Virtualization Labs

Carles Mateu - carles.mateu@udl.cat

1. Installation

We will install virtualbox. If using Linux **do not use VirtualBox from your distribution repositories**. Got to http://virtualbox.org, download it, and install it. And download, also, the extensions package.

We can have several Linux variants to download, including adding the VirtualBox repo to our system:

VirtualBox 6.1.14 for Linux

- Oracle Linux 8 / Red Hat Enterprise Linux 8 / CentOS 8
- Oracle Linux 7 / Red Hat Enterprise Linux 7 / CentOS 7
- Oracle Linux 6 / Red Hat Enterprise Linux 6 / CentOS 6
- Ubuntu 19.10 / 20.04
- Ubuntu 18.04 / 18.10 / 19.04
- Ubuntu 16.04
- Ubuntu 14.04 / 14.10 / 15.04
- Debian 10
- Debian 9
- Debian 8
- openSUSE 15.0
- openSUSE 13.2 / Leap 42
- Fedora 32
- Fedora 31
- Fedora 29 / 30
- Fedora 26 / 27 / 28
- All distributions (built on EL6 and therefore not requiring recent system libraries)

You might want to compare the checksums to verify the integrity of downloaded packages. *The SHA256 checksums should be favored as the MD5 algorithm must be treated as insecure!*

- SHA256 checksums, MD5 checksums

And we have, disregarding whichever operating system we have, to download, and later install, the server extensions:

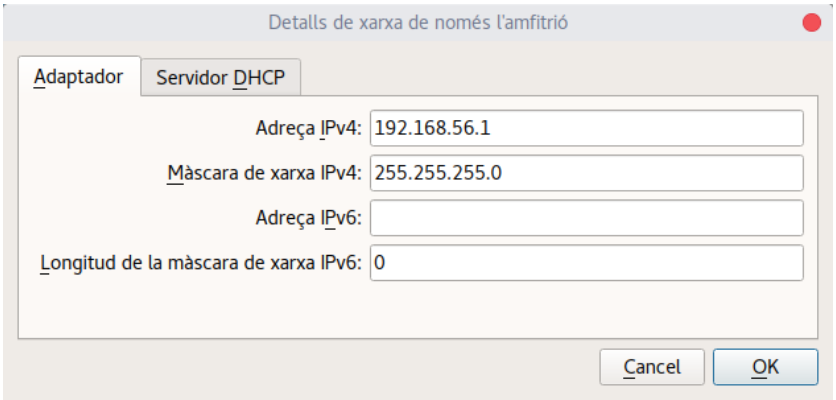
VirtualBox 6.1.14 Oracle VM VirtualBox Extension Pack

- All supported platforms

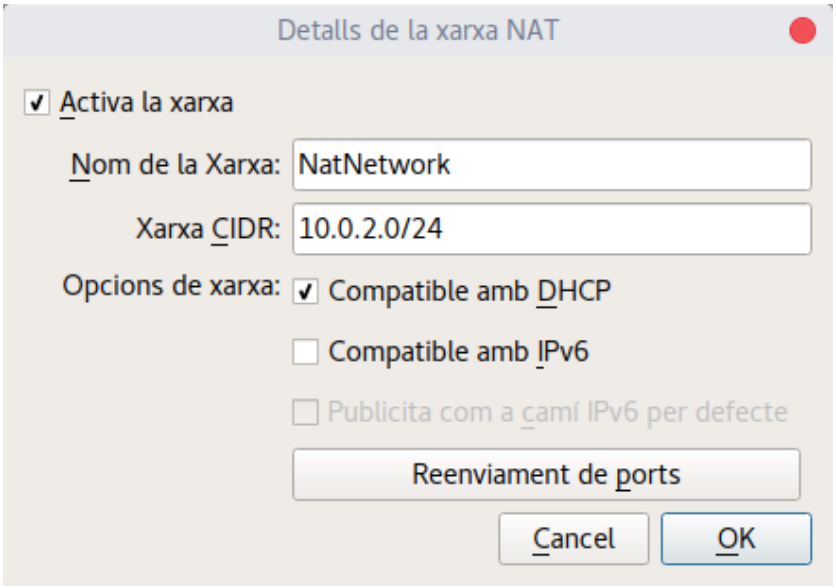
Support for USB 2.0 and USB 3.0 devices, VirtualBox RDP, disk encryption, NVMe and PXE boot for Intel cards. See this chapter from the User Manual for an introduction to this Extension Pack. The Extension Pack binaries are released under the VirtualBox Personal Use and Evaluation License (PUEL). *Please install the same version extension pack as your installed version of VirtualBox.*

After installation we can, if needed, configure the system. There's a couple things that we can configure on VirtualBox that are required for most deployments:

- Internal (Host Only) Networks:



- Nat Networks:



They allow us to create networks that we can connect virtual machines to that are, either restricted to only those virtual machines connected to them plus the virtualbox host, or restricted to virtual machines connected plus the host plus a network NAT service that allows connected VMs to access the internet.

Once we have both VirtualBox and the VirtualBox Extensions we can proceed to create our first VM.

2. VM Creation

When we create a VM we can opt for two methods: *guided* and *expert*. We can select from the initial page wether we want guided or expert mode. Here we give the VM a name, and tell VirtualBox which operating system and variant our new VM will be. It's important to pay atention to whether we are choosing a 32 or a 64 bit variant, as a bad choice here, choosing a 32 bit machine when we want a 64 bit OS will hamper the hability of our VM to boot.

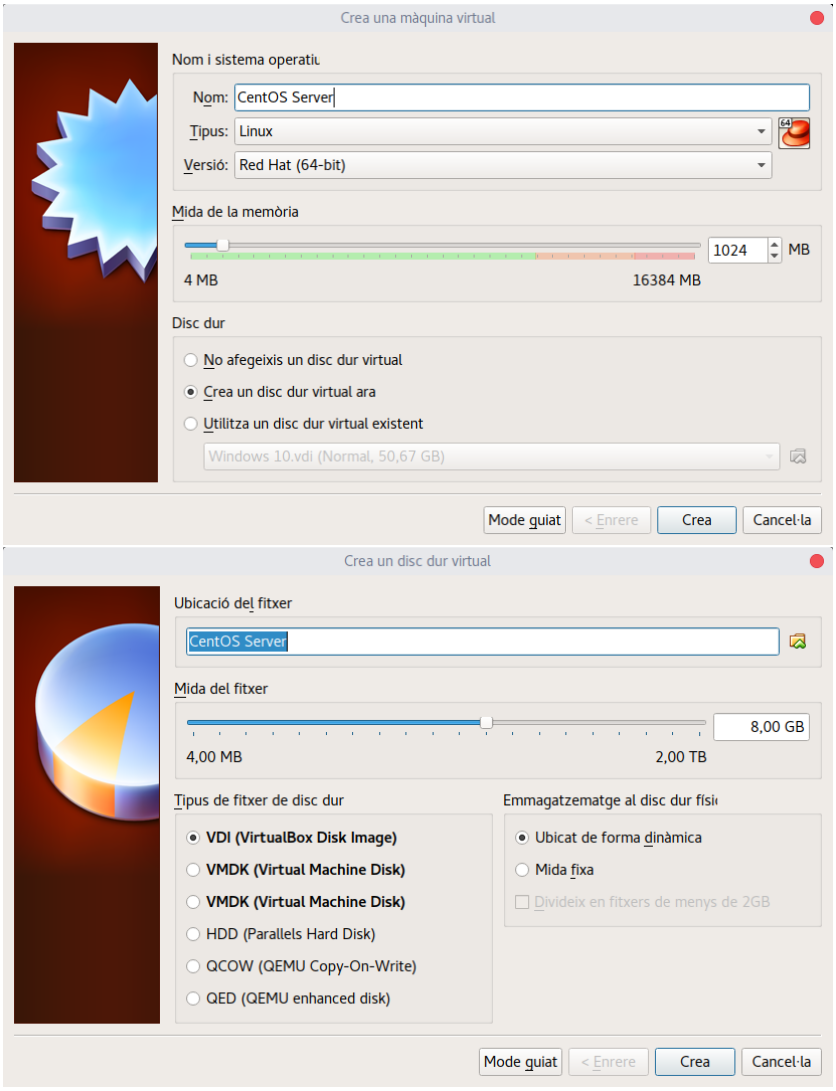
If we continue with the guided method, VirtualBox will ask us for the relevant parameters for the creation of the VM.

First we select the amount of RAM our machine will have access to (limited to the amount of existing memory on the host machine).

Then we select the storage available to our machine, we can leave our VM without hard disk, create a new virtual HD, or reuse an existing one.

We can create one choosing format, size, and location.

Or we can go through the expert mode, that gives us the same options as the guided one, but in only two screens:



3. Create one on expert mode:

   Create one with the following characteristics:

Name	CentOS Server 01
Operating System	RedHat (64bit)
Memory	768MB
Hard Disk	4GB

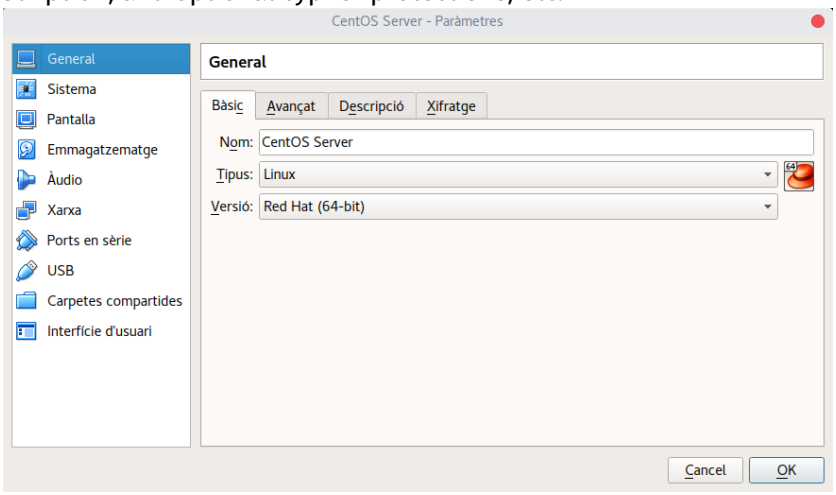
   Don't install anything yet to this VM.

4. VM Configuration

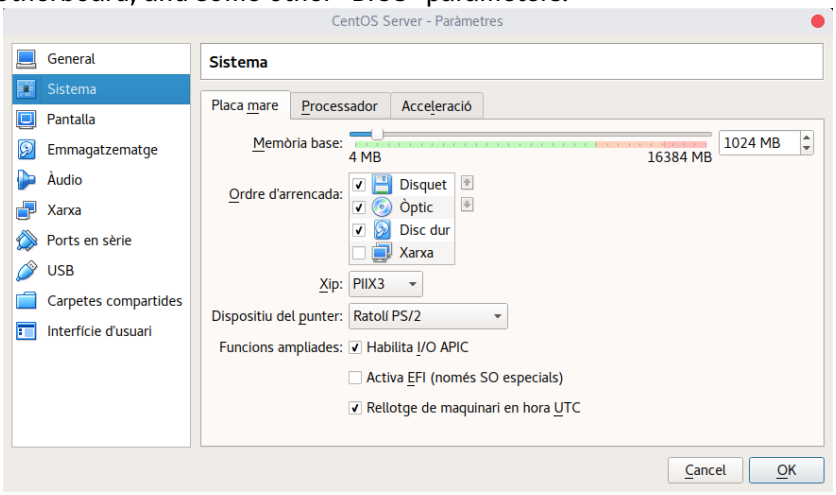
   After we have created our new VM, we can tune several important parameters, specially if we intend to use the newly created VM

regularly or in production.

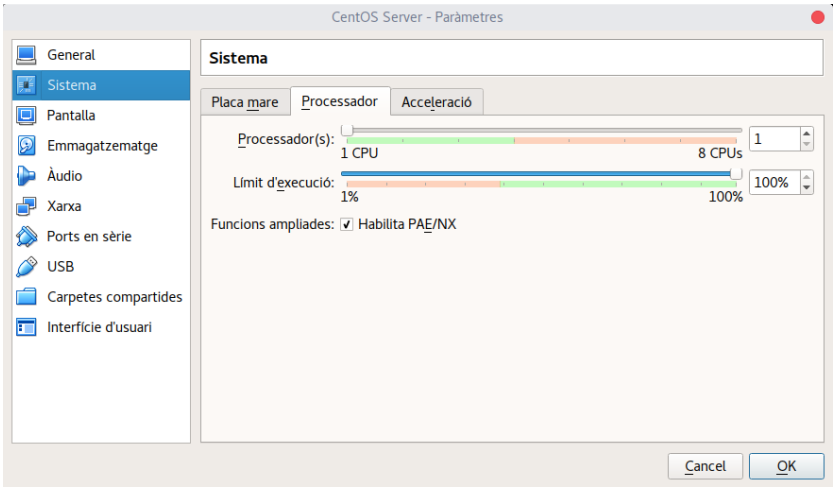
We can change general parameters, as the VM Name, assigned OS, description, and optional cypher protections, etc:



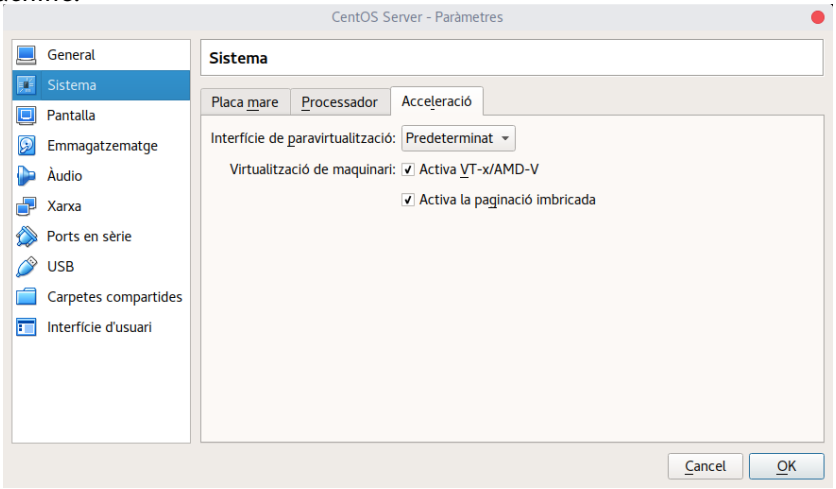
The we can configure the system. We can change the memory assigned to the VM, boot order, which kind of chipset has our virtual motherboard, and some other "BIOS" parameters:



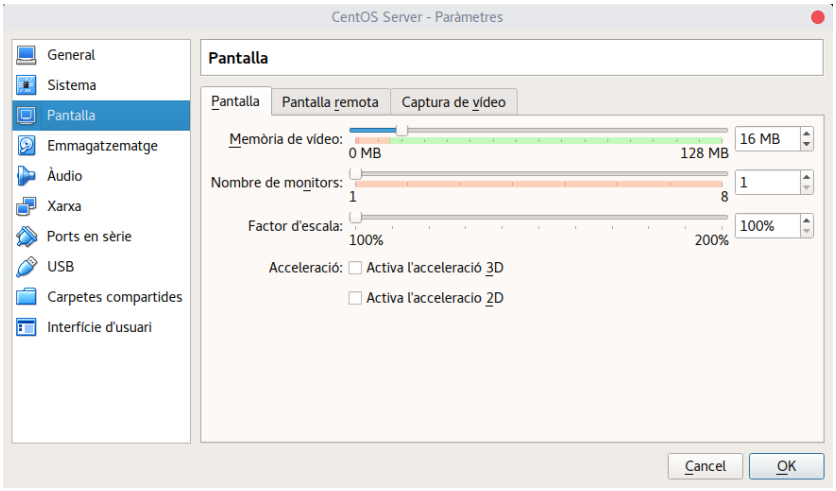
We can assign more CPUs and stablish CPU limits to the VM:



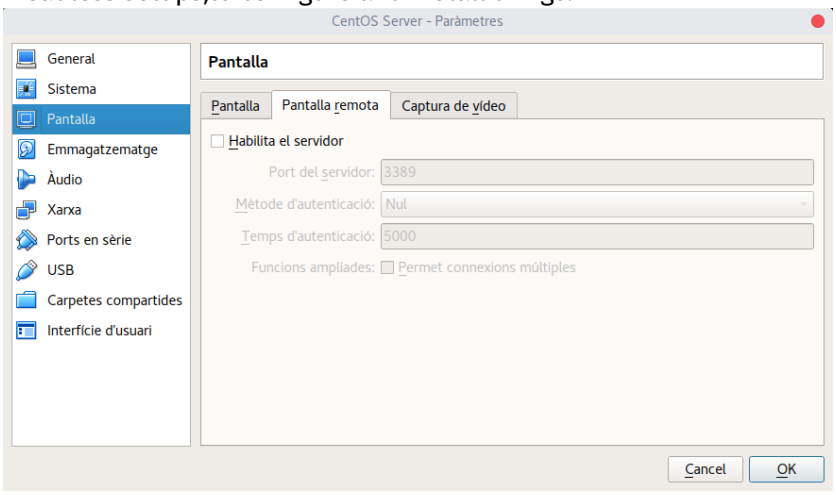
And enable VT-x/AMD-V (hardware Virtualization capabilities), so we coud run another virtual machines hypervisor inside the virtual machine.



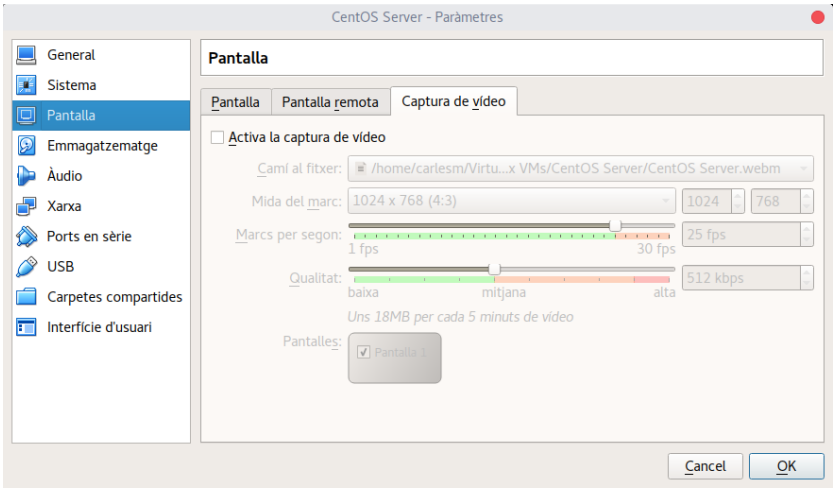
We can then configure the video card characteristics, not really important when using servers, although we will use, during the course, at least one machine with screen/keyboard (Kali Linux).



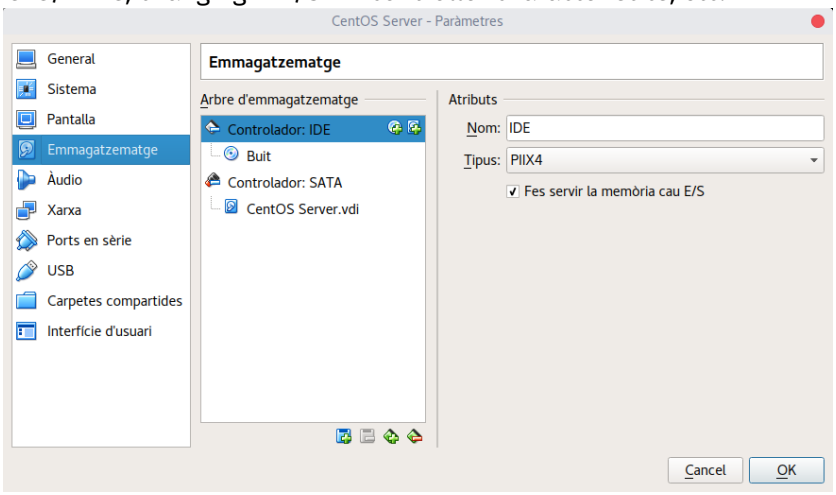
But on screen configuration we can also set up the remote screen sharing. This allows us to access the screen/keyboard of a VM no by connecting directly hardware to the VM but by accessing through a remote desktop connection (RDP). That's really necessary, specially on headless setups,to configure and install things.



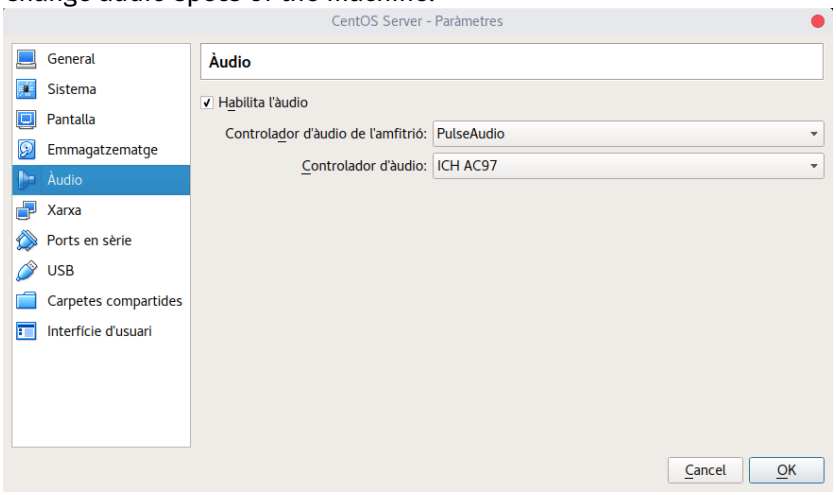
We can also activate video recording of the virtual machine. We could use it to record demos and tutorials of how we do something with a virtual machine.



We can then configure storage options: adding disks, ISO images as CDs/DVDs, changing IDE/SATA controller characteristics, etc.

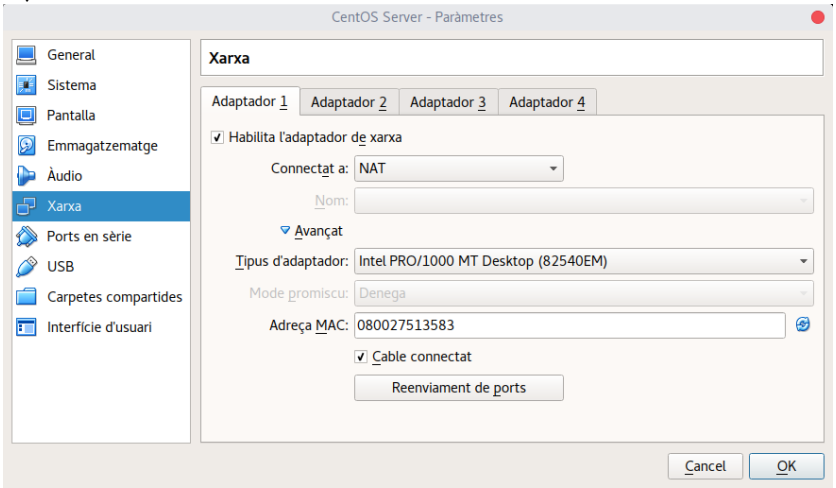


Change audio specs of the machine:

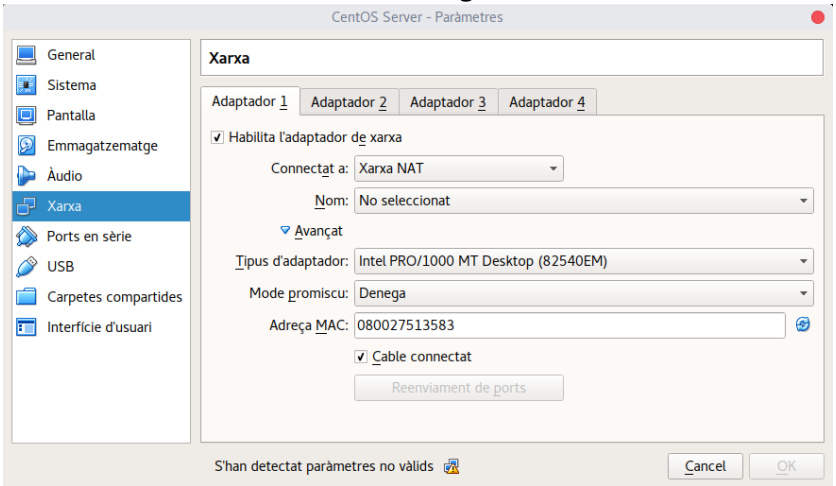


And one of the most important for us. Change the network be-

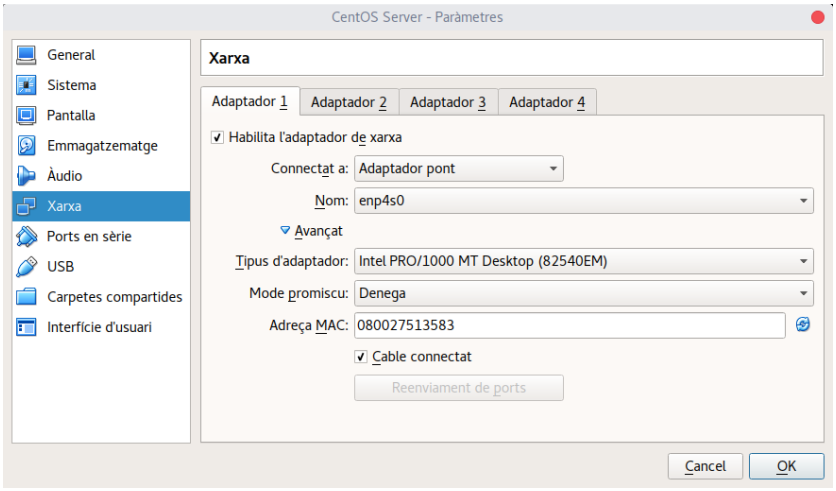
haviour of the VM. We can connect up to 4 network cards, with different types of connections (changing, also, MAC addresses, network card type, etc.). We can add a NAT interface (virtual machine will have a private IP address and will access the internet through the host):



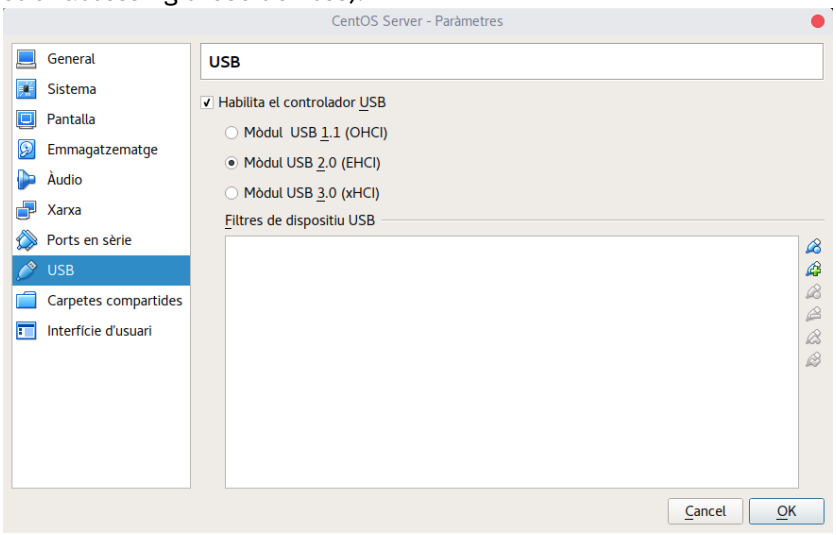
We can add a NAT Network, where the VM will be on a private network that accesses the internet through the host.



Or a bridge interface, where the VM accesses the same network as the host, through the specified interface as if it was directly connected to that network:

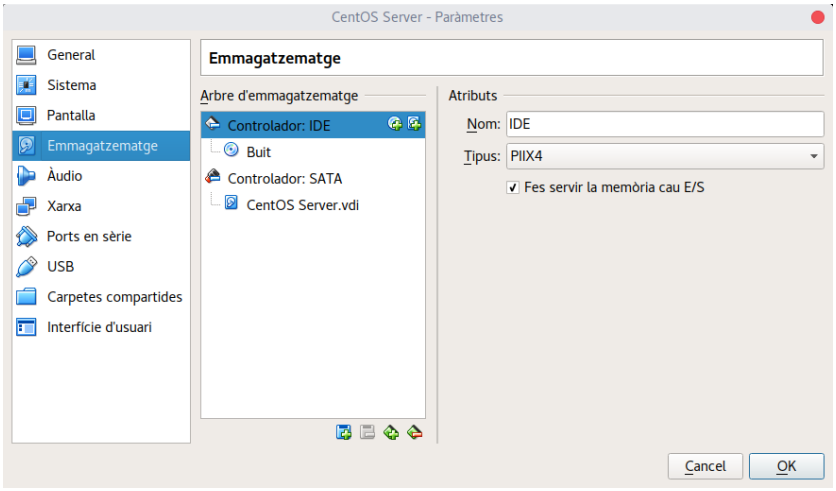


And we can map host USB devices to the VM (this disallows the host of accessing those devices):

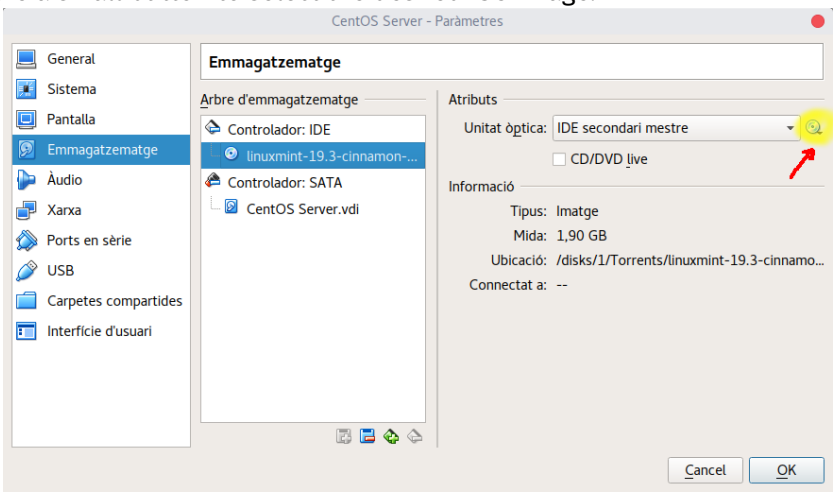


5. How to install OS

You can “insert” a CD (a **.iso**) image into any virtual machine, to do that, go to the Storage parameters screen.



Select the Empty CD in the IDE controller. And to the right, you have a small button to select the desired ISO image.



6. Configuration from command line

VirtualBox VMs can be configured not only through the UI but from the command line, with a tool called VBoxManage.

With this tool we can do all we can do on the UI: create and remove VMs, start and stop them, configure their parameters, etc.

But there are somethings that cannot be done easyly from the UI interface. Some of those are really interesting for us.

7. Listing VMs

First of all we will have to list our VMs, as we will be using either their name or their UUID to issue requests throught VBoxManage.

So we list them with:

```
1 $ VBoxManage list vms
2 "docker-provider" {7919b9c7-0e92-41e2-8609-478ec57d8900}
```

```
"Windows 10 Victim01" {54f7319d-7ea8-4833-bc4d-479ead60652d}
"LinuxAE v4" {e67098b2-64c4-4fob-a96a-3f05e20474f9}
"CentOS Server" {da895bef-5efo-4794-b1f8-9ac58b69b816}
```

We can use, in all commands, either the name or the UUID to refer to a VM.

8. Network traffic tracing

We can activate a network trace on any nic. That will store on a given file all the data comming in and out of a network interface.

For example, to store all network traffic on NIC 1 to network-nic-1-dump.cap we can do:

```
$ VBoxManage controlvm da895bef-5efo-4794-b1f8-9ac58b69b816
    nictracefile 1 network-nic-1-dump.cap
$ VBoxManage controlvm da895bef-5efo-4794-b1f8-9ac58b69b816
    nictrace 1 on
```

9. Mapping PCI NICs

Instead of adding Network connections through the host NIC (either NAT or bridged interfaces), we can map PCI network cards to the VM directly, bypassing completely the server. This is useful on scenarios where we are using **real** server hardware and some VM requires more throughput than the others.

To attach a PCI NIC to a VM we can do:

```
$ VBoxManage modifyvm da895bef-5efo-4794-b1f8-9ac58b69b816 --
    pciattach 04:00.0
```

To get the PCI bus ID of any hardware, on Linux, use: lspci

```
$ lspci
00:00.0 Host bridge: Intel Corporation Core Processor DMI (rev
    11)
00:03.0 PCI bridge: Intel Corporation Core Processor PCI Express
     Root Port 1 (rev 11)
00:08.0 System peripheral: Intel Corporation Core Processor
    System Management Registers (rev 11)
00:10.1 System peripheral: Intel Corporation Core Processor QPI
    Routing and Protocol Registers (rev 11)
00:1a.0 USB controller: Intel Corporation 5 Series/3400 Series
    Chipset USB Universal Host Controller (rev 06)
00:1a.1 USB controller: Intel Corporation 5 Series/3400 Series
    Chipset USB Universal Host Controller
[...]
00:1f.5 IDE interface: Intel Corporation 5 Series/3400 Series
    Chipset 2 port SATA IDE Controller (rev 06)
01:00.0 VGA compatible controller: NVIDIA Corporation GP104 [
    GeForce GTX 1070] (rev a1)
01:00.1 Audio device: NVIDIA Corporation GP104 High Definition
    Audio Controller (rev a1)
02:00.0 USB controller: VIA Technologies, Inc. VL80x xHCI USB
    3.0 Controller (rev 02)
```

```
03:00.0 SATA controller: JMicron Technology Corp. JMB363 SATA/IDE Controller (rev 02)
03:00.1 IDE interface: JMicron Technology Corp. JMB363 SATA/IDE Controller (rev 02)
04:00.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL8111/8168/8411 PCI Express Gigabit Ethernet Controller (rev 03)
```

10. USB Attach

From the UI we can attach USB devices present on the host or generic (empty) devices that we can modify with the info of a given device.

But from the command line we can do, additionally, that while, simultaneously storing the USB traffic on a file (akin to a network trace file).

We will need either the USB address on host or the UUID, we can get it with:

```
$ VBoxManage list usbhost
Host USB Devices:

UUID:               4447efff-7912-4cfd-9791-491eff46aa2d
VendorId:           0x12c9 (12C9)
ProductId:          0x1017 (1017)
Revision:           1.66 (0166)
Port:               2
USB version/speed:  2/Full
Manufacturer:       E-Signal
Product:            USB Gaming Mouse
Address:            sysfs:/sys/devices/pci0000:00/0000:00:1a.7/usb1/1-1/1-1.4/1-1.4.3//device:/dev/vboxusb/001/007
Current State:      Busy

UUID:               443f59df-e0a3-4669-8a9c-7ee34c8568cb
VendorId:           0x0d8c (0D8C)
ProductId:          0x0012 (0012)
Revision:           1.0 (0100)
Port:               1
USB version/speed:  1/Full
Manufacturer:       C-Media Electronics Inc.
Product:            USB Audio Device
Address:            sysfs:/sys/devices/pci0000:00/0000:00:1a.0/usb3/3-2//device:/dev/vboxusb/003/002
Current State:      Busy

UUID:               a965ad72-ffd1-49e2-883d-f40e47049b05
VendorId:           0x046d (046D)
ProductId:          0xc317 (C317)
Revision:           0.112 (00112)
Port:               0
USB version/speed:  1/Low
Manufacturer:       Logitech
Product:            USB Multimedia Keyboard
Address:            sysfs:/sys/devices/pci0000:00/0000:00:1a.7/usb1/1-1/1-1.4/1-1.4.1//device:/dev/vboxusb/001/006
Current State:      Busy
```

```
36
37 UUID:               ba7dc9a7-1d94-4f29-a0a2-a3092539a49d
38 VendorId:           0x046d (046D)
39 ProductId:          0x08c3 (08C3)
40 Revision:           0.5 (0005)
41 Port:               2
42 USB version/speed:  2/High
43 Manufacturer:       Logitech, Inc.
44 Product:            Camera (Notebooks Pro)
45 Address:            sysfs:/sys/devices/pci0000:00/0000:00:1a.7/
       usb1/1-1/1-1.3//device:/dev/vboxusb/001/004
46 Current State:      Busy
```

So, to map the USB camera dumping all traffic to a file:

```
1 $ VBoxManage modifyvm da895bef-5efo-4794-b1f8-9ac58b69b816
      usbattach ba7dc9a7-1d94-4f29-a0a2-a3092539a49d --capturefile
       camera-usb.cap
```

11. Headless Mode

We can run VirtualBox VMs without requiring a screen and keyboard to use the GUI. To do that we have a command, called VBoxHeadless.

We can start a VM with:

```
1 $ VBoxHeadless -startvm da895bef-5efo-4794-b1f8-9ac58b69b816
```

or an equivalent VBoxManage command:

```
1 $ VBoxManage startvm  da895bef-5efo-4794-b1f8-9ac58b69b816 --
      type headless
```

To stop/reset/etc the VM we have the VBoxManage controlvm command:

```
1 $ VBoxManage controlvm  da895bef-5efo-4794-b1f8-9ac58b69b816
      poweroff
```

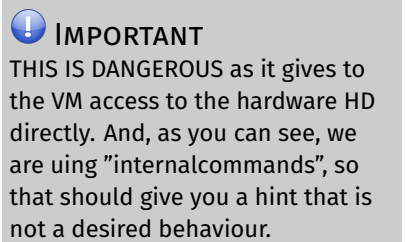
If we are running a headless VM that requires screen interaction, if we configure the remote display on that machine, we can use any RDP client to access the VM screen.

12. Storage management

We can do all the things we can do with the UI using the command line tool, and some more, but storage management is a bit complex and the UI makes it easy for almost all cases. But there is a usage, that we can require, specially if we have to do a forensics analysis of a machine that has to be done using command line.

That usage is giving to VM complete access to an existing physical Hard Disk (or partition on an existing physical Hard Disk). That way the VM will behave as is that HD was in fact it's configured HD.

To do that we will create, first, a VMDK (virtual disk image) that has no storage but references the physical HD, for example, /dev/sdf



```
$ VBoxManage internalcommands createrawvmdk -filename ourimage.
    vmdk -rawdisk /dev/sdf
```

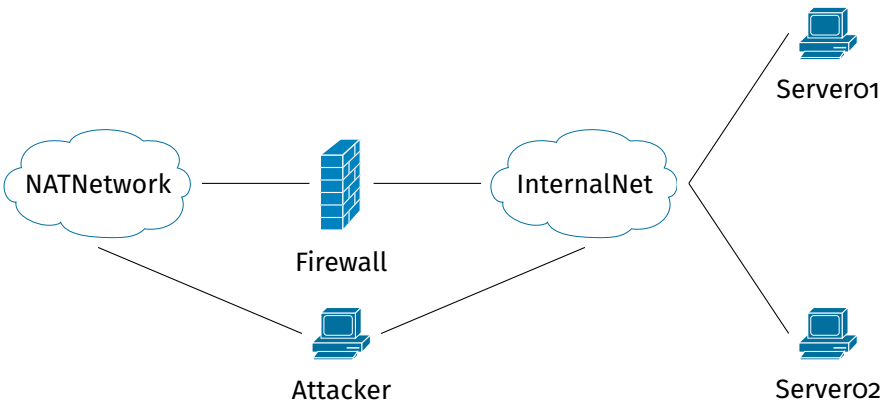
This will create a file ourimage.vmdk that we can attach to the VM, either using the UI or with:

```
$ VBoxManage storageattach da895bef-5efo-4794-b1f8-9ac58b69b816
    --storagectl "IDE Controller" \
 --port o --device o --type hdd --medium ourimage.vmdk
```

We can use this to test a suspect HD (after making copies of it) on a VM, not on the real host, isolating it from the network while tracing all traffic, for instance.

13. Our scenario

This is the network scenario we will be building:



14. Create Server01, Server02, Firewall and Attacker

For Server01 we can reuse the already existing VM. To add an indentical machine (Server02) we can either add an new VM with the same characteristics or we can clone the existing Server01 and changing its name.

Create now, both, the Firewall and the Attacker:

Name	Firewall
Operating System	RedHat (64bit)
Memory	768MB
Hard Disk	4GB

Name	Kali Linux
Operating System	Debian (64bit)
Memory	>1024MB
Hard Disk	>8GB

15. Create the networks

On the one hand we have to create the Internet for our scenario, so we will create a NAT Network, call it natnetwork. On the other create an internal network, call it, internaltnet.

When created, assign interfaces as depicted, i.e., firewall to both networks, attacker to both networks, and the servers only to the internal network.