KVM (Kernel Virtual Machine)

Name – Atharv Natu

Roll No. & Div - 63 TY CS-D

What is KVM ?

- KVM (for Kernel-based Virtual Machine) is a full virtualization solution for Linux on x86 hardware containing virtualization extensions (Intel VT or AMD-V).
- Using KVM, one can run multiple virtual machines running unmodified Linux or Windows images. Each virtual machine has private virtualized hardware: a network card, disk, graphics adapter, etc.
- KVM is open source software. The kernel component of KVM is included in mainline Linux, as of 2.6.20. KVM was first announced in 2006 and merged into the mainline Linux kernel version a year later. Because KVM is part of existing Linux code, it immediately benefits from every new Linux feature, fix, and advancement without additional engineering.

How does KVM Work ?

- KVM converts Linux into a type-1 (bare-metal) hypervisor.
- All hypervisors need some operating system-level components—such as a memory manager, process scheduler, input/output (I/O) stack, device drivers, security manager, a network stack, and more—to run VMs.
- KVM has all these components because it’s part of the Linux kernel. Every VM is implemented as a regular Linux process, scheduled by the standard Linux scheduler, with dedicated virtual hardware like a network card, graphics adapter, CPU(s), memory, and disks.

KVM features

KVM is part of Linux. Linux is part of KVM. Everything Linux has, KVM has too. But there are specific features that make KVM an enterprise’s preferred hypervisor.

Security

KVM uses a combination of security-enhanced Linux (SELinux) and secure virtualization (sVirt) for enhanced VM security and isolation. SELinux establishes security boundaries around VMs. sVirt extends SELinux's capabilities, allowing Mandatory Access Control (MAC) security to be applied to guest VMs and preventing manual labeling errors.

Storage

KVM is able to use any storage supported by Linux, including some local disks and network-attached storage (NAS). Multipath I/O may be used to improve storage and provide redundancy. KVM also supports shared file systems so VM images may be shared by multiple hosts. Disk images support thin provisioning, allocating storage on demand rather than all up front.

Hardware support

KVM can use a wide variety of certified Linux-supported hardware platforms. Because hardware vendors regularly contribute to kernel development, the latest hardware features are often rapidly adopted in the Linux kernel.

Memory management

KVM inherits the memory management features of Linux, including non-uniform memory access and kernel same-page merging. The memory of a VM can be swapped, backed by large volumes for better performance, and shared or backed by a disk file.

Live migration

KVM supports live migration, which is the ability to move a running VM between physical hosts with no service interruption. The VM remains powered on, network connections remain active, and applications continue to run while the VM is relocated. KVM also saves a VM's current state so it can be stored and resumed later.

Performance and scalability

KVM inherits the performance of Linux, scaling to match demand load if the number of guest machines and requests increases. KVM allows the most demanding application workloads to be virtualized and is the basis for many enterprise virtualization setups, such as datacenters and private clouds (via OpenStack®).

Scheduling and resource control

In the KVM model, a VM is a Linux process, scheduled and managed by the kernel. The Linux scheduler allows fine-grained control of the resources allocated to a Linux process and guarantees a quality of service for a particular process. In KVM, this includes the completely fair scheduler, control groups, network name spaces, and real-time extensions.

Lower latency and higher prioritization

The Linux kernel features real-time extensions that allow VM-based apps to run at lower latency with better prioritization (compared to bare metal). The kernel also divides processes that require long computing times into smaller components, which are then scheduled and processed accordingly.

Requirements for installing virtual machines on KVM

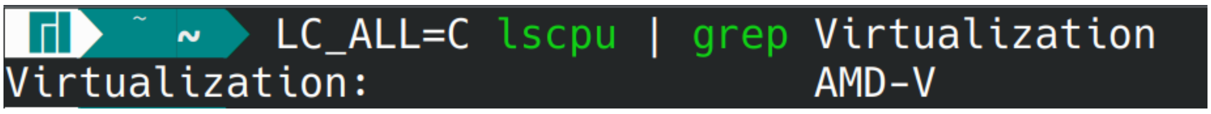
1) Linux Operating System
2) Hardware Virtualization Support
3) Minimum 2 Processor Cores
4) Minimum 4 GB RAM
5) Minimum 20 GB Free Storage Space

Pre-installation configuration

You may need to enable virtualization support in your BIOS. All x86_64 processors manufactured by AMD and Intel in the last 10 years support virtualization. If it looks like your processor does not support virtualization, it is almost certainly turned off in the BIOS.

You can check whether your processor supports hardware virtualization with the following command

```
$ LC_ALL=C lscpu | grep Virtualization
```



```
 ~  LC_ALL=C lscpu | grep Virtualization
Virtualization:                     AMD-V
```

As you can see in the above image, we have an AMD processor which has support for hardware virtualization.

If nothing is displayed after running the command, then your CPU does not support hardware virtualization, and you will not be able to use KVM.

Installing KVM on different Linux Distros

1) Debian and Ubuntu based Distros

sudo apt install qemu-kvm bridge-utils virt-manager libosinfo-bin -y

2) Fedora

sudo dnf -y install bridge-utils libvirt virt-install qemu-kvm

3) CentOS or RedHat

yum install qemu-kvm libvirt virt-install virt-manager virt-install -y

4) Arch Linux based Distros

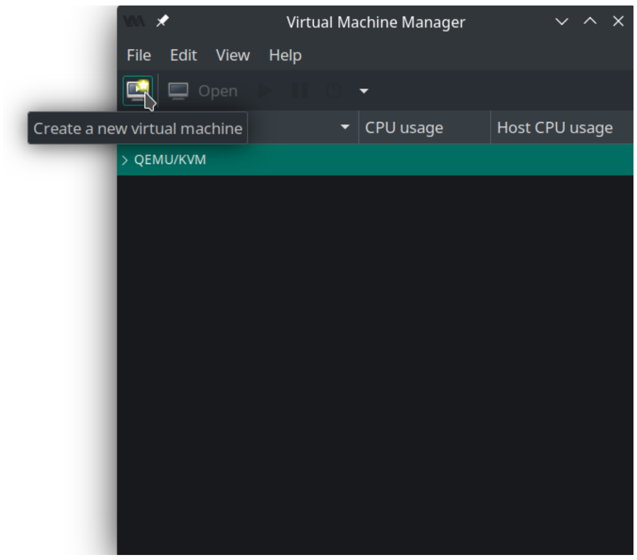
sudo pacman -S qemu qemu-arch-extra ovmf bridge-utils dnsmasq vde2 openbsd-netcat ebtables iptables

Steps to install Windows 10 using KVM

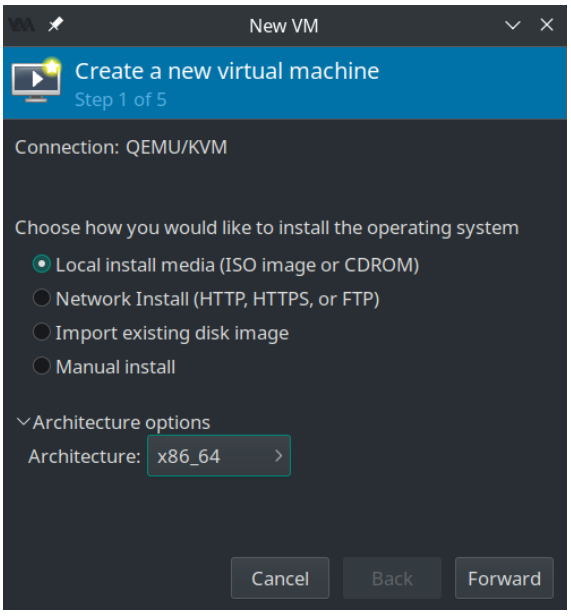
- Create Windows 10 VM for KVM
- Customize Hardware
- Add Graphics Hardware
- Update Virtual Network Interface
- Add Windows 10 storage and hardware drivers for KVM
- Configure boot device order
- Begin Windows 10 installation
- Install Windows virtio drivers
- Install QXL display drivers

Create Windows 10 VM

1. Open Virtual Machine Manager by searching in installed applications or by opening the terminal and typing ***virt-manager***
2. Click on the **Create a new virtual machine** button



3. Select local install media, select architecture as x86_x64 and click Forward



New VM

Create a new virtual machine
Step 1 of 5

Connection: QEMU/KVM

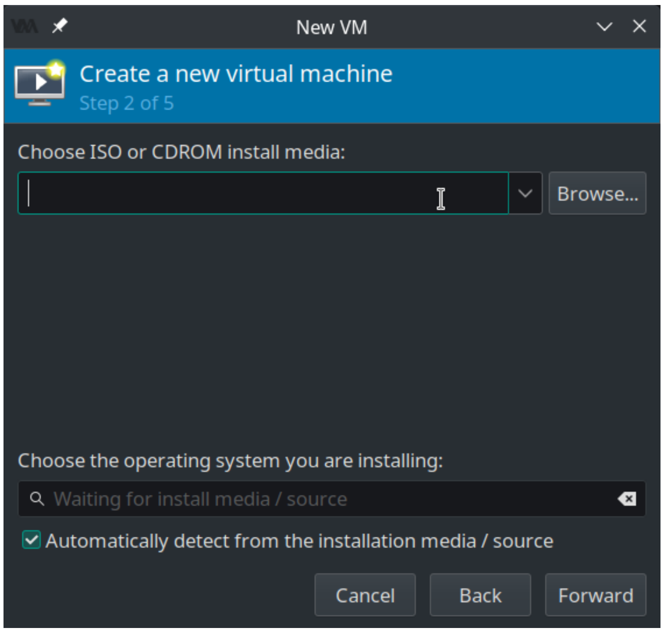
Choose how you would like to install the operating system

- [✓ Local install media (ISO image or CDROM)]
- Network Install (HTTP, HTTPS, or FTP)
- Import existing disk image
- Manual install

Architecture options
Architecture: x86_64

Cancel Back Forward

4. You need to point the Windows 10 ISO image to the Virtualization manager, click on Browse.



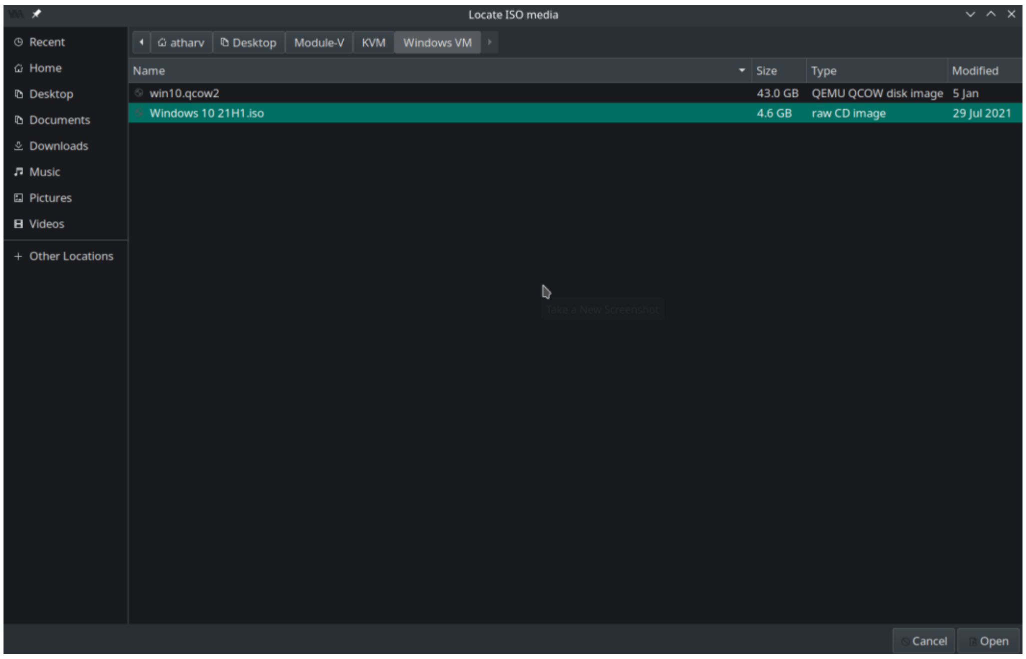
Choose ISO or CDROM install media:

Choose the operating system you are installing:

Waiting for install media / source

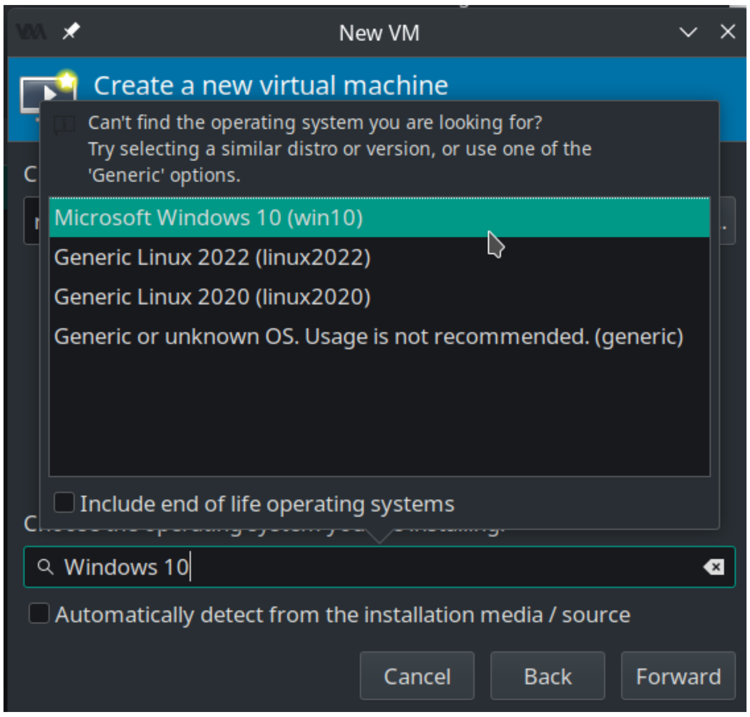
☑ Automatically detect from the installation media / source

5. Select Windows 10 ISO image and click on open



Name	Size	Type	Modified
win10.qcow2	43.0 GB	QEMU QCOW disk image	5 Jan
Windows 10 21H1.iso	4.6 GB	raw CD image	29 Jul 2021

6. Uncheck the option Automatically detect operating system based on install media, and select the OS type as Windows and Microsoft Windows 10 and click Forward.



Can't find the operating system you are looking for?
Try selecting a similar distro or version, or use one of the
'Generic' options.

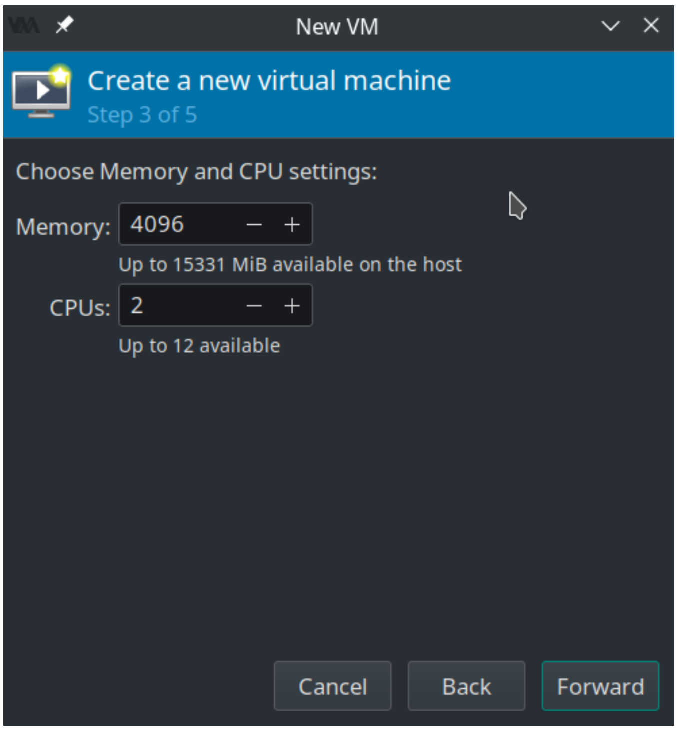
Microsoft Windows 10 (win10)
Generic Linux 2022 (linux2022)
Generic Linux 2020 (linux2020)
Generic or unknown OS. Usage is not recommended. (generic)

☐ Include end of life operating systems

Windows 10

☐ Automatically detect from the installation media / source

7. Now, you need to specify Memory and CPU configuration according to your requirements and available amount of resources.

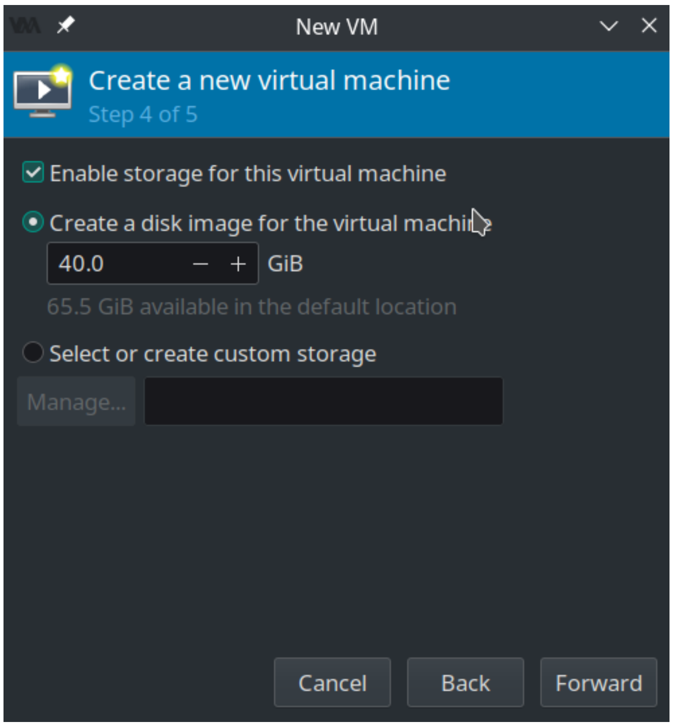


Choose Memory and CPU settings:

Memory: 4096
Up to 15331 MiB available on the host

CPUs: 2
Up to 12 available

8. In the next step, we need to create a virtual hard disk. The default storage allocated is 40GB, but according to your requirement and available resources, you can determine your storage size.



☑ Enable storage for this virtual machine

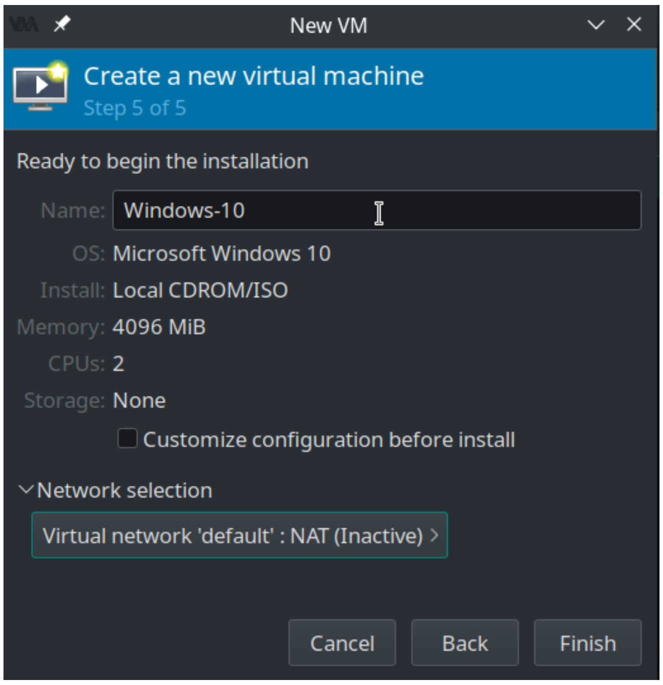
● Create a disk image for the virtual machine

40.0 GiB

65.5 GiB available in the default location

○ Select or create custom storage

9. On to the final step of creating the virtual machine, name your virtual machine and determine the network type. Then, click on finish button.



Ready to begin the installation

Name: Windows-10

OS: Microsoft Windows 10

Install: Local CDROM/ISO

Memory: 4096 MiB

CPUs: 2

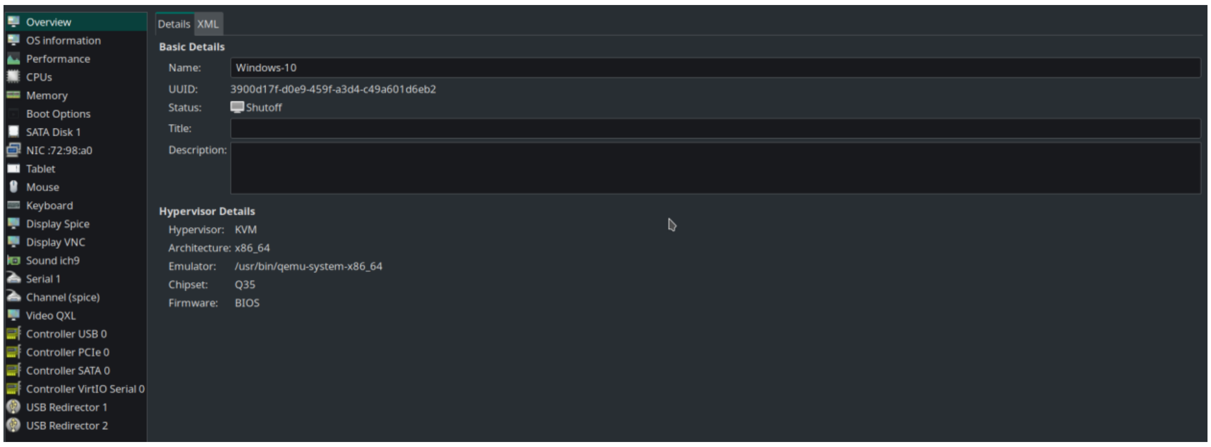
Storage: None

☐ Customize configuration before install

Network selection

Virtual network 'default' : NAT (Inactive)

10. After finishing, now check the virtual machine details, where you can see the Hypervisor details along with configuring the machine.



Basic Details

Name: Windows-10

UUID: 3900d17f-d0e9-459f-a3d4-c49a601d6eb2

Status: Shutoff

Title:

Description:

Hypervisor Details

Hypervisor: KVM

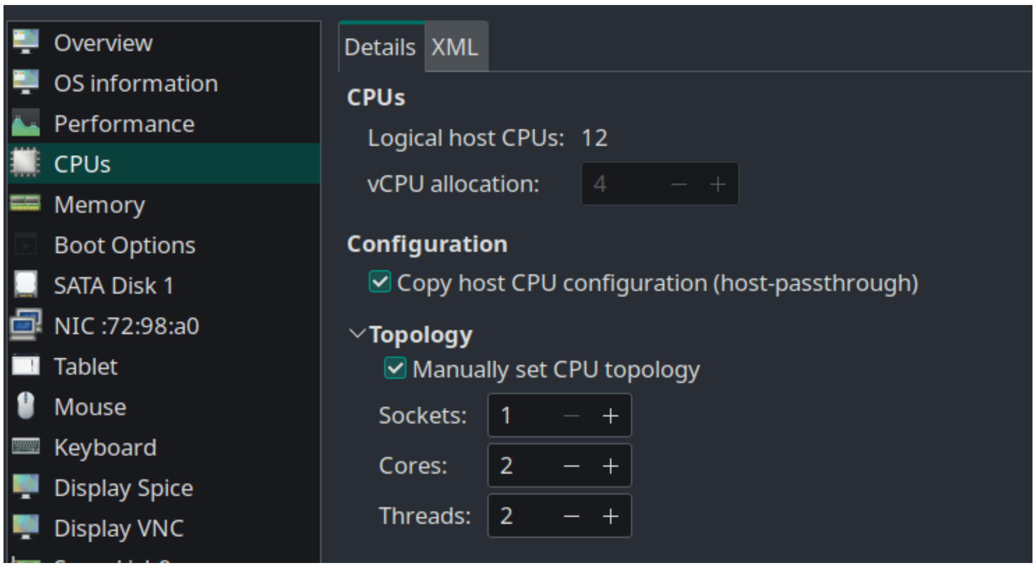
Architecture: x86_64

Emulator: /usr/bin/qemu-system-x86_64

Chipset: Q35

Firmware: BIOS

11. In the CPUs section, select the topology options, with your preferred cores and threads. Remember, you also need CPU cores for the host to run KVM.



CPUs

Logical host CPUs: 12

vCPU allocation: 4

Configuration

✓ Copy host CPU configuration (host-passthrough)

Topology

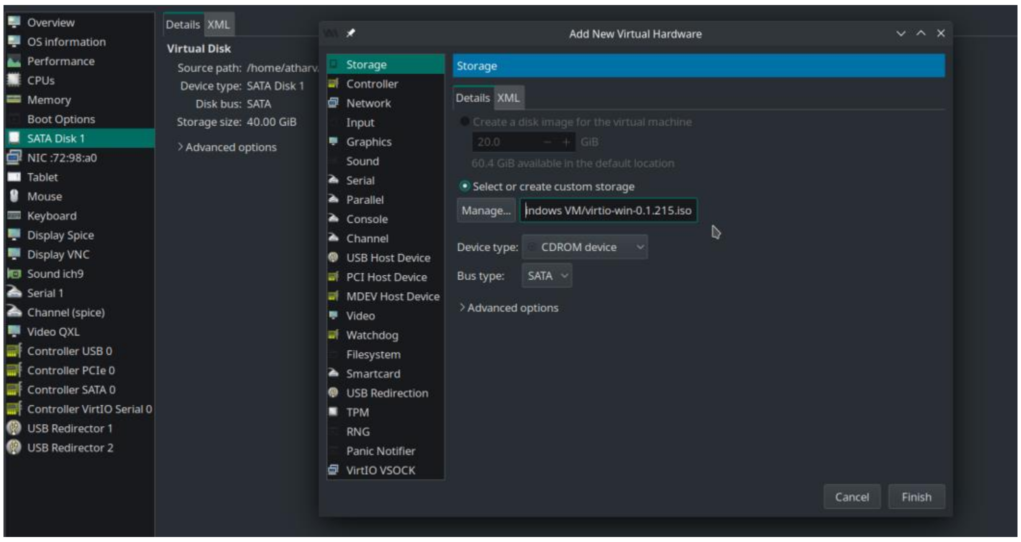
✓ Manually set CPU topology

Sockets: 1

Cores: 2

Threads: 2

12. Add the Windows Guest Tools ISO file, set the Device Type as CDROM Device.



Virtual Disk

Source path: /home/atharv

Device type: SATA Disk 1

Disk bus: SATA

Storage size: 40.00 GiB

Advanced options

Storage

Create a disk image for the virtual machine

20.0 GiB

60.4 GiB available in the default location

● Select or create custom storage

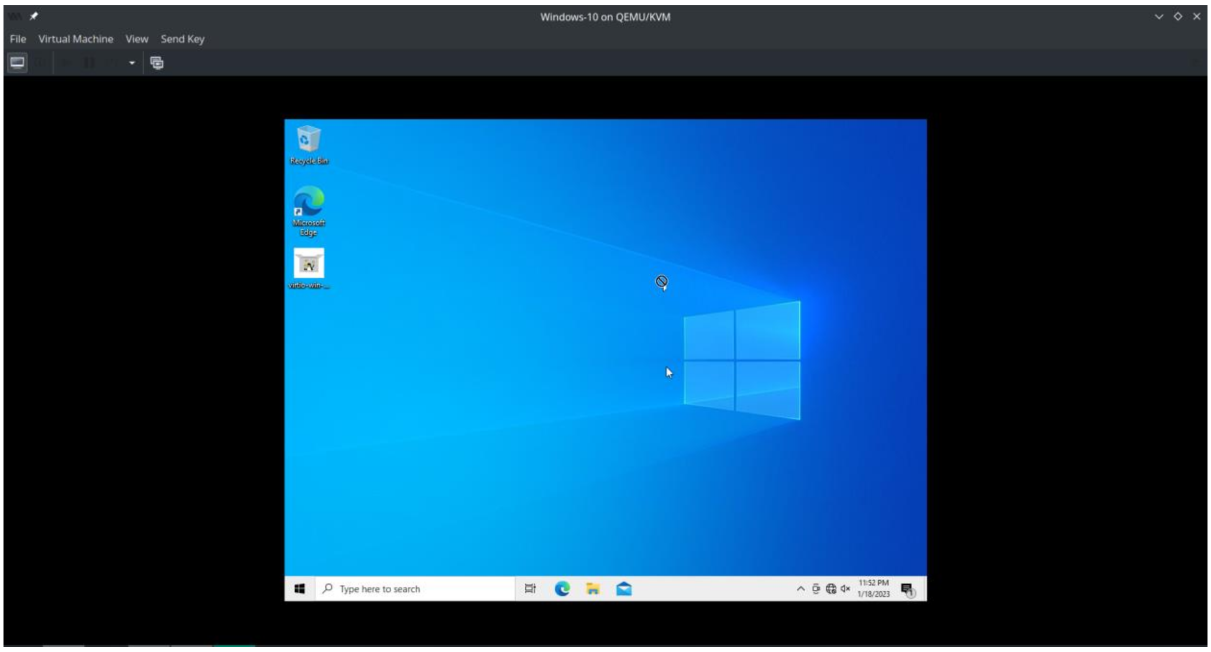
Manage... indows VM/virtio-win-0.1.215.iso

Device type: CDROM device

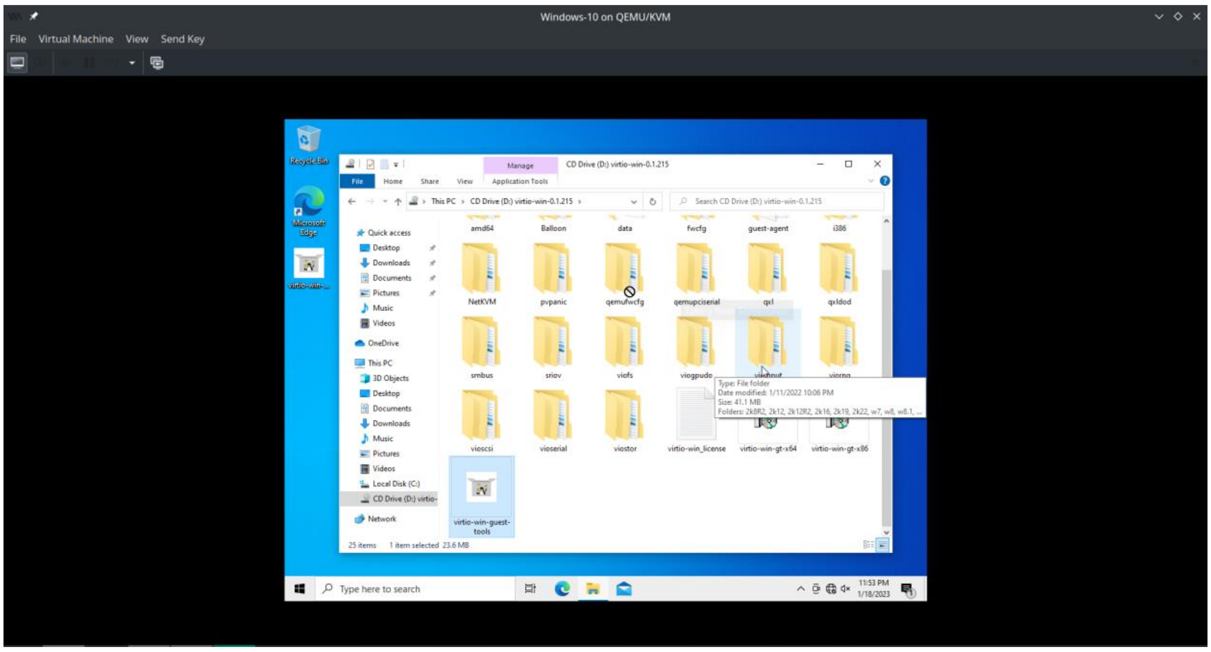
Bus type: SATA

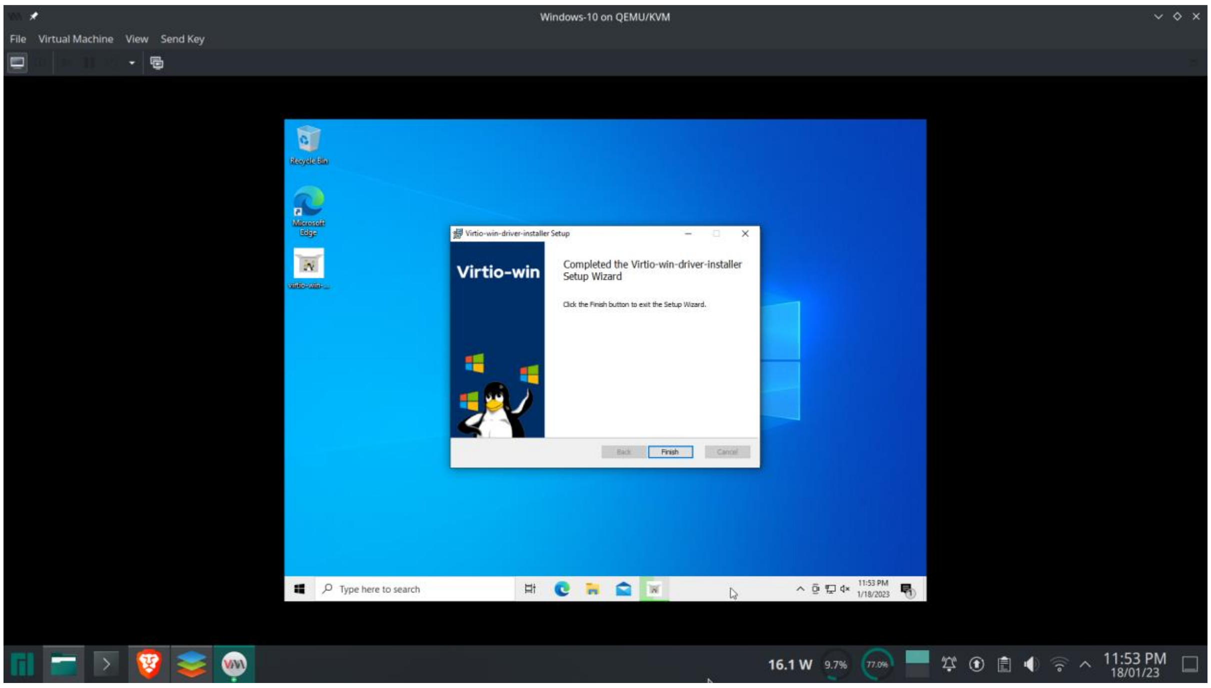
Advanced options

13. In this way, our virtual machine is created and configured according to our needs. Now, run the virtual machine, the Windows ISO will boot. Follow the traditional steps for installing Windows 10 according to your requirements. After the installation is complete, you will see the following screen.



14. As we have inserted the guest tools ISO as CDROM, open it using Explorer and run the **virtio-win-guest-tools.exe.**

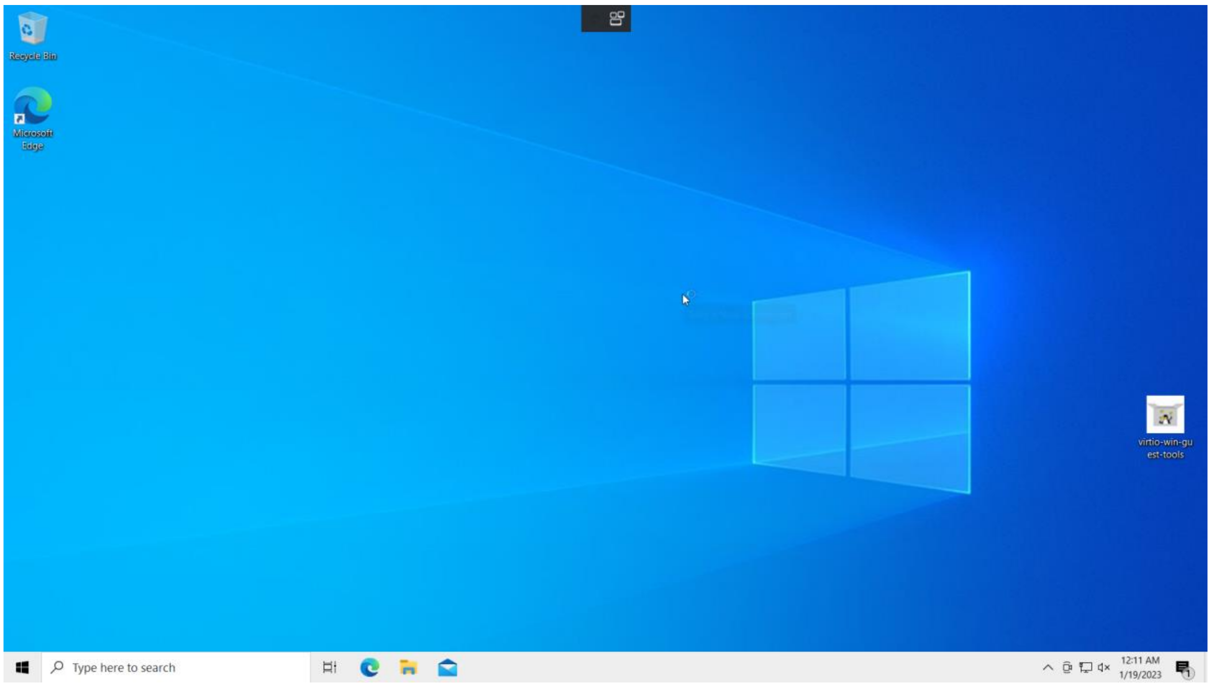



Windows-10 on QEMU/KVM
File Virtual Machine View Send Key
Virtio-win-driver-installer Setup
Virtio-win
Back
Finish
Cancel
Type here to search

Completed the Virtio-win-driver-installer Setup Wizard

Click the Finish button to exit the Setup Wizard.

15. After the guest tools are installed, you will have graphics acceleration, which will enable full screen and scaling. The guest tools come with collection of drivers, which will enable as close to native performance on the virtual machine.



References

1. *https://www.linux-kvm.org/page/Main_Page*

2. *https://help.ubuntu.com/community/KVM/Installation*

3. *https://wiki.archlinux.org/title/KVM*

4. *https://www.linux-kvm.org/page/WindowsGuestDrivers/Download_Drivers*