1.9 GiB) copied, 13 s, 160 MB/s
2097152+0 records in
2097152+0 records out
2147483648 bytes (2.1 GB, 2.0 GiB) copied, 13.471 s, 159 MB/s

$ ls -lh disk01.raw
-rw-r--r--. 1 root root 2.0G Mar  4 18:30 disk01.raw
$ du -sh disk01.raw
2.1G    disk01.raw

$ qemu-img info disk01.raw
image: disk01.raw
file format: raw
virtual size: 2 GiB (2147483648 bytes)
disk size: 2 GiB

$ virt-sparsify disk01.raw --convert qcow2 disk01.qcow2
[ 0.0] Create overlay file in /tmp to protect source disk
[ 0.0] Examine source disk
[ 1.1] Copy to destination and make sparse
[ 4.3] Sparsify operation completed with no errors.
virt-sparsify: Before deleting the old disk, carefully check that the
target disk boots and works correctly.

$ ls -lh disk01.*
-rw-r--r--. 1 root root 193K Mar  4 18:32 disk01.qcow2
-rw-r--r--. 1 root root 2.0G Mar  4 18:30 disk01.raw

$ du -sh disk01.qcow2
196K    disk01.qcow2

$ qemu-img info disk01.qcow2
image: disk01.qcow2
file format: qcow2
virtual size: 2 GiB (2147483648 bytes)
disk size: 196 KiB
cluster_size: 65536
Format specific information:
    compat: 1.1
    lazy refcounts: false
    refcount bits: 16
    corrupt: false
```

```
#12
$ virt-tail -d debian11 /var/log/messages
--- /var/log/messages ---
Mar 4 17:16:11 debian kernel: [ 3901.801322] pci 0000:07:00.0: BAR 1: assigned [mem 0xfc000000-0xfc000fff]
Mar 4 17:16:11 debian kernel: [ 3901.802409] pcieport 0000:00:02.6: PCI bridge to [bus 07]
Mar 4 17:16:11 debian kernel: [ 3901.803251] pcieport 0000:00:02.6: bridge window [io 0x7000-0x7fff]
Mar 4 17:16:11 debian kernel: [ 3901.805365] pcieport 0000:00:02.6: bridge window [mem 0xfc000000-0xfc1fffff]
Mar 4 17:16:11 debian kernel: [ 3901.807002] pcieport 0000:00:02.6: bridge window [mem 0xfde00000-0xfdf7ffff]
```

```
#13
$ virt-tar-out -d debian11 /etc /tmp/etc.tar

$ ls -lh /tmp/etc.tar
-rw-r--r--. 1 root root 2.0M Mar  4 18:41 /tmp/etc.tar

$ tar -tf /tmp/etc.tar | head
./
./ld.so.conf
./sudoers.d/
```



```
./sudoers.d/README
./pam.d/
./pam.d/other
./pam.d/common-account
./pam.d/sudo
./pam.d/su
./pam.d/login

$ virt-tar-in --live -a Fedora-Cloud-Base-27-1.6.x86_64.qcow2 /tmp/documentation.tar /tmp

$ virt-ls -a Fedora-Cloud-Base-27-1.6.x86_64.qcow2 /tmp/usr/share
doc
man
```

```
#14
$ virt-top

virt-top 19:08:40 - x86_64 4/4CPU 3142MHz 7468MB 0.3% 0.6% 0.4% 0.3% 0.3% 0.4% 0.4% 0.3%
13 domains, 2 active, 2 running, 0 sleeping, 0 paused, 11 inactive D:0 O:0 X:0
CPU: 0.3% Mem: 4096 MB (4096 MB by guests)

  ID S RDRQ WRRQ RXBY TXBY %CPU %MEM  TIME  NAME
  -- -- --
    1 R    0    0    0    0  0.2 27.0 0:42.77 alpine
    2 R    0    0  104    0  0.2 27.0 0:56.79 debian11
```

Crear / Clonar / Preparar màquines virtuals <pendent>

- ☐ virt-builder
- ☐ virt-builder-repository
- ☐ Virt-clone
- ☐ virt-sysprep
- ☐ virt-customize

- ☐ virt-install
- ☐ vish [command]

virt-builder

Virt-builder is a tool for quickly building new virtual machines. You can build a variety of VMs for local or cloud use, usually within a few minutes or less. Virt-builder also has many ways to customize these VMs. Everything is run from the command line and nothing requires root privileges, so automation and scripting is simple.

Note that virt-builder does not install guests from scratch. It takes cleanly prepared, digitally signed OS templates and customizes them. This approach is used because it is much faster, but if you need to do fresh installs you may want to look at virt-install(1) and oz-install(1).

```
#1
$ virt-builder -l | head
opensuse-13.1      x86_64      opensUSE 13.1
opensuse-13.2      x86_64      opensUSE 13.2
opensuse-42.1      x86_64      opensUSE Leap 42.1
opensuse-tumbleweed x86_64      opensUSE Tumbleweed
alma-8.5           x86_64      AlmaLinux 8.5
centos-6           x86_64      CentOS 6.6
centos-7.0         x86_64      CentOS 7.0
centos-7.1         x86_64      CentOS 7.1
centos-7.2         x86_64      CentOS 7.2 (aarch64)
centos-7.2         x86_64      CentOS 7.2

$ virt-builder -l | grep debian
debian-6           x86_64      Debian 6 (Squeeze)
debian-7           sparc64     Debian 7 (Wheezy) (sparc64)
debian-7           x86_64      Debian 7 (wheezy)
debian-8           x86_64      Debian 8 (jessie)
debian-9           x86_64      Debian 9 (stretch)
debian-10          x86_64      Debian 10 (buster)
debian-11          x86_64      Debian 11 (bullseye)

$ virt-builder -notes debian-11
Debian 11 (bullseye)

This is a minimal Debian install.

This image is so very minimal that it only includes an ssh server
This image does not contain SSH host keys. To regenerate them use:

--firstboot-command "dpkg-reconfigure openssh-server"
```

This template was generated by a script in the libguestfs source tree:
builder/templates/make-template.ml
Associated files used to prepare this template can be found in the
same directory.

```
#2
#Examples (from man page)

virt-builder fedora-27

virt-builder fedora-27 -o mydisk.img

virt-builder fedora-27 --format qcow2

virt-builder fedora-27 --size 20G

virt-builder fedora-27 -o mydisk.qcow2 -format qcow2 --size 20G

virt-builder fedora-27 --root-password file:/tmp/rootpw

virt-builder fedora-27 --hostname virt.example.com

virt-builder fedora-27 --install "inkscape,@Xfce Desktop, geany"

virt-builder debian-11 --update
```

```
#3
cat <<'EOF' > /tmp/dnf-update.sh
    dnf -y --best update
EOF

virt-builder fedora-27 --firstboot /tmp/dnf-update.sh

virt-builder fedora-27 --firstboot-command 'dnf -y --best update'

virt-builder fedora-27 \
    --edit '/etc/dnf/dnf.conf:
        s/gpgcheck=1/gpgcheck=0/'

virt-builder --firstboot-command \
    'useradd -m -p "" rjones ; chage -d 0 rjones'

virt-builder centos-6 \
    --edit '/etc/sysconfig/keyboard: s/^KEYTABLE=.* /KEYTABLE="uk"/'
```

virt-clone

virt-clone is a command line tool for cloning existing virtual machine images using the "libvirt" hypervisor management library. It will copy the disk images of any existing virtual machine, and define a new guest with an identical virtual hardware configuration. Elements which require uniqueness will be updated to avoid a clash between old and new guests.

The --auto-clone option will generate all needed input, aside from the source guest to clone. virt-clone does not change anything inside the guest OS, it only duplicates disks and does host side changes.

So things like changing passwords, changing static IP address, etc are outside the scope of this tool. For these types of changes, please see virt-sysprep.

```
#1
```

```
# Examples (from man page)
# Clone the guest called "demo" on the default connection, auto generating a new name and
disk clone path.
```

```
# virt-clone \
    --original demo \
    --auto-clone
```

```
# Clone the guest called "demo" which has a single disk to copy
```

```
# virt-clone \
    --original demo \
    --name newdemo \
    --file /var/lib/xen/images/newdemo.img
```

```
# Clone a QEMU guest with multiple disks
```

```
# virt-clone \
    --connect qemu:///system \
    --original demo \
    --name newdemo \
    --file /var/lib/xen/images/newdemo.img \
    --file /var/lib/xen/images/newdata.img
```

```
# Clone a guest to a physical device which is at least as big as the original guests
disks. If the destination device is bigger, the new guest can do a filesystem resize
when it boots.
```

```
# virt-clone \
    --connect qemu:///system \
    --original demo \
    --name newdemo \
    --file /dev/HostVG/DemoVM \
    --mac 52:54:00:34:11:54
```

virt-sysprep

Virt-sysprep can reset or unconfigure a virtual machine so that clones can be made from it. Steps in this process include removing SSH host keys, removing persistent network MAC configuration, and removing user accounts. Virt-sysprep can also customize a virtual machine, for instance by adding SSH keys, users or logos. Each step can be enabled or disabled as required.

Virt-sysprep modifies the guest or disk image in place. The guest must be shut down. If you want to preserve the existing contents of the guest, you must snapshot, copy or clone the disk first. You do not need to run virt-sysprep as root.

virt-customize

Virt-customize can customize a virtual machine (disk image) by installing packages, editing configuration files, and so on.

Virt-customize modifies the guest or disk image in place. The guest must be shut down. If you want to preserve the existing contents of the guest, you must snapshot, copy or clone the disk first. You do not need to run virt-customize as root

--

Clients d'escriptoris remots <pendent>

- ☐ virt-viewer
- ☐ spice
- ☐ vncviewer
- ☐ remmina

Descripció XML de les VM <pendent>

- ☐ virt-xml
- ☐ virt-xml-vlidade
- ☐ virsh [command]

Instal·lacions

☐ Fedora-32

☐ Debian-11

☐ Ubuntu-20

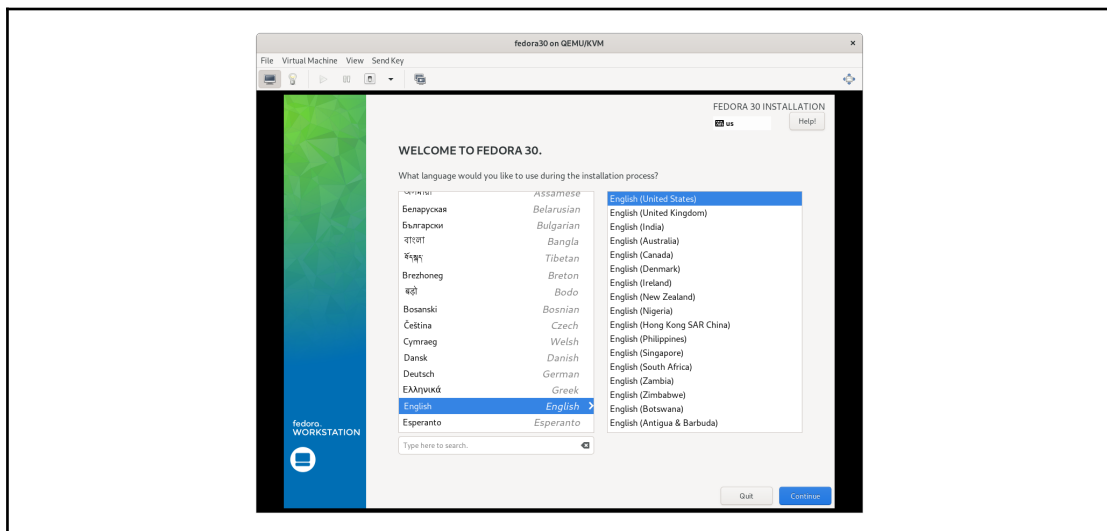
Fedora-32

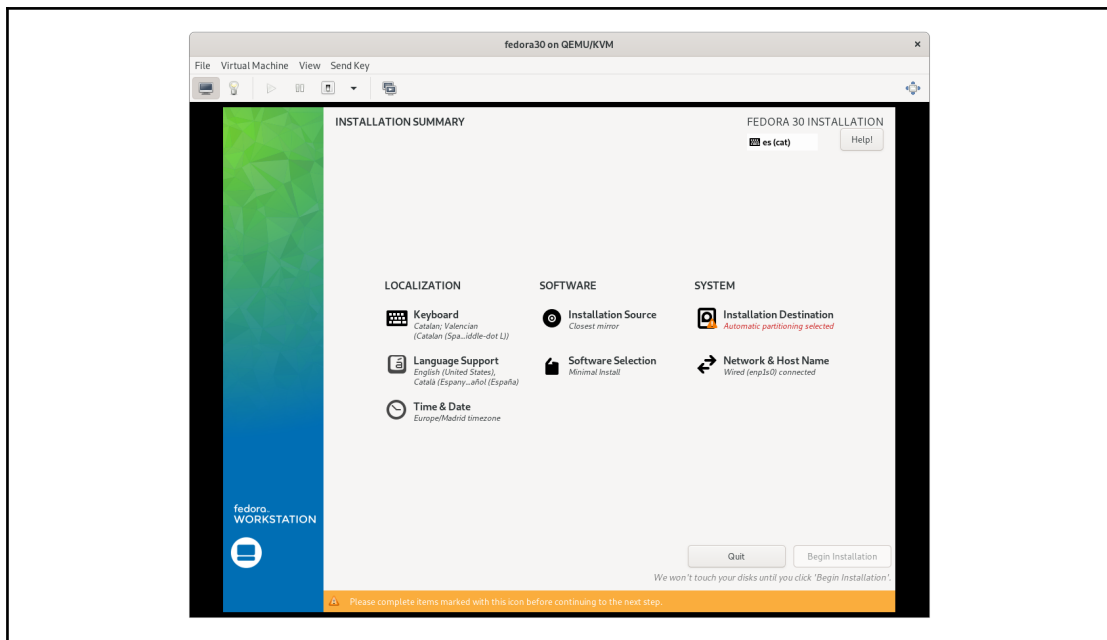
Layout per fer particions:

- Tot en una sola, sense swap
- Partició arrel (/) i partició swap.
- Recomanat: /boot, / (arrel), swap
- Amb homes: /boot, / (arrel), /home, swap
- Altres punts de muntatge: /boot, /home, /var, /run, etc

Instal·lació Fedora-32

- Components generals: installation-language, keyboard, language, source (closest mirror), software (minimal)
- Particionat: automàtic
- Característiques: 325 paquets, 5-10 min, 300MiB
- reboot, atenció a eliminar ara el device de CD, ja no cal, però no eliminar el fitxer .iso associat.
- Ordres a verificar: loadkeys es, df -h, lsbl, mount

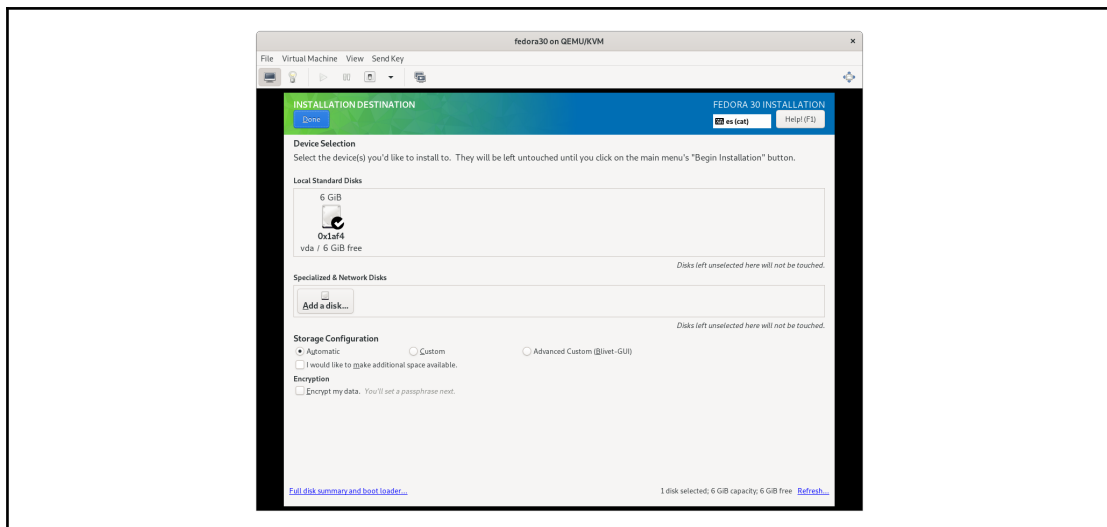




Opcions de particionat de Fedora-32: **automàtic**

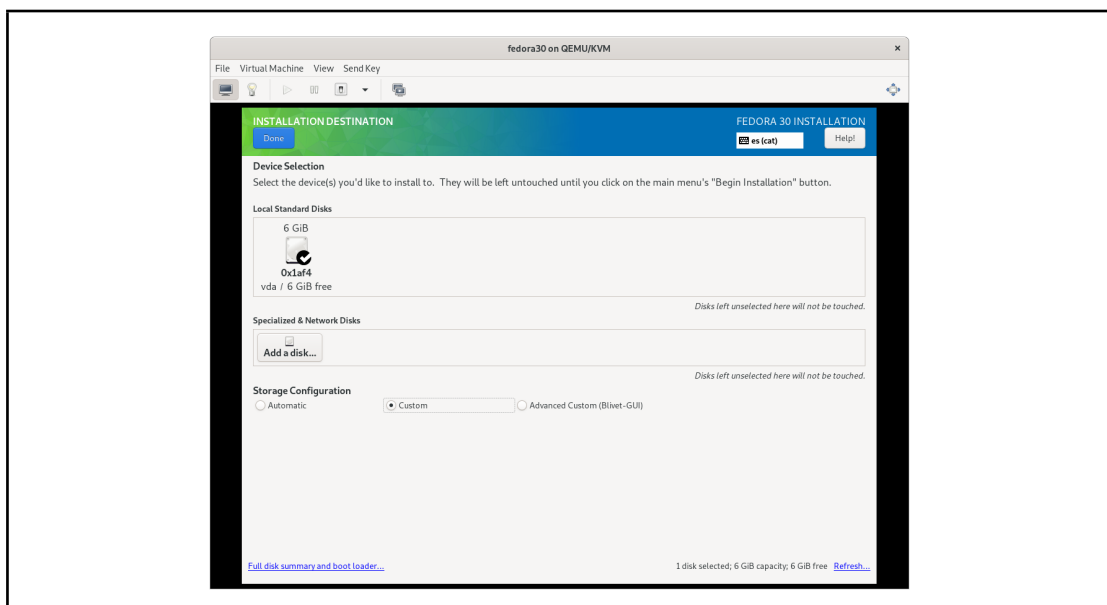
- automàtic
 - genera una partició /dev/vda1 de /boot
 - una partició /dev/vda de la que en fa un LVM un anomenat
 - fedora-root que és la / (arrel)
 - fedora-swap que és la partició swap

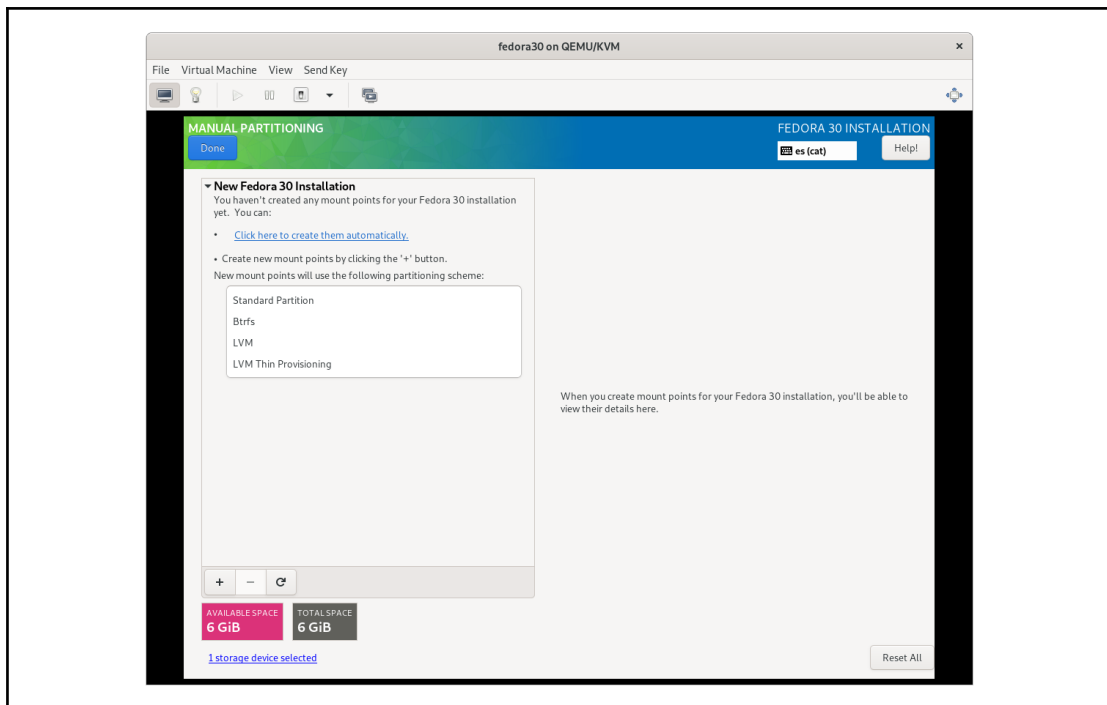
```
[root@localhost ~]# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sr0                  11:0    1 1024M  0 rom
vda                  252:0    0    6G  0 disk
├─vda1                252:1    0    1G  0 part /boot
└─vda2                252:2    0    5G  0 part
   └─fedora-root       253:0    0  4.4G  0 lvm  /
      └─fedora-swap    253:1    0  616M  0 lvm  [SWAP]
[root@localhost ~]# _
```



Opcions de particionat de Fedora-32: **Custom**

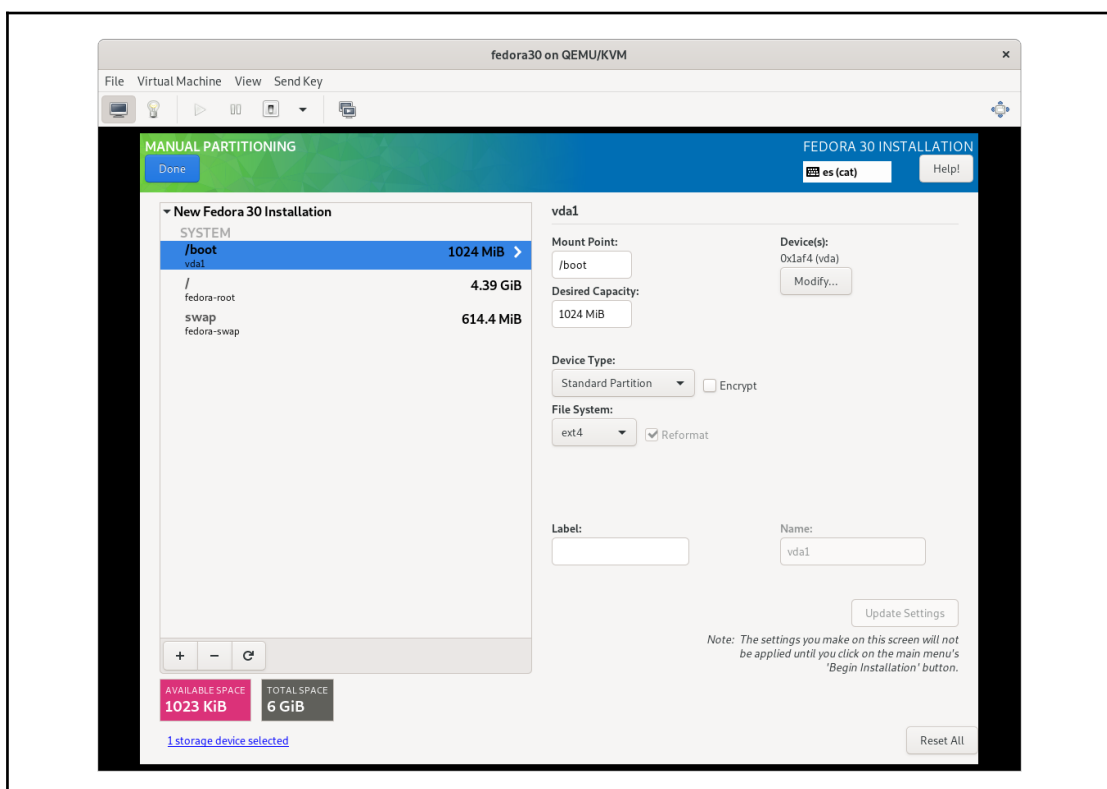
- Custom
 - Permet generar-les automàticament (clicar per observar què passa i estudiar quina és la configuració que genera)
 - Partition schema:
 - LVM Utilitza volums lògics
 - LVM Thin provisioning
 - Btrfs Sistema de fitxers Btrfs
 - Standard Partition (opció a usar per fer-les nosaltres manualment)





Exemple amb LVM + Create them automatically

- /boot (standard partition)
- / (arrel) un LVM anomenat fedora-root
- swap un LVM anomenat fedora-swap

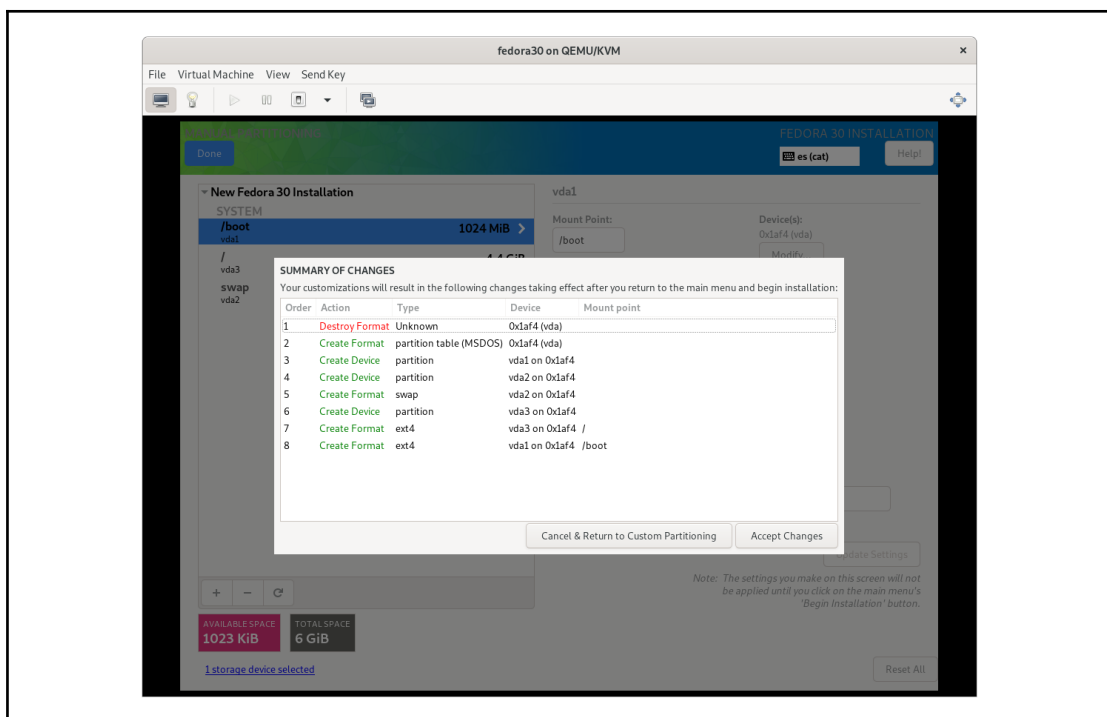
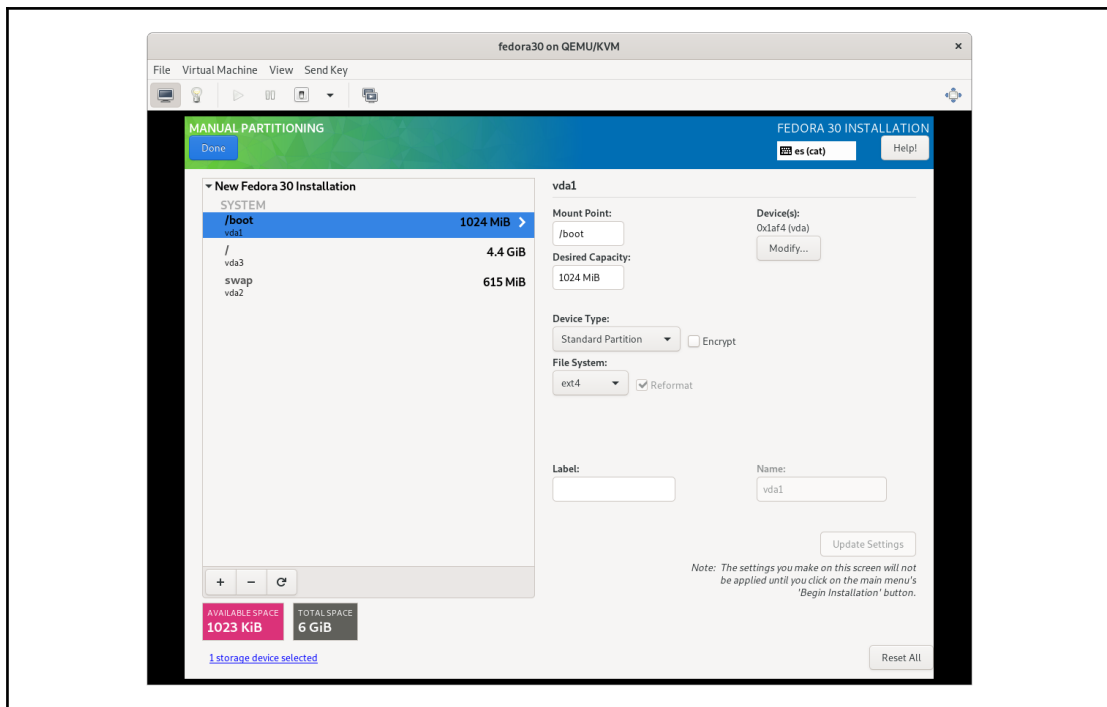


Observar les opcions de configuració manual

- **Globals:**
 - Available espace
 - Total Space
 - + (per crear noves particions)
 - - (per eliminar una partició existent)
 - @ reload storage (per recarregar la configuració de disc)
 - Reset all (elimina tot el que s'ha configurat i comença de nou)
 - Storage device selected (mostra quins dispositius hi ha)
- **vda1**
 - Mount Point
 - Desired Capacity
 - Device Type (+ Encrypt)
 - File system
 - Label
- **Update settings** (desar els canvis)
- **Tricks:**
 - Observeu que cal modificar alguna cosa per poder fer el update settings.
 - En tot moment podeu fer el rescan i tornar a començar o el reset all.
 - Practiqueu diferents tipus de particionat sense fer la instal·lació realment. Fins i tot podeu desar-los formatant el disc (si és virtual, amb els reals de l'aula no!). Podem fer els accept changes i després NO fer la instal·lació. El disc dur no es formata realment fins al procés d'instal·lació.
 - Observar des de la consola (F2 de Send key) les particions amb les ordres fdisk -l, lsblk, blkid (F6 per tornar a la consola gràfica amb Send key).

Exemple amb Standard Partition + Create them automatically

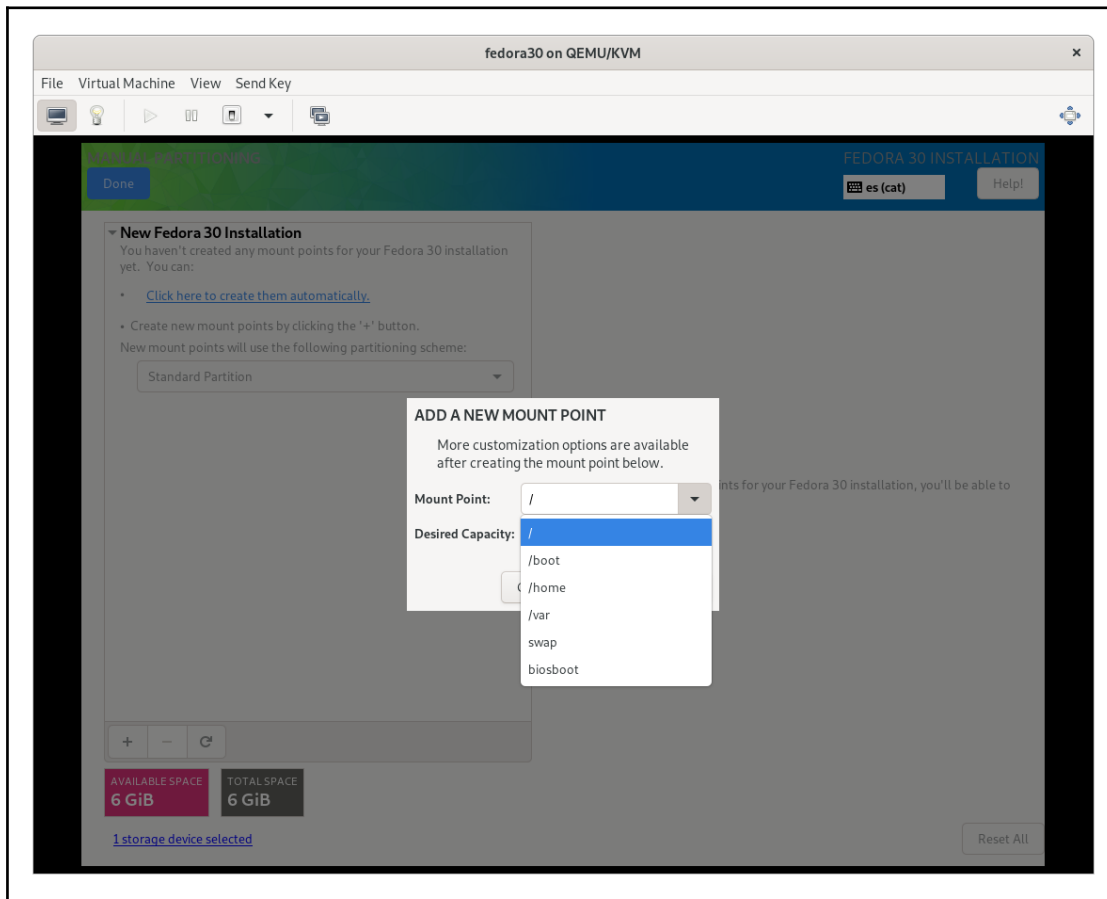
- /boot (/dev/vda1)
- / (arrel) (/dev/vda3)
- swap (/dev/vda2)



Exemple de creació de particions manualment (gràfic)

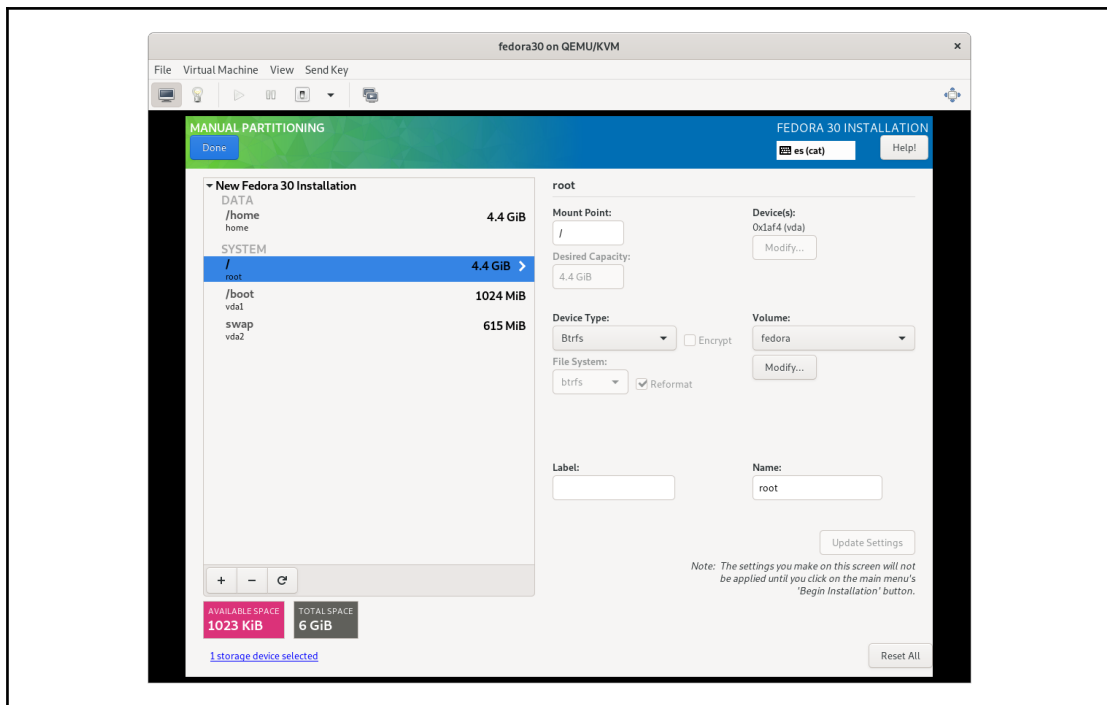
- Si es selecciona que es vol fer una configuració Custom i les particions de tipus Standard Partition es poden usar els botons + i – per crear particions.
- Observar que per crear una nova partició cal indicar:
 - Mount point
 - Capacity
- El Mount Point es pot indicar manualment però n'hi ha uns quants de ja predefinits:
 - /

- /boot
- /home
- /var
- swap
- Biosboot
- També es pot anar a la consola (F2 amb Send key) i crear allà manualment les particions. Un cop fetes en tornar a la sessió gràfica (F6) cal fer el Rescan del disc.



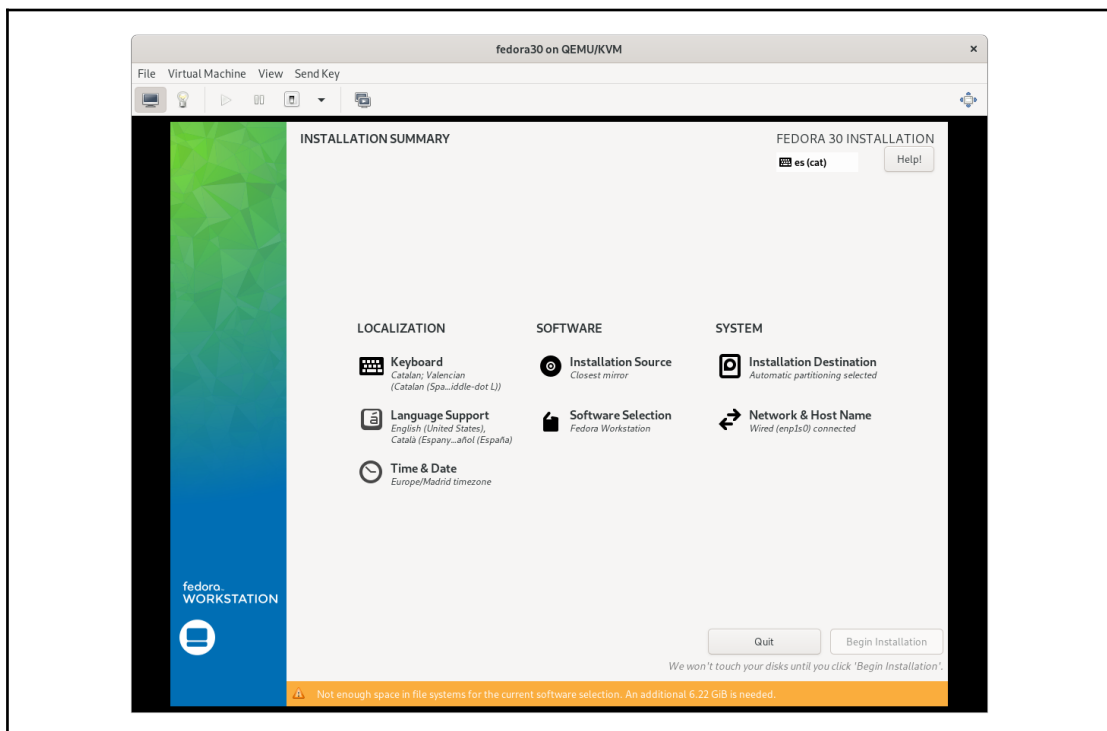
Exemple de creació de particions Btrfs

- /home de tipus Btrfs
- / (arrel) de tipus Btrfs
- /boot (standard partition)
- swap (standard partition)



Exemple d'espai insuficient per a la instal·lació

- En aquest exemple podem veure que en intentar fer una instal·lació Workstation en lloc de minimal es queixa de que li falten 6 GiB més (en total en requereix +12 GiB).

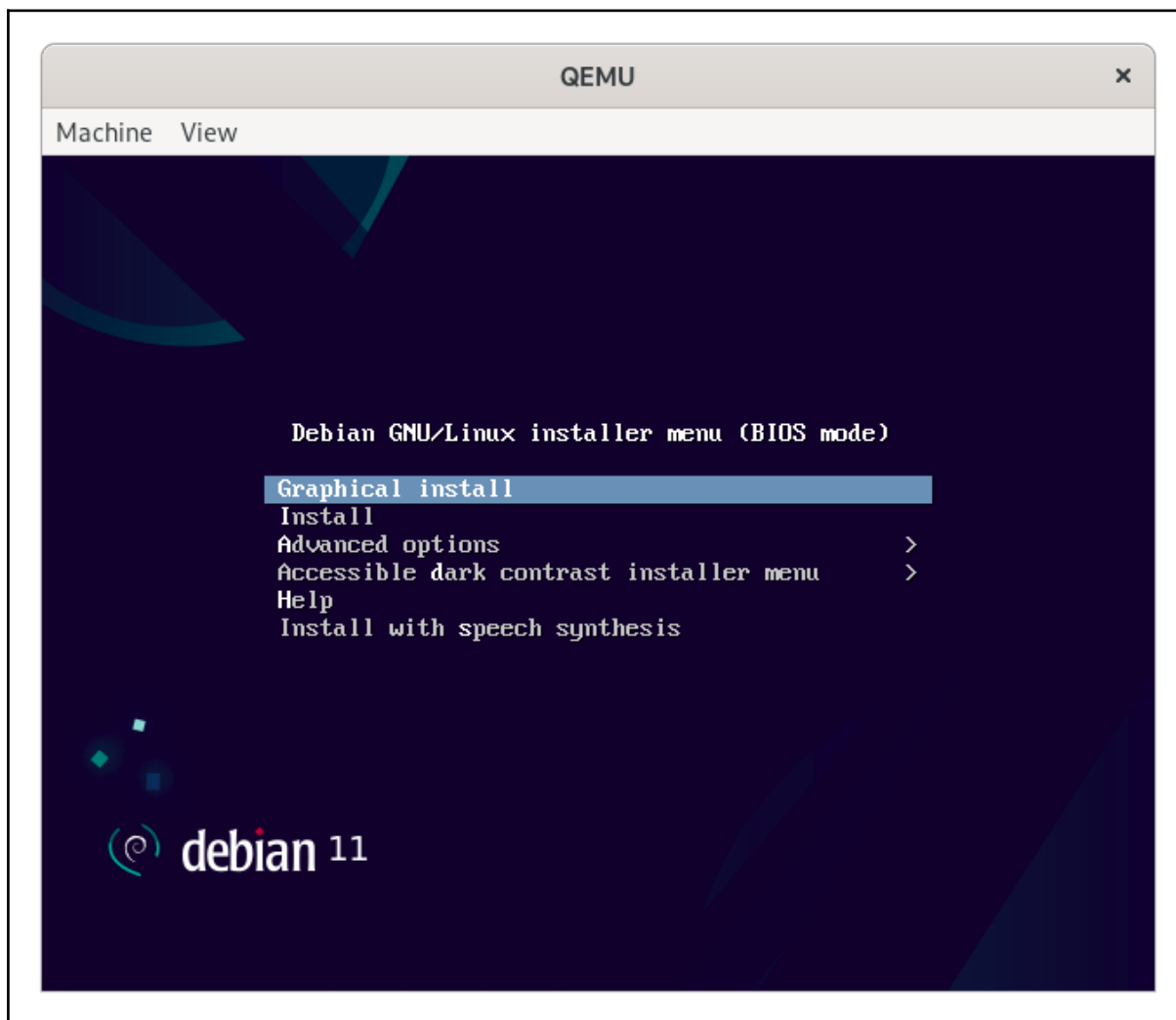


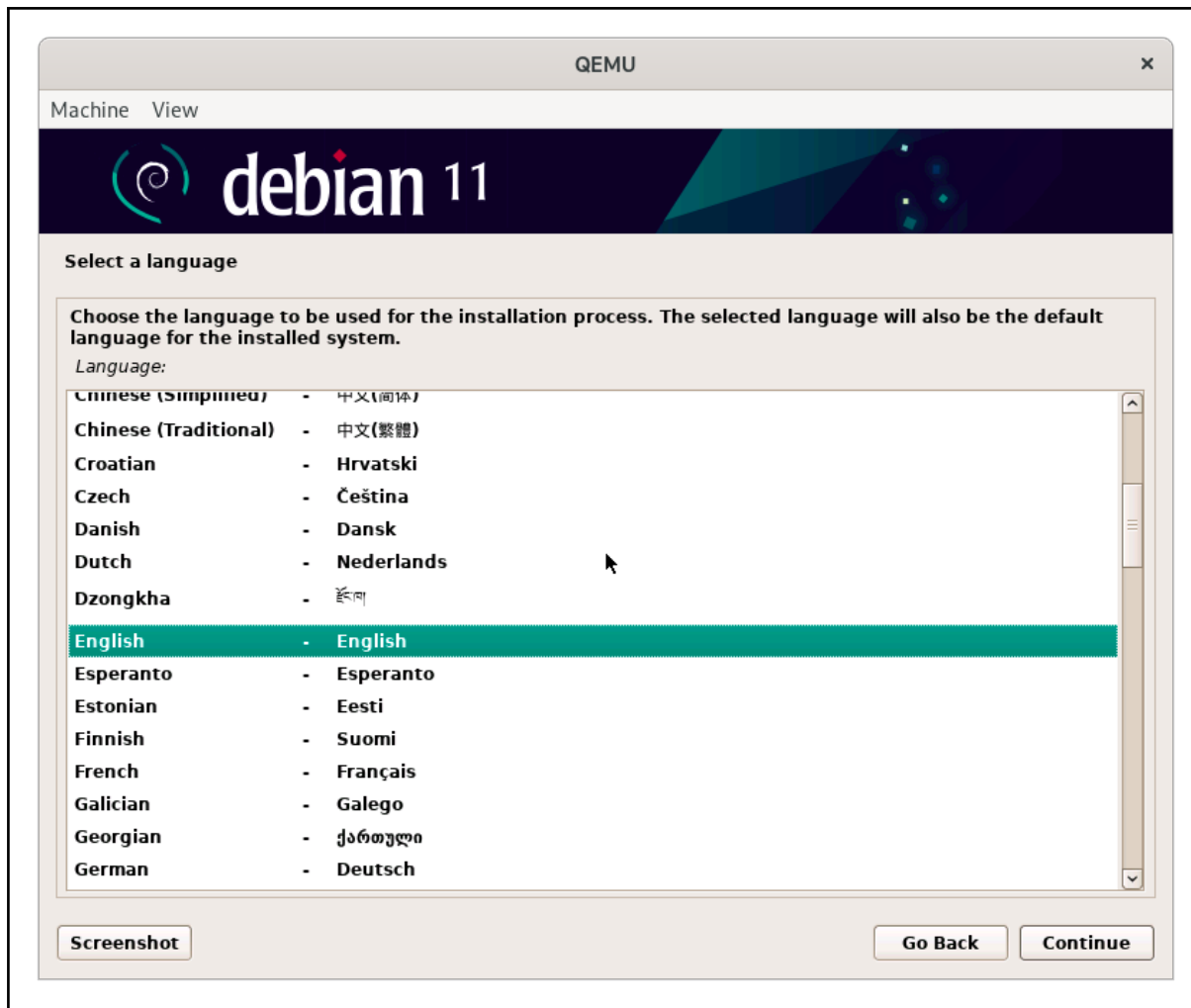
Debian-11

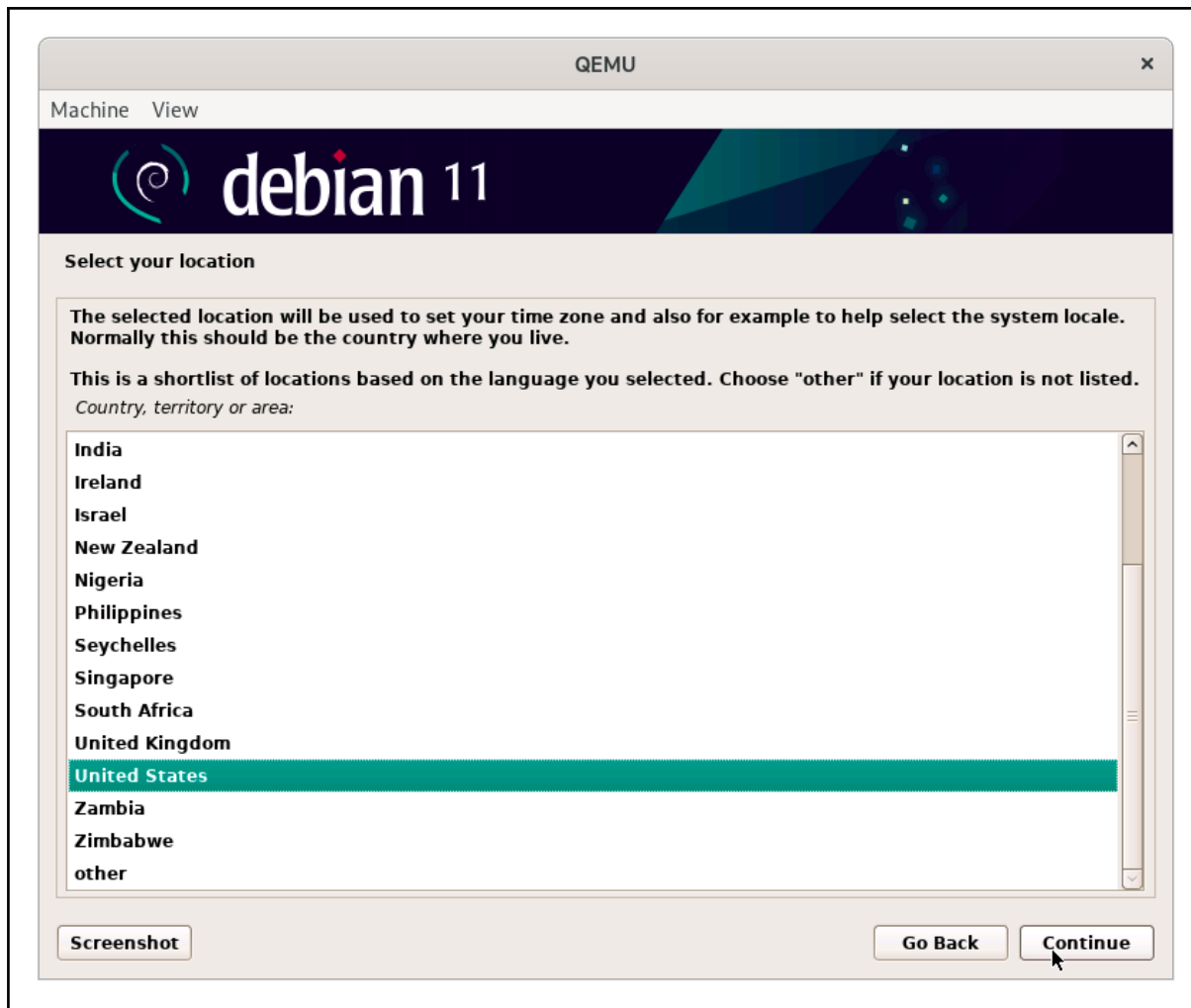
Debian-11 Bullseye

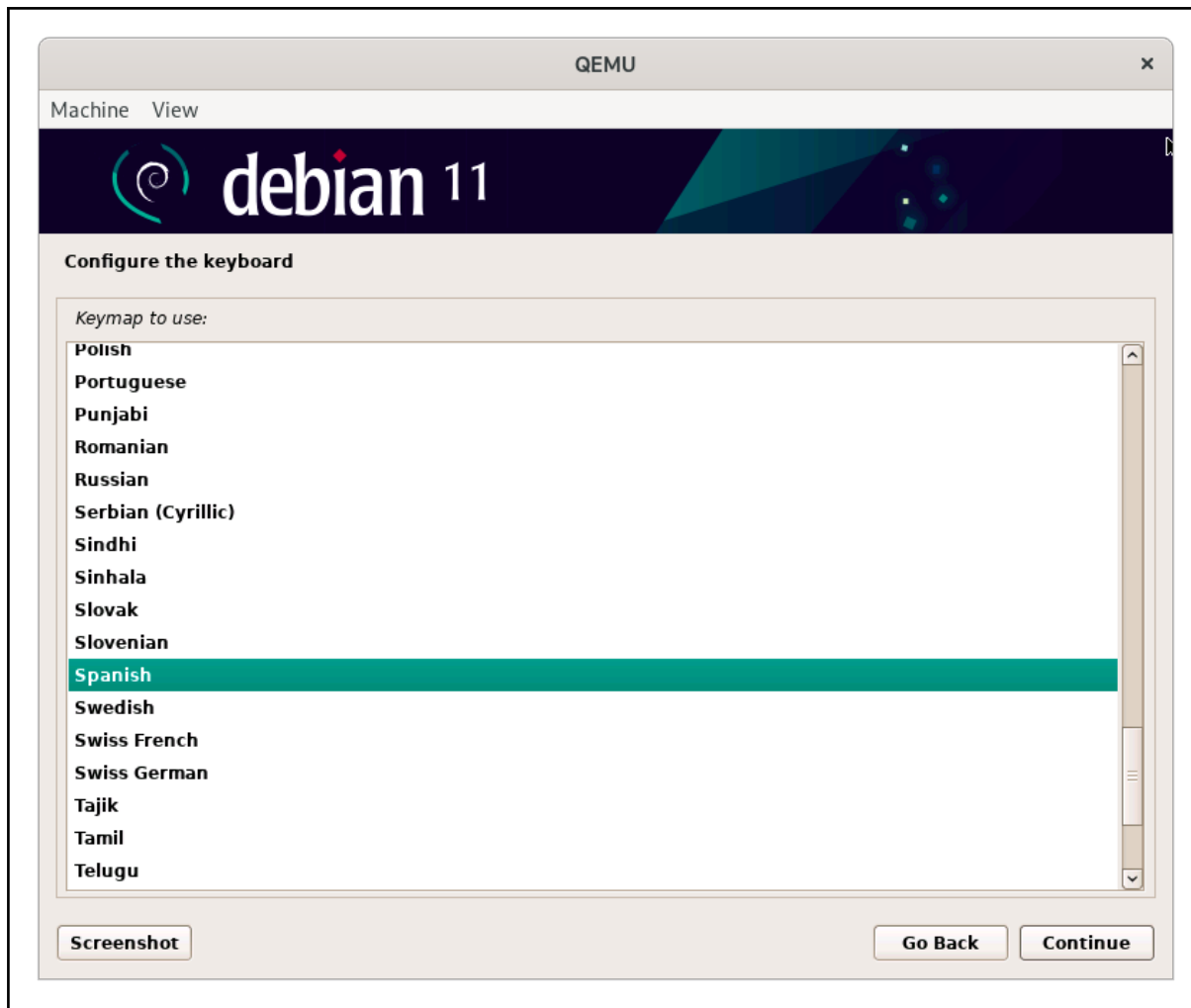
- Practicar el procediment d'instal·lació de Debian-11
- Observar les opcions de particionat automàtiques i manuals.

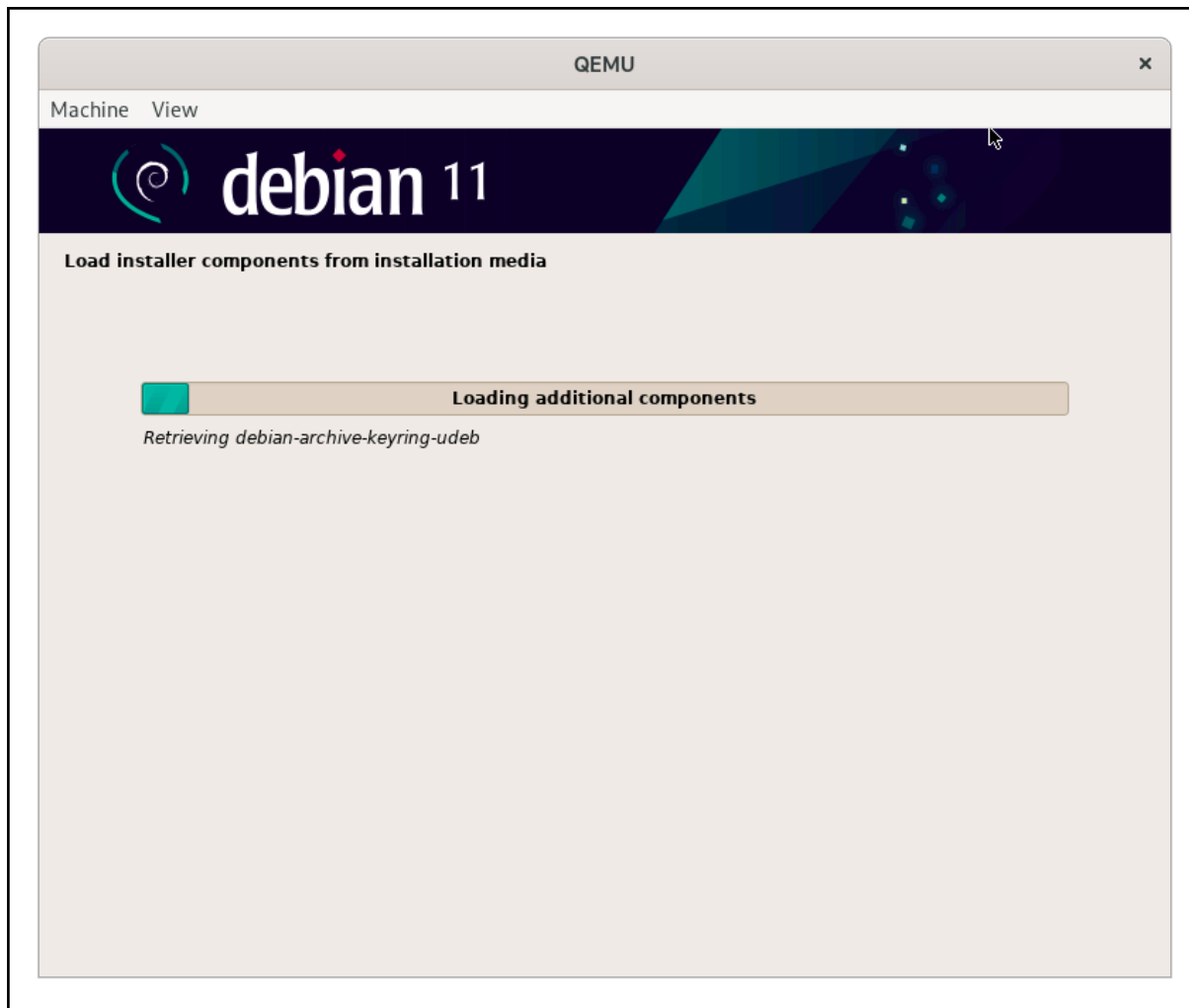
```
$ qemu-system-x86_64 -m 4096 -hda disk.img -cdrom ../debian-11.1.0-amd64-netinst.iso -boot once=d
```

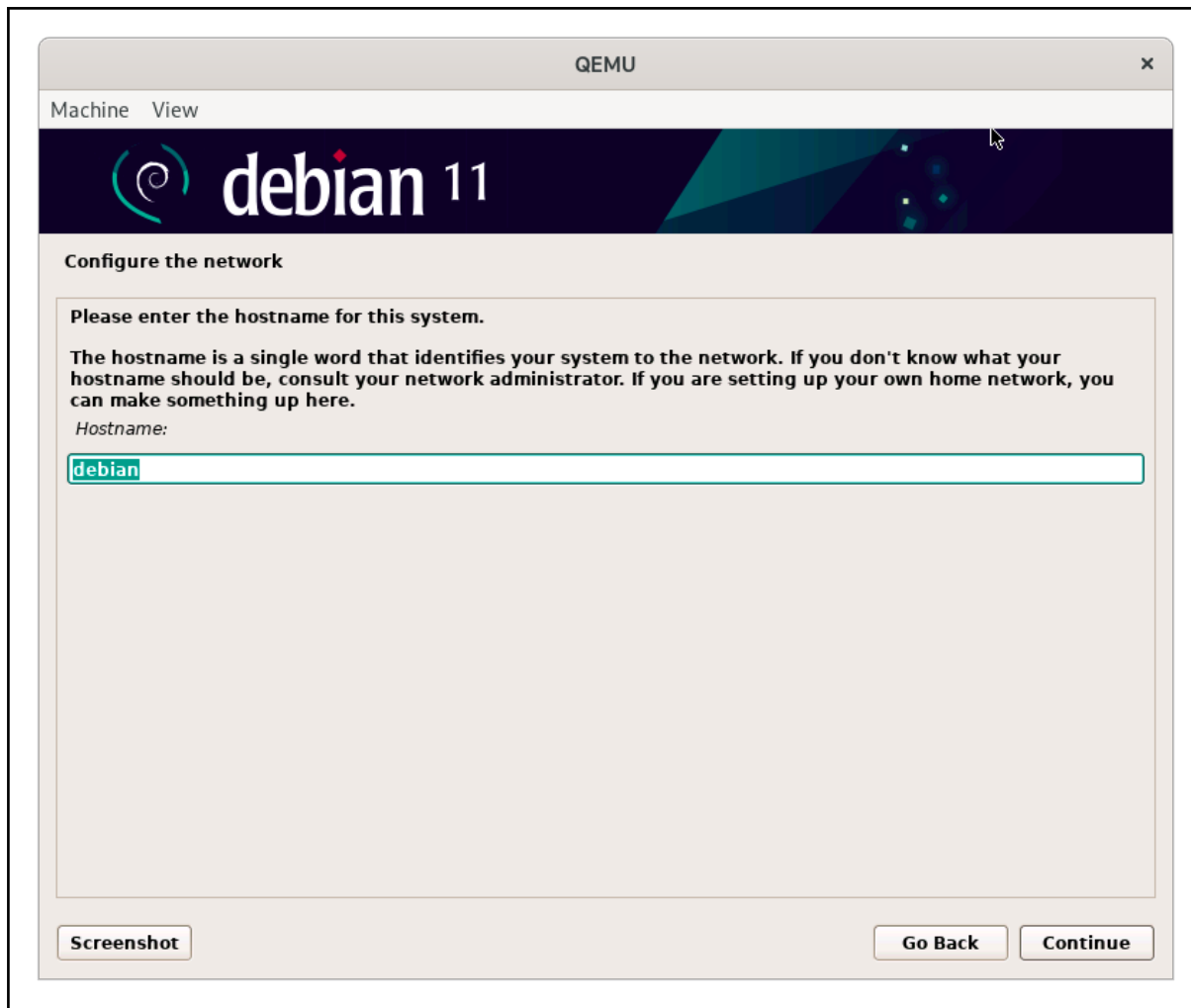


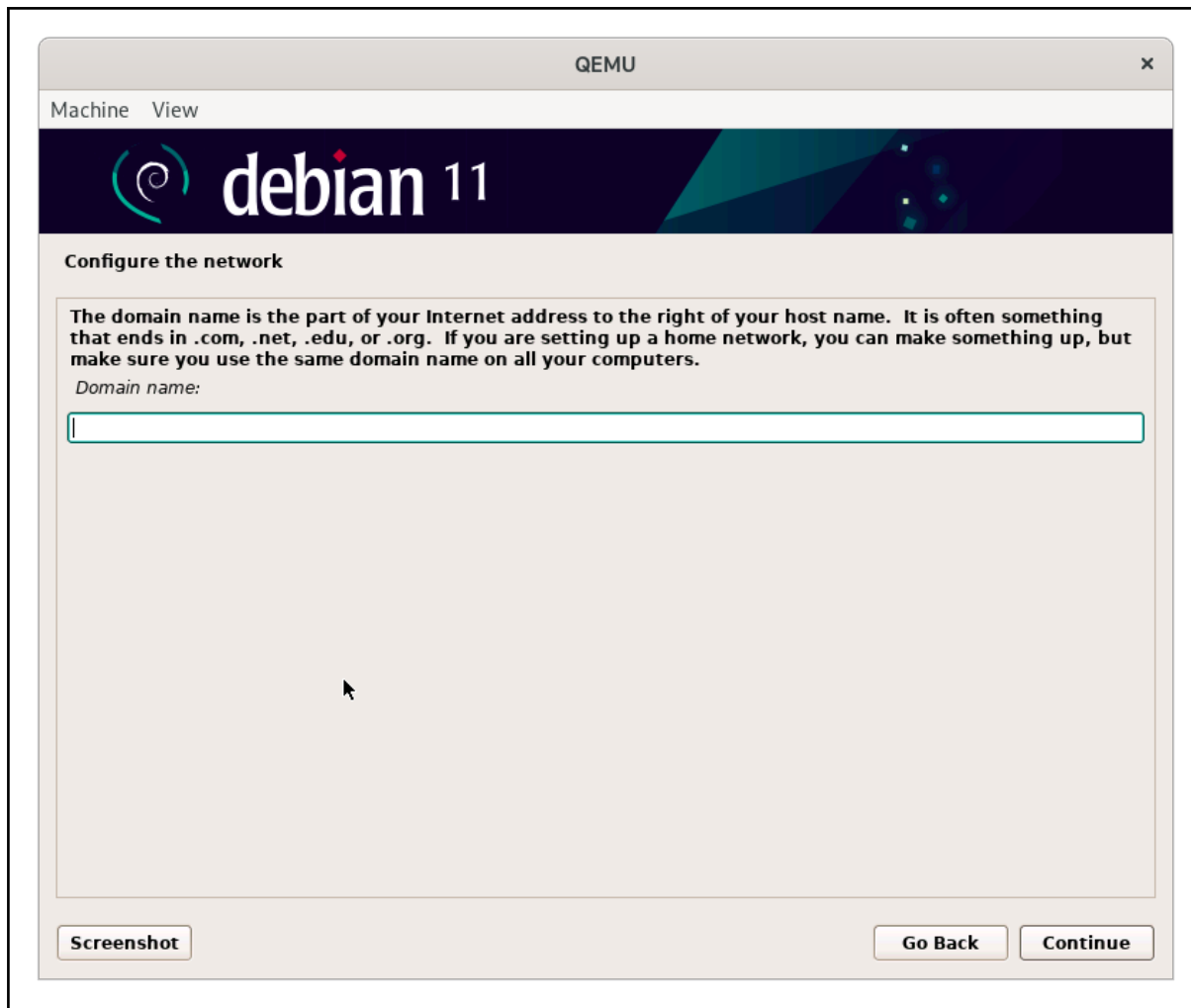

















QEMU

MachineView



Set up users and passwords

A good password will contain a mixture of letters, numbers and punctuation and should be changed at regular intervals.
Choose a password for the new user:

•••••

☐ Show Password in Clear

Please enter the same user password again to verify you have typed it correctly.
Re-enter password to verify:

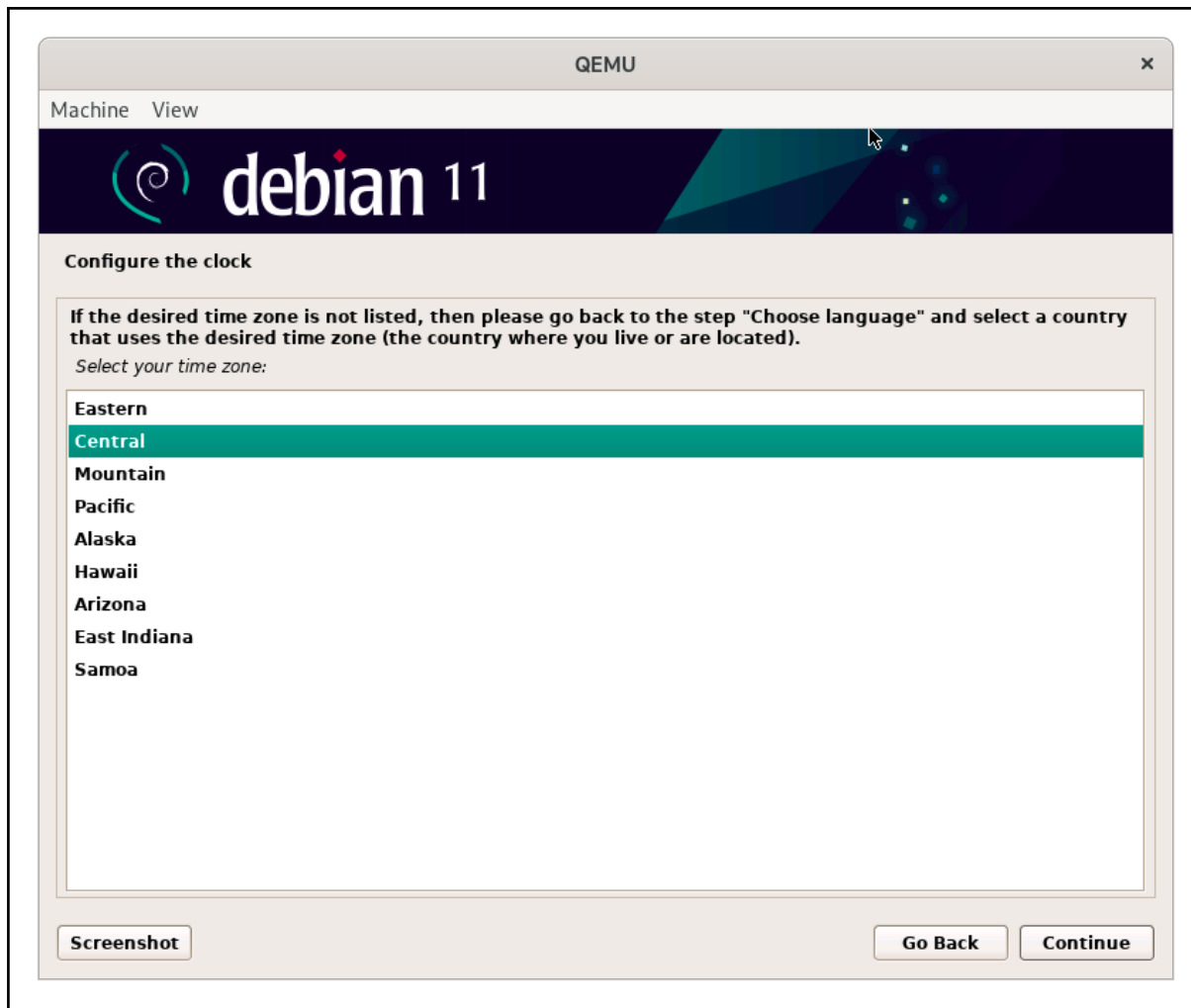
•••••

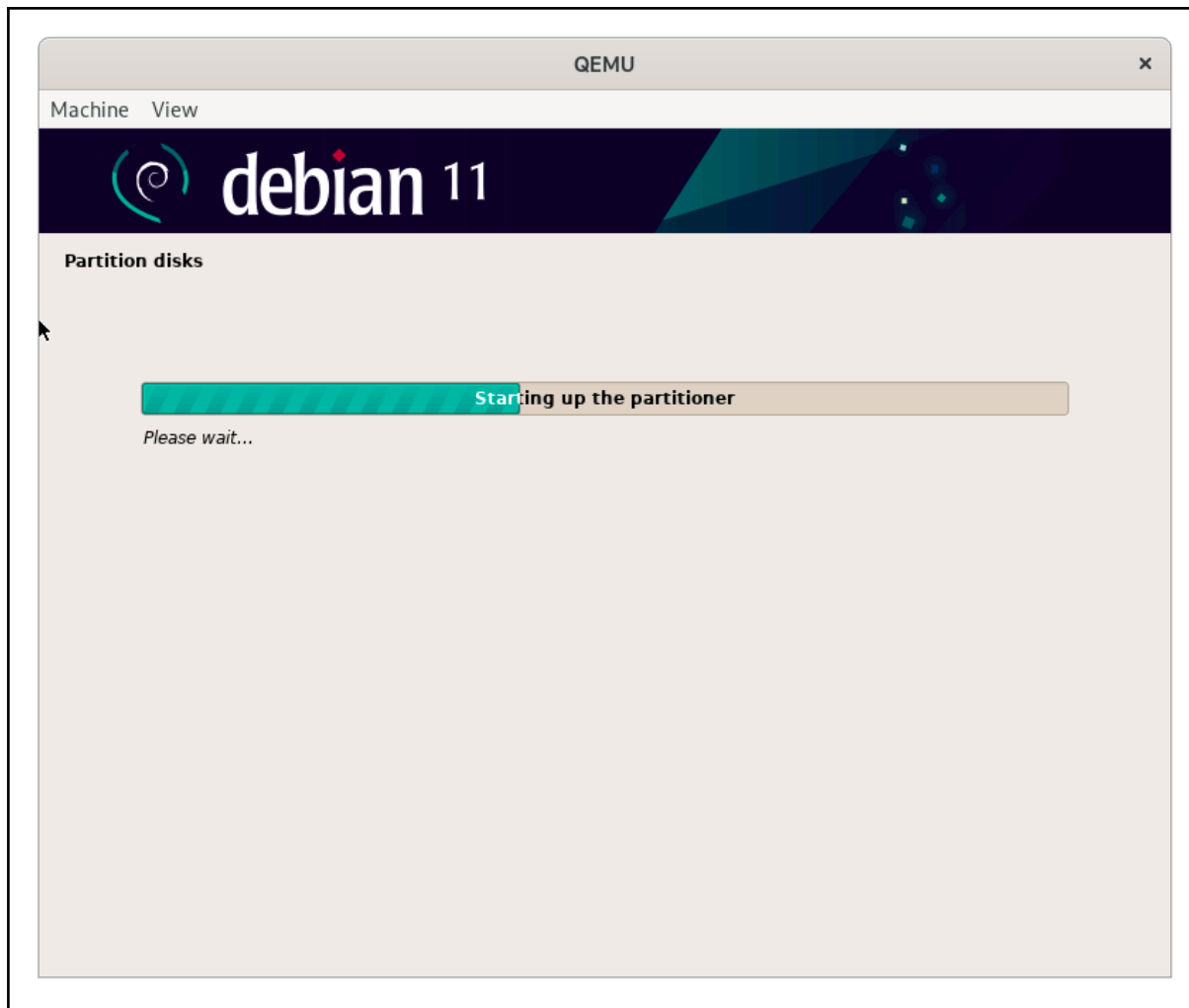
☐ Show Password in Clear

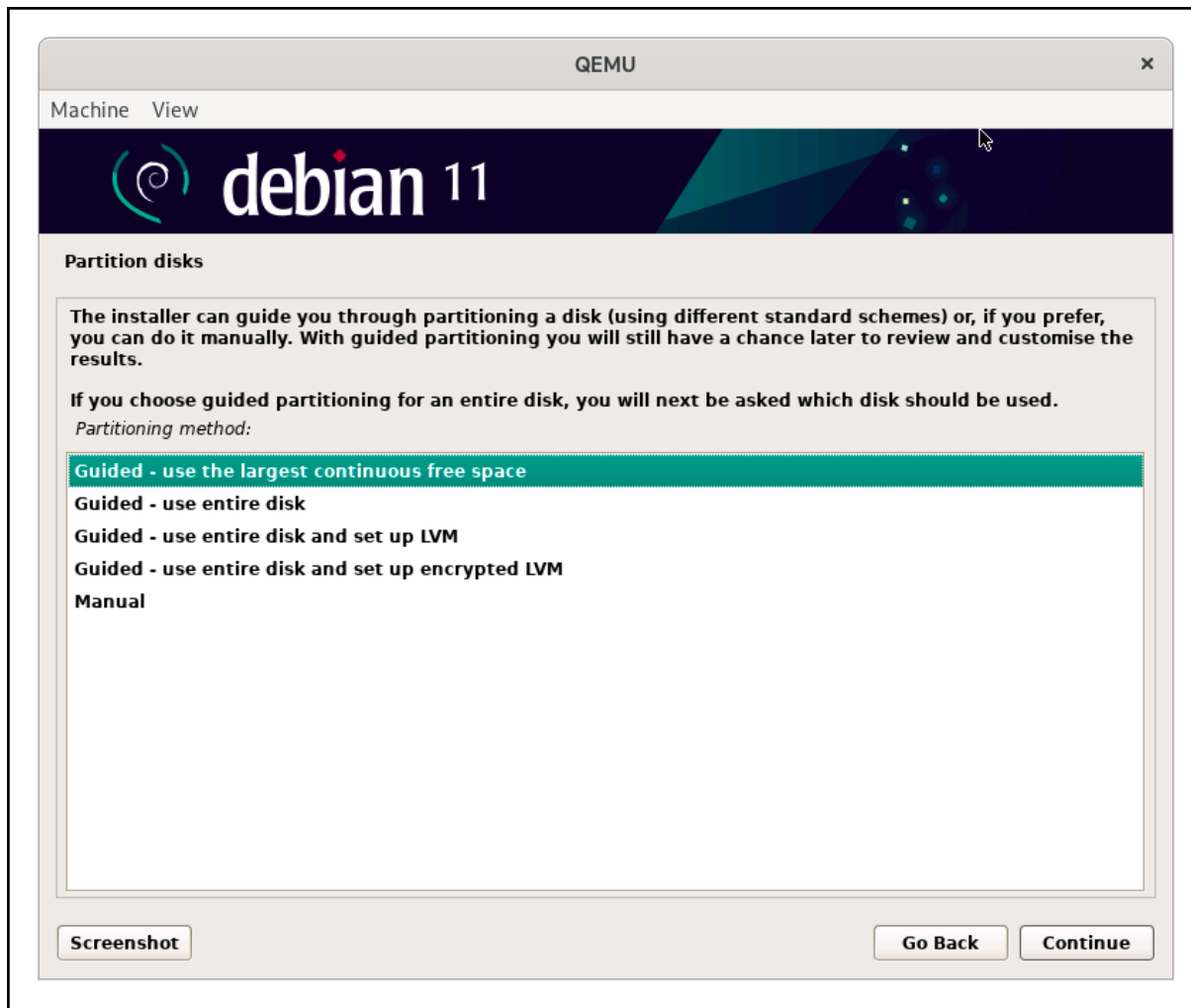
Screenshot

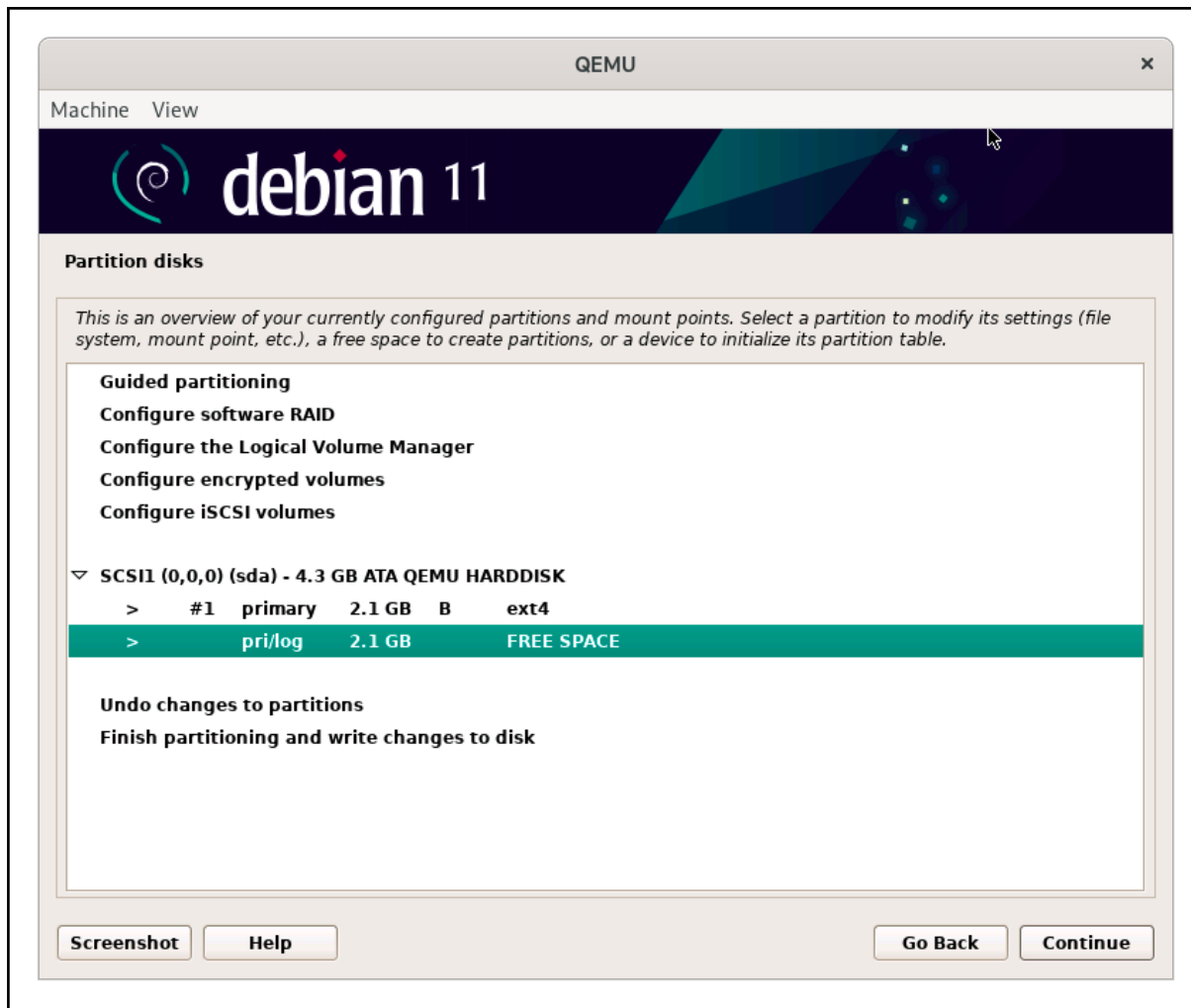
Go Back

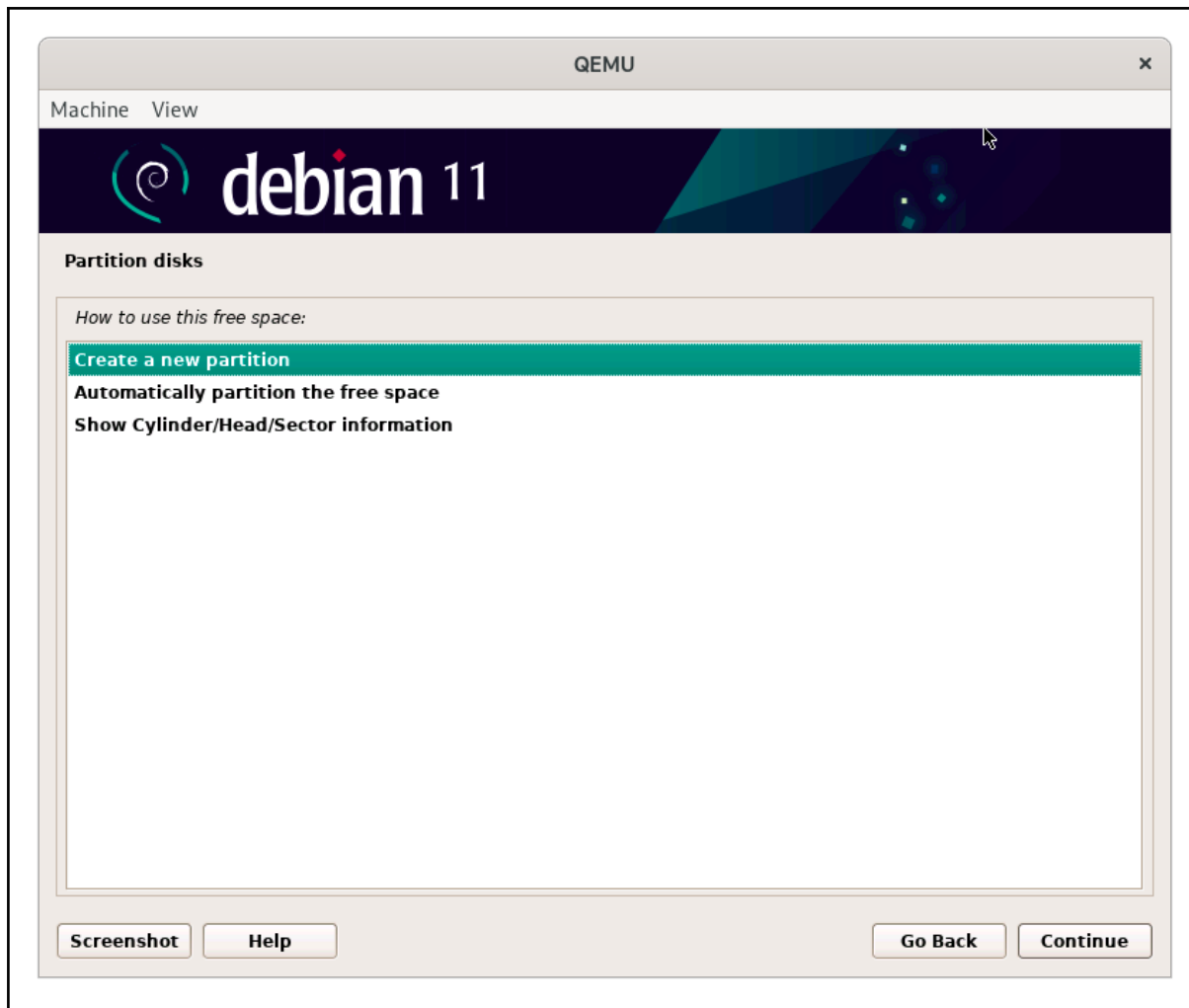
Continue



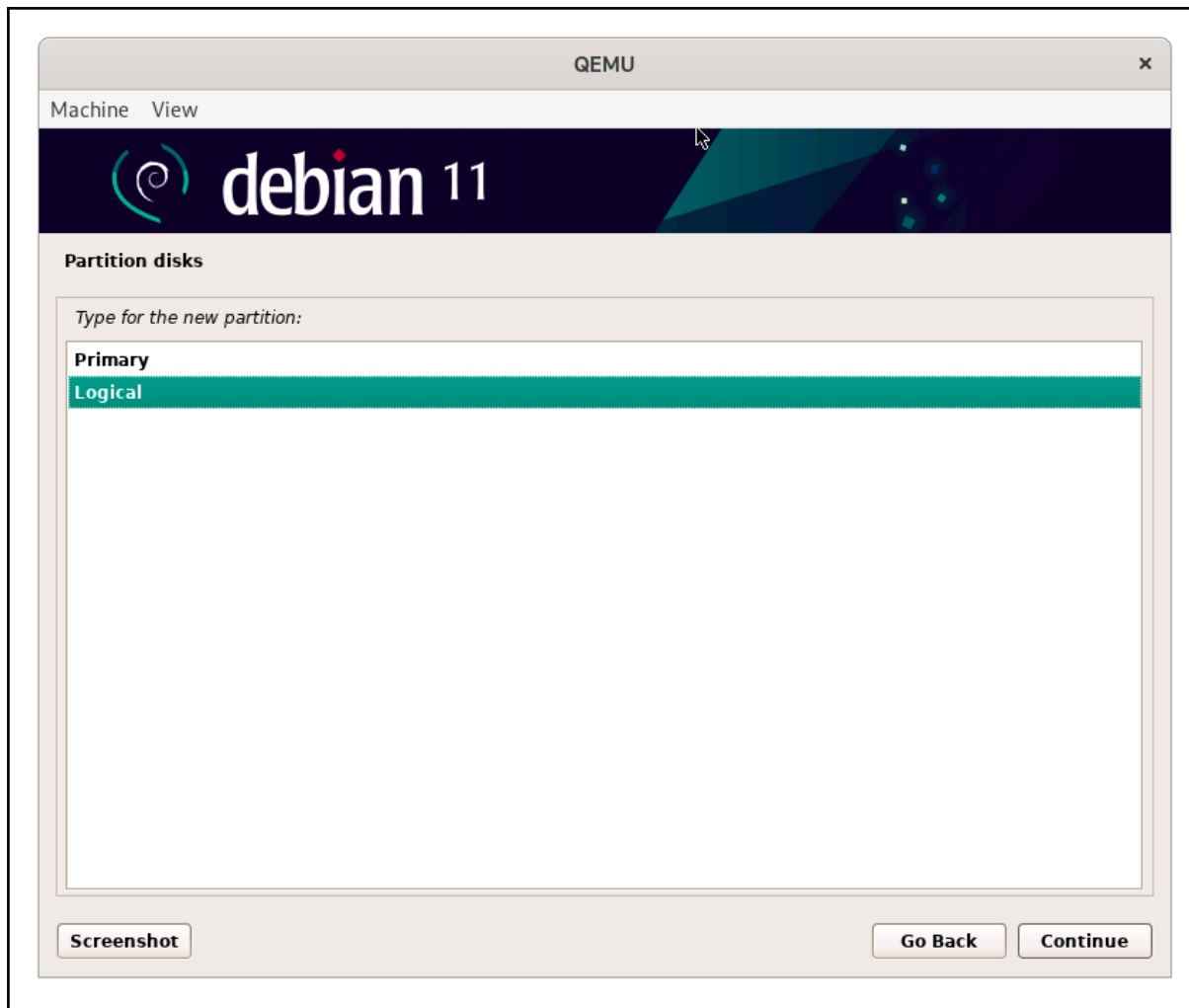


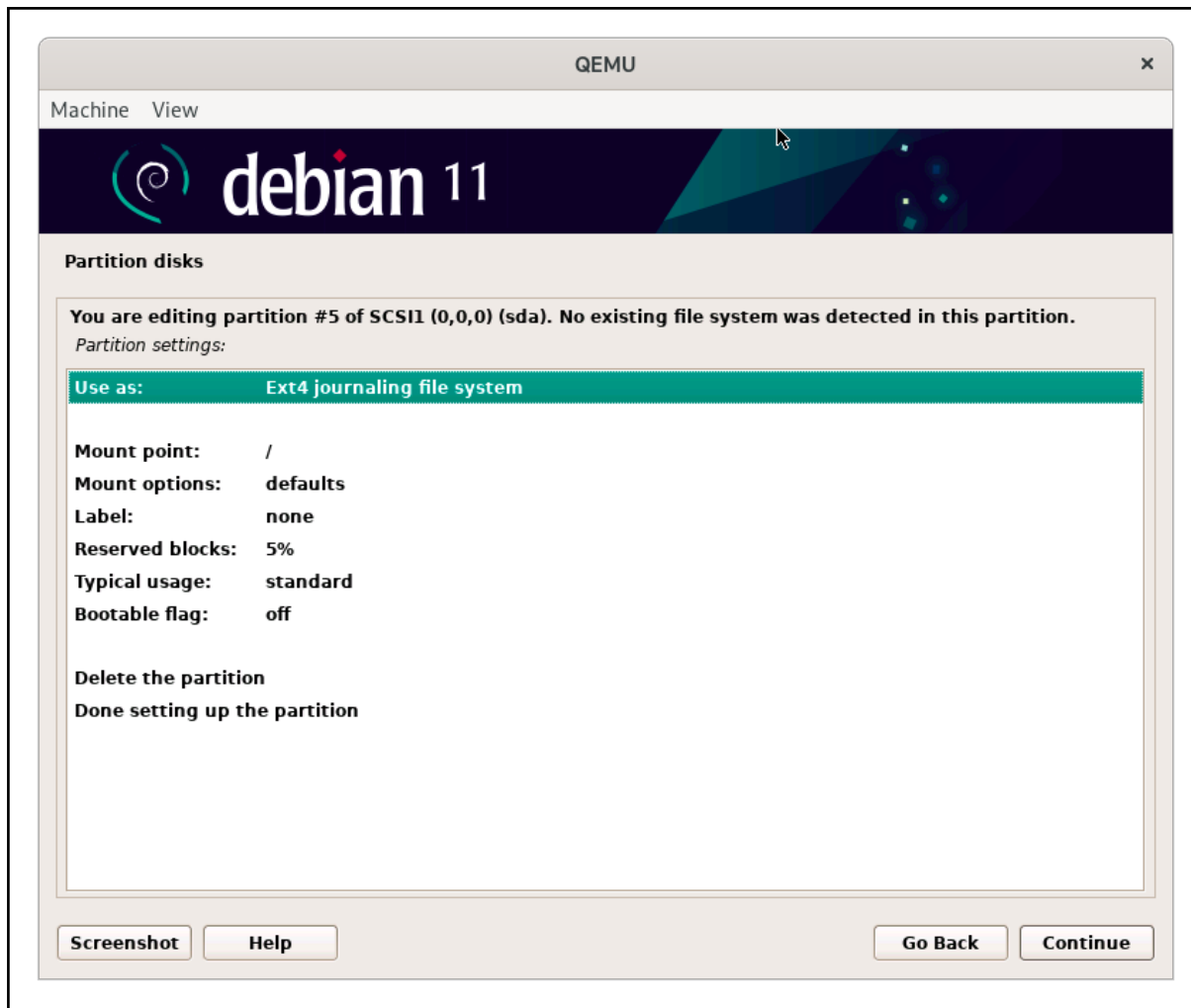


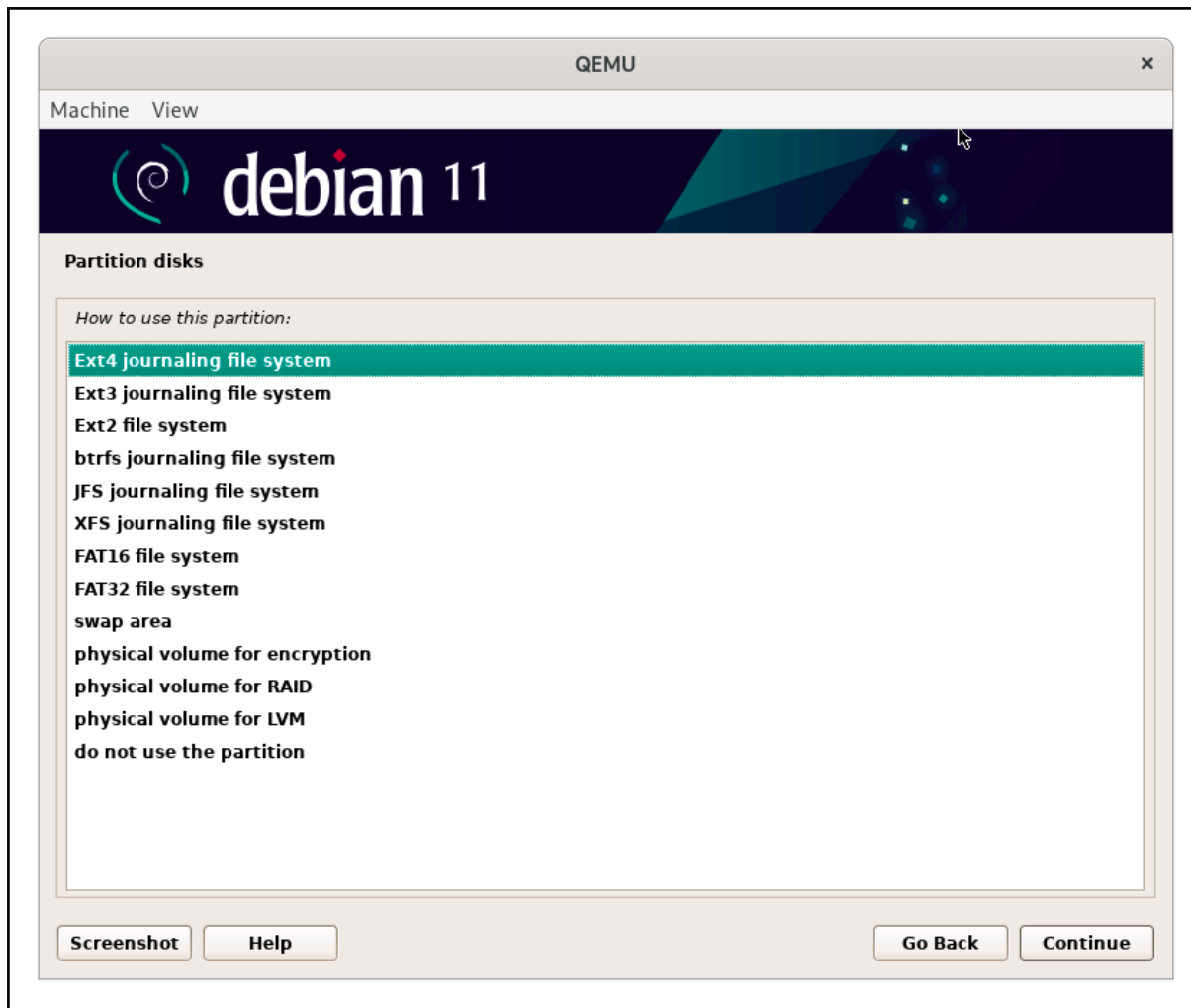


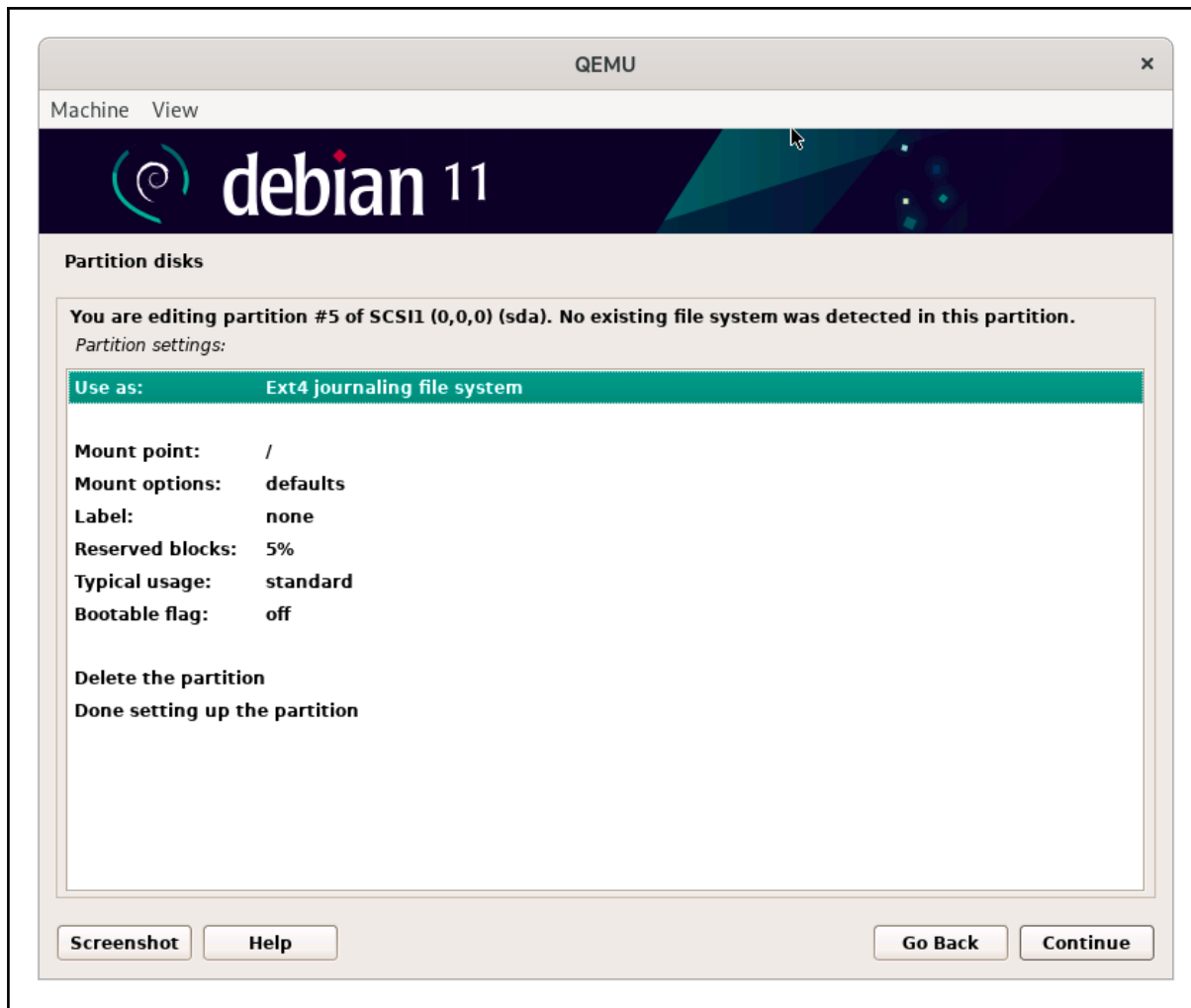


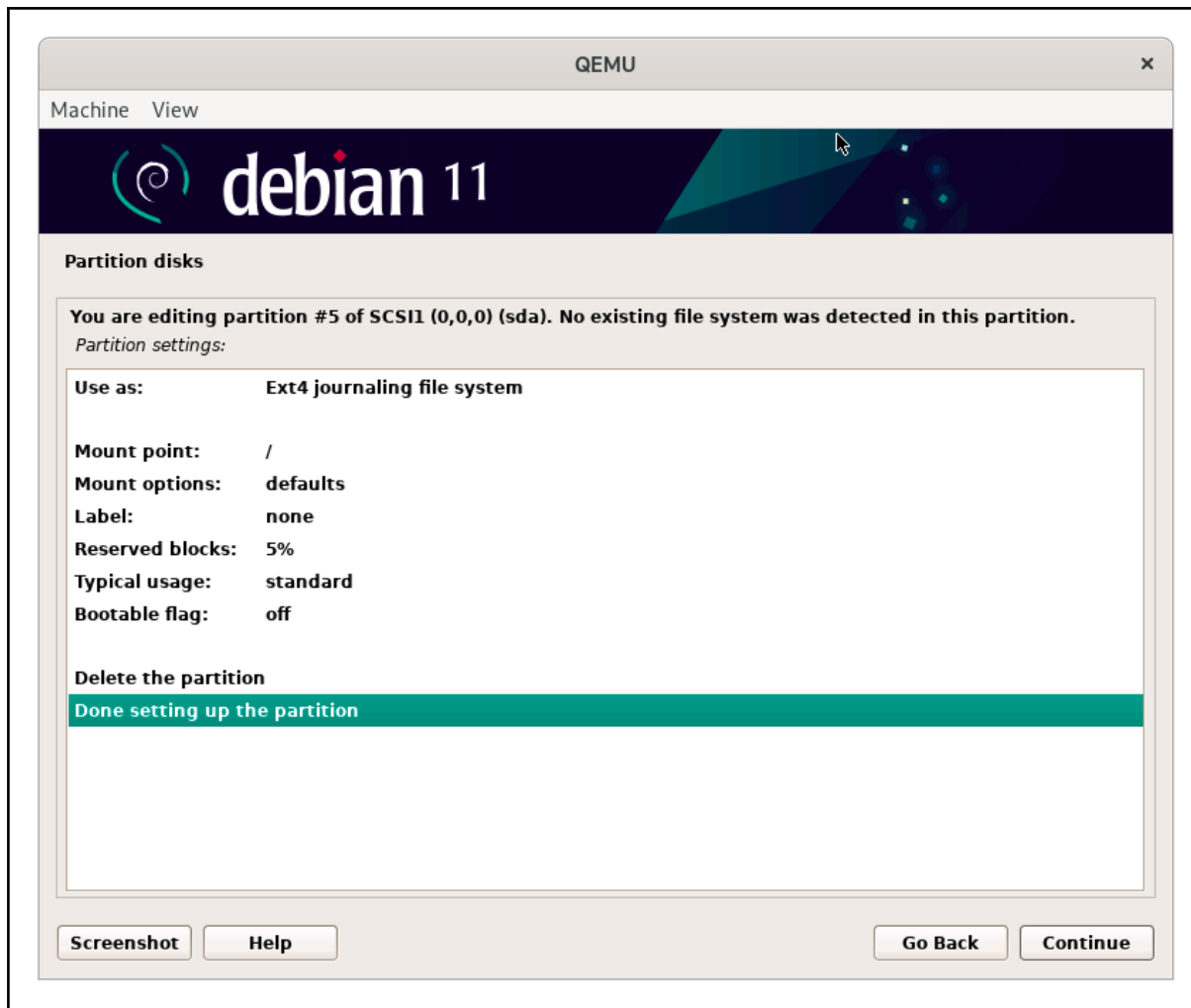


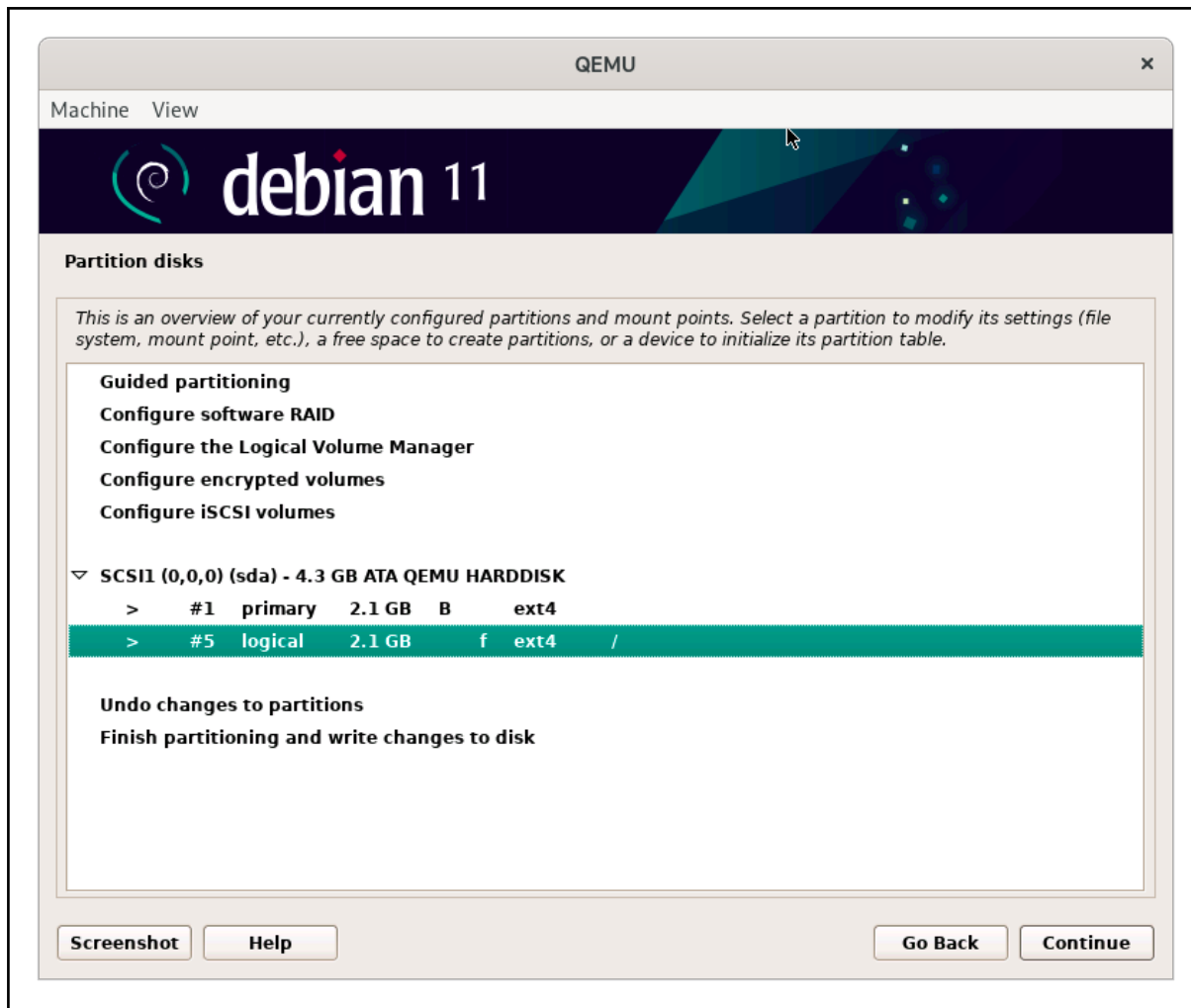


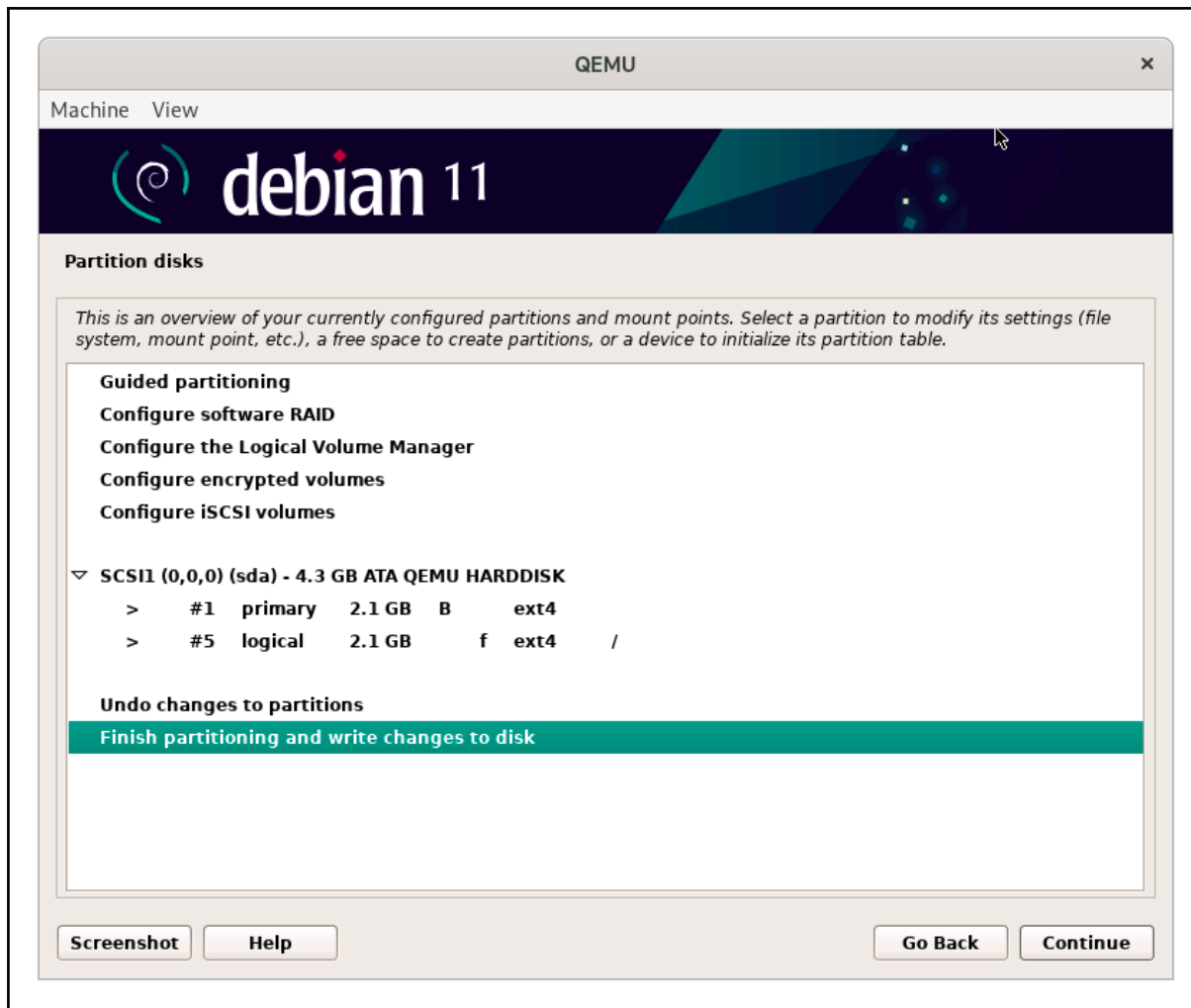






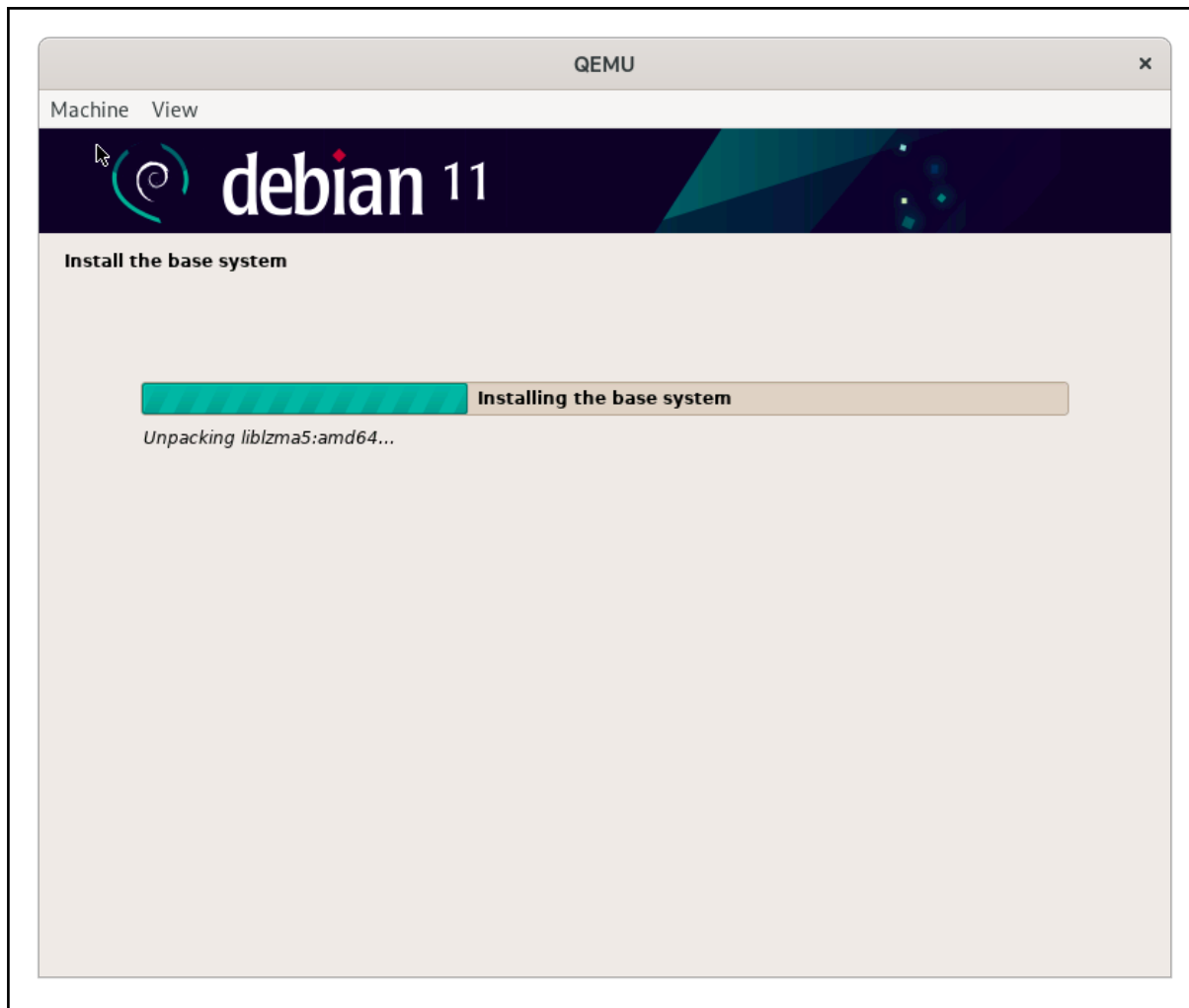




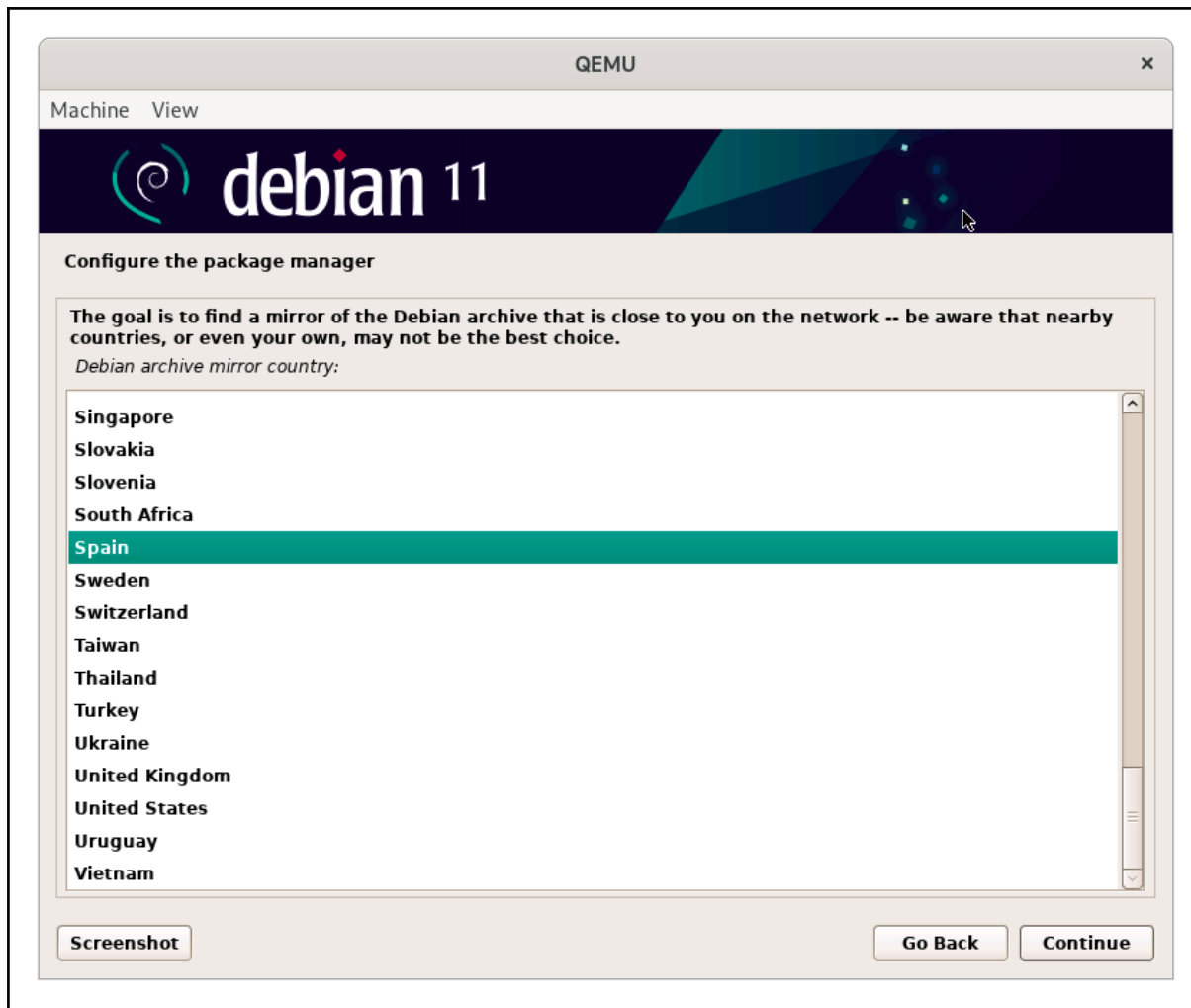








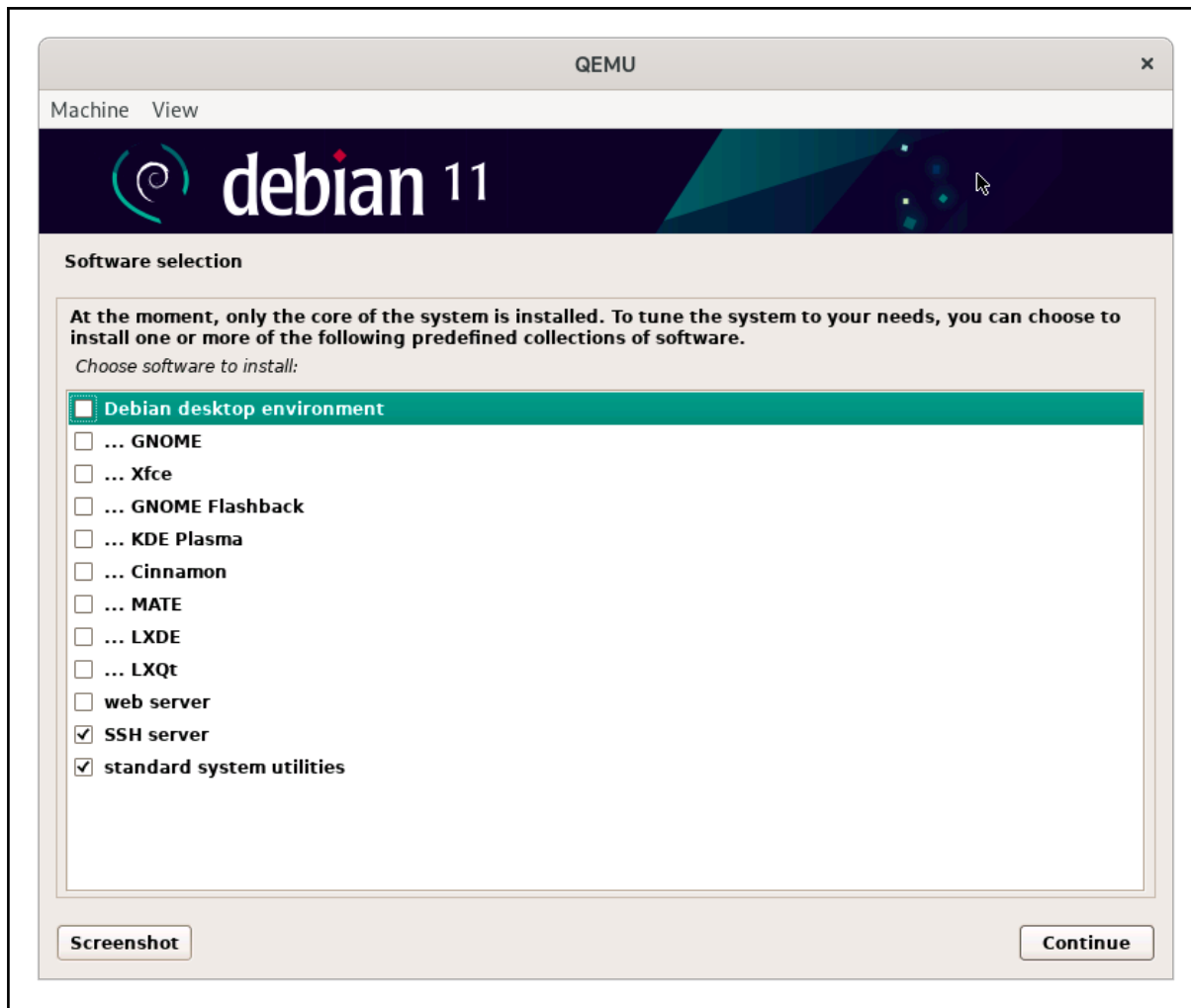






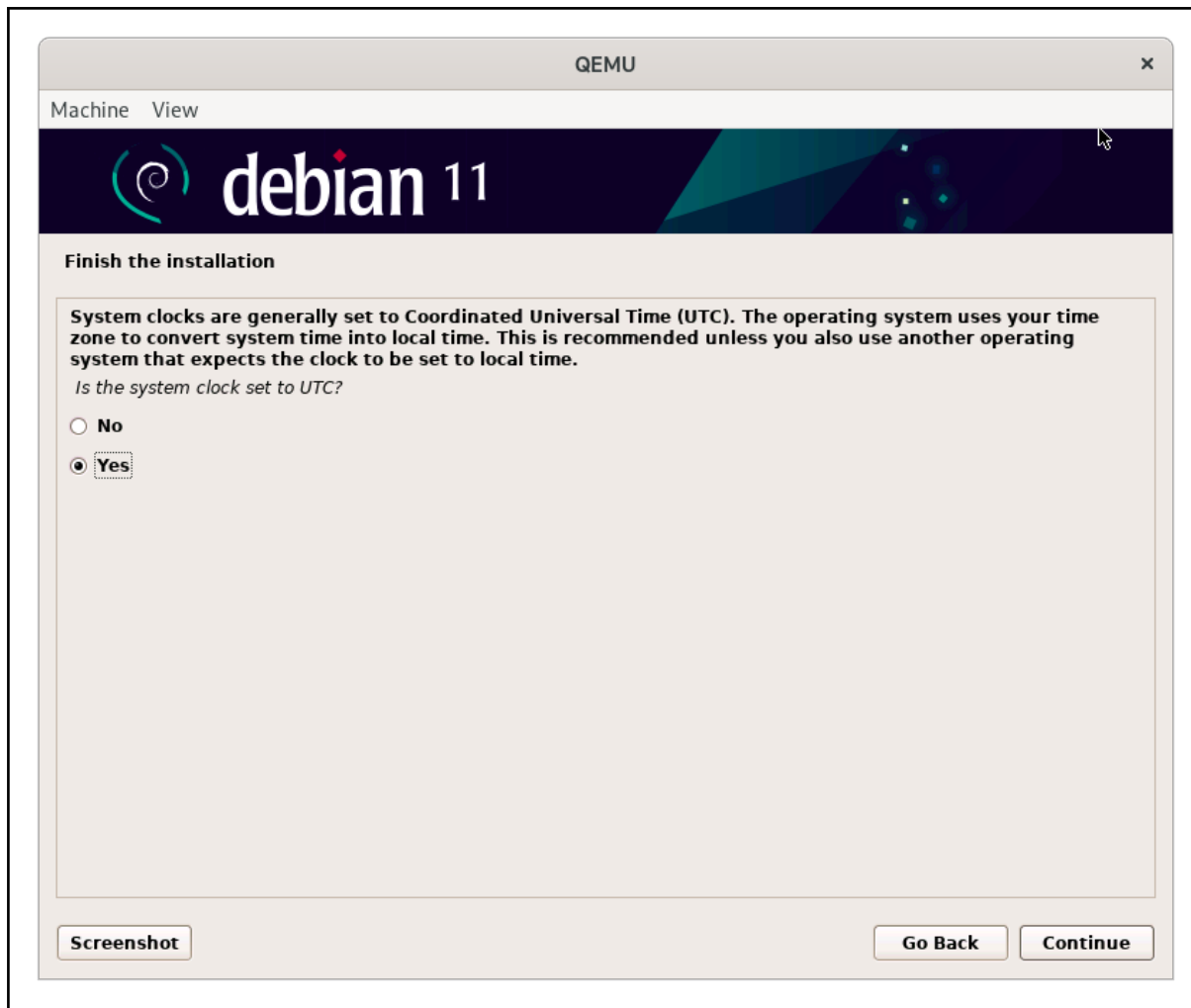




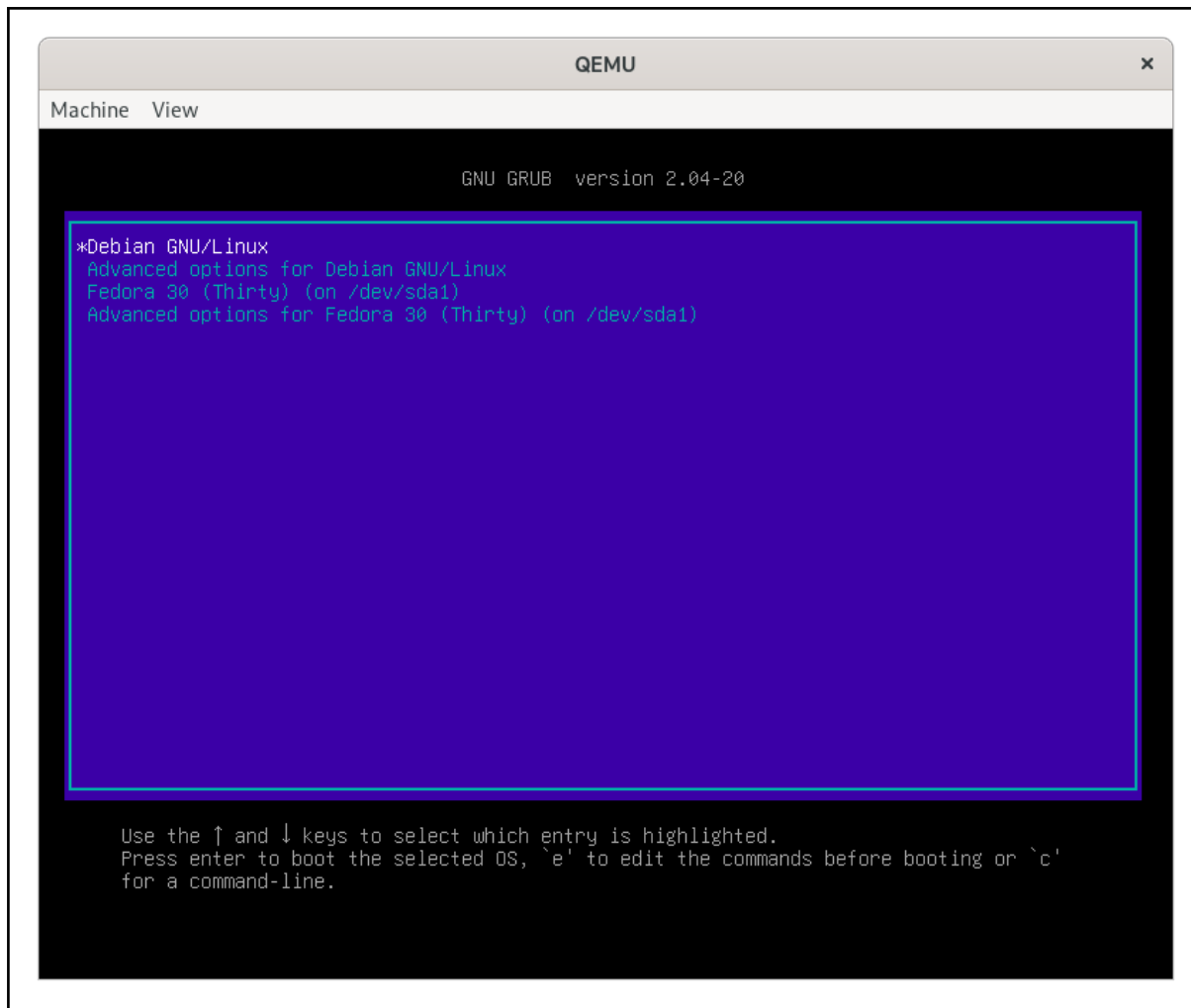












```
QEMU
Machine View

Debian GNU/Linux 11 debian tty1

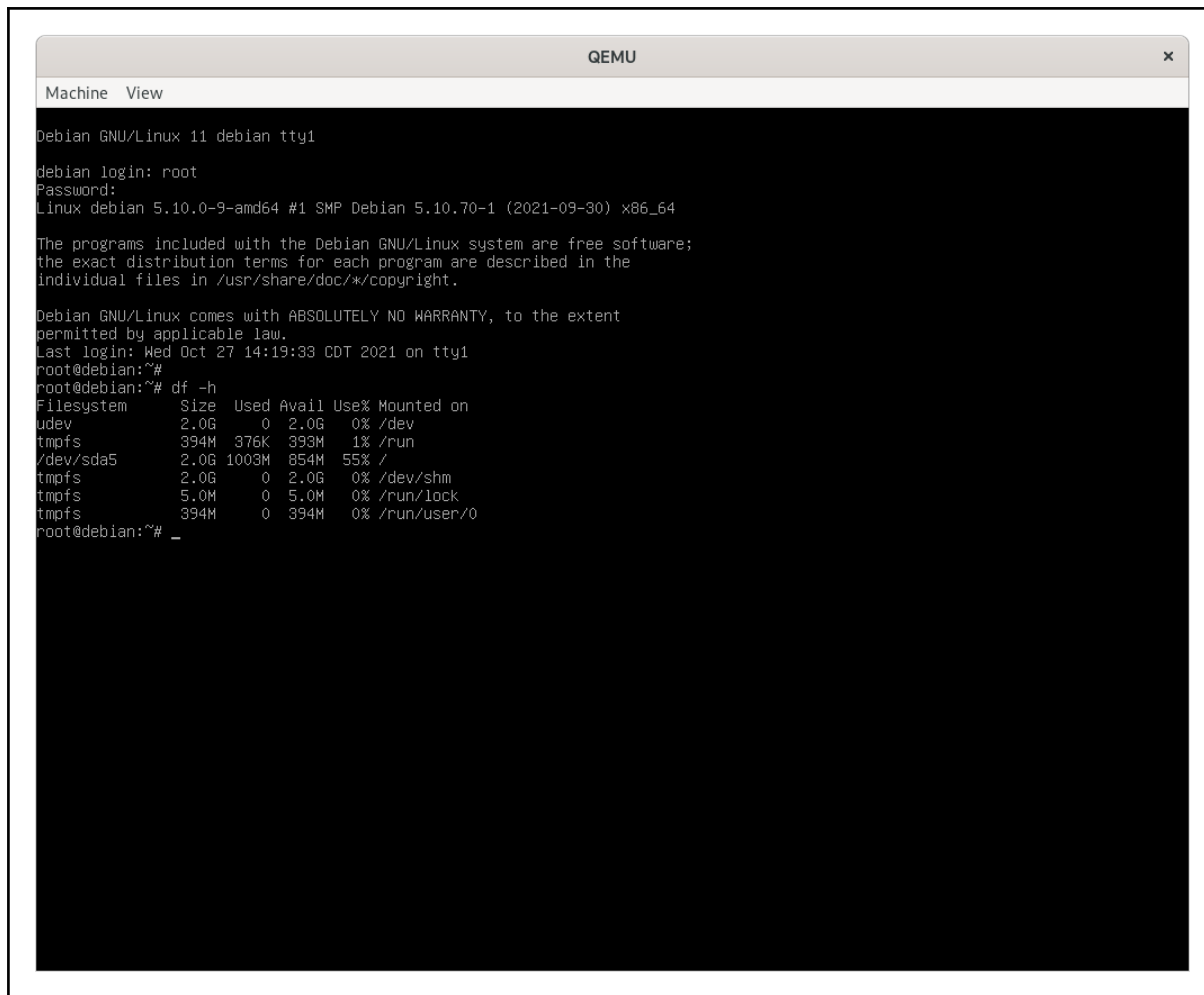
debian login: root
Password:
Linux debian 5.10.0-9-amd64 #1 SMP Debian 5.10.70-1 (2021-09-30) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Oct 27 14:19:27 CDT 2021 on tty1
root@debian:~#
root@debian:~# fdisk -l
[ 93.447578] blk_update_request: I/O error, dev fd0, sector 0 op 0x0:(READ) flags 0x0 phys_seg 1 prio class 0
[ 93.486694] blk_update_request: I/O error, dev fd0, sector 0 op 0x0:(READ) flags 0x0 phys_seg 1 prio class 0

Disk /dev/sda: 4 GiB, 4294967296 bytes, 8388608 sectors
Disk model: QEMU HARDDISK
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: 0x42d65dc9

Device      Boot    Start        End    Sectors    Size Id Type
/dev/sda1   *            2048    4196351    4194304     2G 83 Linux
/dev/sda2                4198398    8386559    4188162     2G  5 Extended
/dev/sda5                4198400    8386559    4188160     2G 83 Linux
root@debian:~# _
```



Ubuntu 20 <pendent>
