



A

Project Report On

“KVM Nested Virtualization

Using

ORACLE VIRTUALBOX with CentOS”

BY

Sr No	Name
1	Rani Behare
2	Sakshi Shaha
3	Gauri Vetat
4	Renuka Chavan
5	Prajwal Nimkarde
6	Harsahl Patil
7	Dipankar Samantha

Under the guidance of

Mr.Zakir Hussain

INDEX

Sr No	Title
1	Problem Statement
2	Objective
3	Hardware and Software Requirement
4	Virtualization in Linux
5	Nested Virtualization
6	Kernel-Based Virtual Machine
7	Implementation with Output
8	Conclusion

Problem Statement

With the increasing demand for virtualization in modern computing environments, the ability to run virtual machines (VMs) within other VMs has become essential. This capability, known as nested virtualization, allows for enhanced testing, development, and deployment of cloud services and applications. KVM (Kernel-based Virtual Machine) is a popular open-source virtualization technology in Linux, while Oracle Virtual Box is a widely-used platform for running VMs across different operating systems.

The integration of KVM nested virtualization within Oracle Virtual Box on a CentOS host presents several challenges and opportunities. Specifically, the project aims to address the following key issues:

1. **Configuration Complexity:** Setting up nested virtualization requires intricate configuration steps. Determining the correct parameters and settings within both KVM and Virtual Box can be daunting, especially for users unfamiliar with virtualization technologies.
2. **Performance Implications:** Understanding the performance impact of running nested VMs is crucial for effective resource management. Assessing how the overhead of virtualization affects CPU, memory, and I/O performance is essential for applications requiring high availability and responsiveness.
3. **Compatibility and Stability:** Ensuring compatibility between KVM, Oracle Virtual Box, and CentOS is necessary for stable operation. The project will investigate potential issues arising from different versions and configurations, focusing on how they can affect the reliability of the virtualization environment.
4. **Use Case Scenarios:** Identifying practical use cases for nested virtualization in real-world applications is vital for demonstrating its value. The project will explore scenarios where nested virtualization can enhance development, testing, and production environments.
5. **Documentation and Best Practices:** Providing clear documentation and guidelines for configuring and utilizing KVM nested virtualization on Oracle Virtual Box with CentOS will aid users in overcoming the existing knowledge barriers.

Objective

The primary objective of this project is to systematically investigate and document the process of enabling KVM nested virtualization on Oracle Virtual Box using CentOS. This includes configuring the environment, analysing performance metrics, identifying potential challenges, and providing best practices and recommendations for users.

By addressing these issues, this project aims to contribute to the broader understanding of nested virtualization and its applications in both development and production environments, ultimately empowering users to leverage advanced virtualization techniques effectively.

Hardware and Software Requirement:

- **Hardware Requirement:**

1. Operating System: Windows
2. Processor: Intel i5 and Above
3. RAM: 8GB and Above
4. Virtualization: enabled through BIOS

- **Software Requirement :**

1. Oracle Virtual Box
2. Virtualization Manager
3. CentOS and mintOS

Virtualization in Linux

Virtualization in Linux refers to the creation of virtual versions of physical resources, such as servers, storage devices, or network resources. It allows multiple virtual instances to run on a single physical machine, effectively maximizing resource utilization and providing isolation between different environments.

Here are some key concepts related to virtualization in Linux:

1. Hypervisors: These are the software layers that enable virtualization.

There are two types:

Type 1 (Bare-metal): Runs directly on the hardware (e.g., VMware ESXi, Microsoft Hyper-V).

Type 2 (Hosted): Runs on top of a host operating system (e.g., VMware Workstation, Oracle Virtual Box).

2. KVM (Kernel-based Virtual Machine): A popular virtualization solution for Linux that turns the kernel into a Type 1 hypervisor. KVM allows Linux to run multiple isolated virtual environments called guests.

3. LXC (Linux Containers): A lightweight virtualization method that uses the host kernel to run multiple isolated Linux environments, known as containers. Unlike traditional virtual machines, containers share the same operating system kernel.

4. QEMU: An open-source emulator that works with KVM to provide full system emulation. It allows for the emulation of various hardware platforms.

5. Storage and Networking: Virtualization requires efficient storage solutions (like LVM or Ceph) and virtual networking to ensure that virtual machines or containers can communicate effectively.

6. Management Tools: There are various tools for managing virtual environments, such as OpenStack for cloud infrastructure, or Proxmox and oVirt for more traditional virtualization management.

Virtualization is widely used for server consolidation, testing and development environments, and cloud computing. It enhances flexibility, scalability, and resource management in IT environments.

Nested Virtualization

Nested virtualization allows you to run a virtual machine (VM) inside another VM. This capability is particularly useful in various scenarios, especially in development, testing, and educational environments. Here are some key uses and benefits of nested virtualization:

Key Uses of Nested Virtualization

1. Development and Testing:

- Hypervisor Development: Developers can create and test new hypervisors or virtualization management tools without needing separate physical hardware.
- Cloud Development: Developers can simulate cloud environments locally, testing how applications behave in nested hypervisor setups.

2. Training and Education:

- Learning and Experimentation: Students or professionals can experiment with virtualization technologies and learn about hypervisors, networking, and systems management without needing multiple physical servers.

3. CI/CD Pipelines:

- In continuous integration and continuous deployment (CI/CD) environments, nested virtualization can help create isolated testing environments for different versions of applications.

4. Enhanced Isolation:

- Nested VMs can be useful in multi-tenant environments where additional layers of isolation are required, improving security for different clients or applications.

5. Resource Management:

- It allows for more flexible resource allocation, enabling the hosting of multiple virtualized environments on a single physical server.

6. Remote and Edge Computing:

- In edge computing scenarios, nested virtualization can help deploy lightweight hypervisors on edge devices, allowing for localized management of VMs.

Benefits of Nested Virtualization

1. **Cost Efficiency:** Reduces the need for multiple physical machines, allowing organizations to utilize existing hardware more effectively.
2. **Flexibility:** Supports a variety of testing and development scenarios, making it easier to create complex virtual environments.
3. **Easy Rollback:** Allows for easy snapshots and rollbacks of nested VMs, making it simpler to manage different configurations and states.
4. **Seamless Integration:** Works well with existing virtualization solutions, allowing for a smoother transition and integration into current workflows.
5. **Testing Complex Architectures:** Enables the simulation of complex cloud architectures and distributed systems without needing extensive physical resources.

Nested virtualization is a powerful feature that expands the possibilities of virtualization, making it valuable for developers, educators, and organizations looking to optimize resource utilization while experimenting with virtualization technologies. Whether for testing, training, or deploying complex environments, nested virtualization offers significant advantages in flexibility and efficiency.

Kernel-Based Virtual Machine:

KVM, or Kernel-based Virtual Machine, is a virtualization technology built into the Linux kernel. It allows you to run multiple virtual machines (VMs) on a Linux host, each with its own operating system. Here's a deeper look at KVM and why you might choose to use it:

What is KVM?

- Integration with Linux: KVM turns the Linux kernel into a Type 1 (bare-metal) hypervisor, leveraging the kernel's capabilities to manage system resources efficiently.
- Virtualization Extensions: KVM requires hardware virtualization extensions (Intel VT-x or AMD-V) to operate effectively. These extensions allow better performance by enabling direct execution of guest code on the host CPU.
- QEMU: KVM is often used alongside QEMU, a generic and open-source machine emulator and virtualizer, which provides hardware emulation and additional features for managing VMs.

Key Features of KVM

1. Performance: Since KVM is part of the Linux kernel, it benefits from kernel optimizations, leading to excellent performance compared to other virtualization solutions.
2. Scalability: KVM can support a large number of VMs on a single host, making it suitable for both small and large environments.
3. Flexibility: You can run different operating systems on the VMs, including various Linux distributions and Windows, allowing for diverse workloads.
4. Management Tools: There are various management tools available for KVM, such as libvirt, Virt-Manager, and oVirt, which simplify the creation, management, and monitoring of VMs.
5. Snapshot and Cloning: KVM supports features like snapshots (capturing the state of a VM at a point in time) and cloning (creating copies of VMs), which are useful for backup and development.

Why Use KVM in Linux?

1. **Cost-Effectiveness:** KVM is open-source and included with many Linux distributions, making it a cost-effective solution for virtualization without licensing fees.
2. **Resource Utilization:** By consolidating multiple workloads on fewer physical servers, KVM helps maximize resource utilization, reducing hardware costs and energy consumption.
3. **Cloud Compatibility:** KVM is widely used in cloud infrastructure solutions (e.g., OpenStack), making it suitable for organizations looking to build cloud environments.
4. **Integration with Existing Infrastructure:** For organizations already using Linux, KVM integrates well with existing tools and workflows, reducing the learning curve and setup time.

Overall, KVM is a powerful and versatile virtualization solution for Linux that can meet a wide range of needs, from personal projects to enterprise-level deployments.

Implementation with Output:

Step 1: Download Oracle Virtual Box and Download CentOS ISO in it

- Go to the CentOS official website (www.centos.org) and download the latest version of CentOS (e.g., CentOS 9) in ISO format.
- Make sure to download the correct architecture (32-bit or 64-bit) that matches your system.

Step 2: Create a New Virtual Machine in Virtual Box

- Open Oracle Virtual Box and click on "New" to create a new virtual machine.
- Enter a name for your virtual machine (e.g., "CentOS 9") and select "Linux" as the type.
- Choose "Red Hat (64-bit)" as the version and click "Next".
- Allocate at least 2048 MB of RAM and click "Next".
- Create a new virtual hard disk by clicking "Create a virtual hard disk now" and follow the wizard to create a 20 GB disk.
- Click "Create" to create the virtual machine.

Step 3: Configure Virtual Machine Settings

- Select the virtual machine and click "Settings".
- In the "Storage" tab, select the empty CD/DVD drive and click on the CD/DVD icon to browse for the CentOS ISO file.
- Select the ISO file and click "OK".
- In the "Display" tab, select "VBoxSVGA" as the graphics controller and set the video memory to 128 MB.
- Click "OK" to save the changes.

Step 4: Start the Virtual Machine and Begin Installation

- Click "Start" to power on the virtual machine.
- The CentOS installation wizard will boot up. Select your language and click "Continue".
- Choose "Install CentOS 8" and click "Continue".

- Accept the license agreement and click "Continue".
- Select the installation destination (e.g., the 20 GB virtual hard disk) and click "Continue".
- Configure the network settings (e.g., set the IP address, subnet mask, gateway, and DNS servers) and click "Continue".
- Set the root password and create a user account (optional) and click "Continue".
- Review the installation summary and click "Begin Installation".

Step 5: Complete the Installation

- The installation process will take around 10-15 minutes to complete.
- Once the installation is complete, click "Reboot" to restart the virtual machine.
- Remove the CentOS ISO from the virtual machine's CD/DVD drive by clicking "Devices" > "Optical Drives" > "Remove disk from virtual drive".
- The virtual machine will boot up into the CentOS GUI.

Step 6: Initial System Configuration

- Log in to the system using the root password or the user account you created during installation.
- Configure the system date and time, and set the timezone.

Step 7: Create Virtual Machines in Linux Using KVM (Kernel-based Virtual Machine)

Make sure that your system has the hardware virtualization extensions: For Intel-based hosts, verify the CPU virtualization extension [vmx] are available using following command.

```
[root@server ~]# grep -e 'vmx' /proc/cpuinfo
```

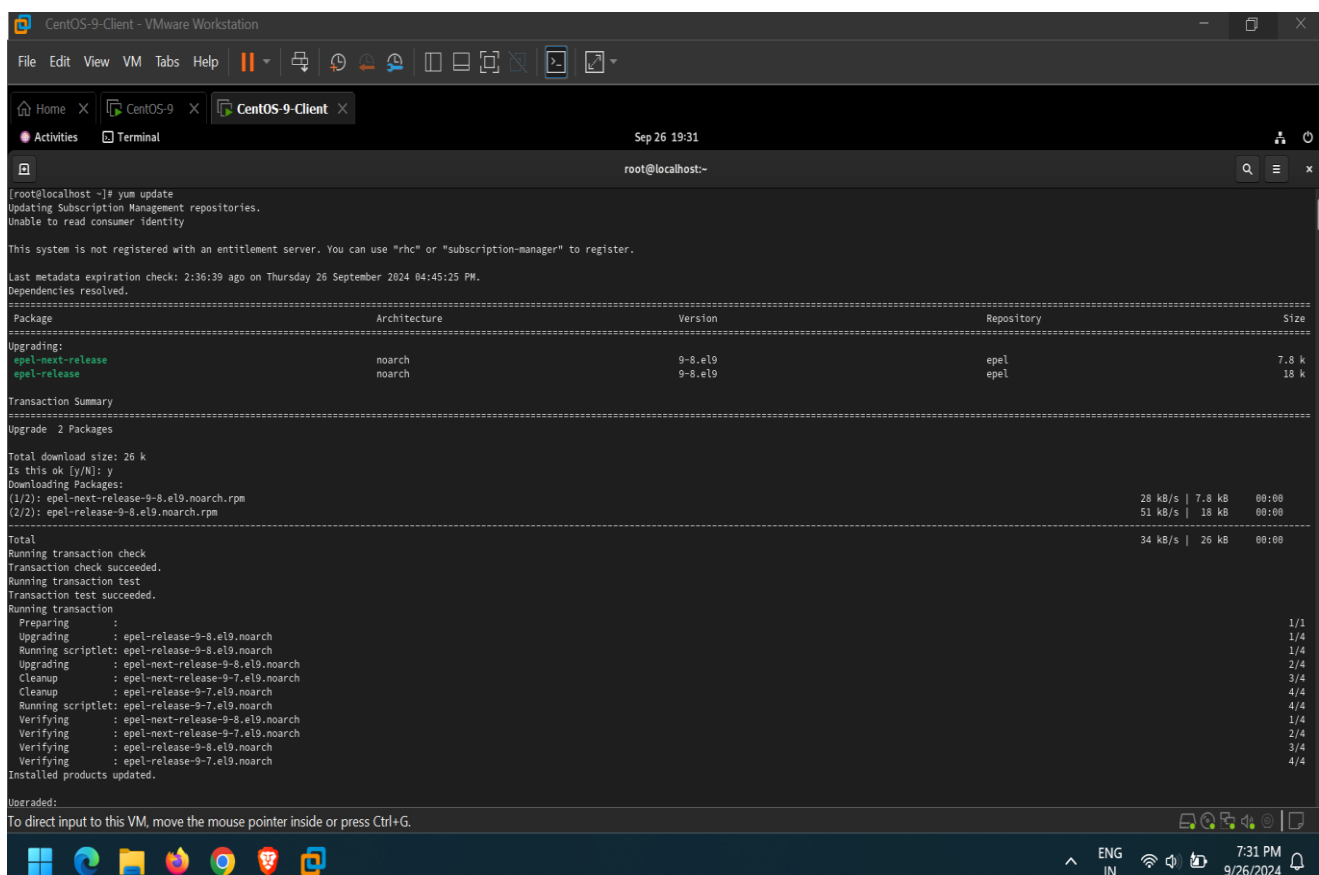
For AMD-based hosts, verify the CPU virtualization extension [svm] are available.

```
[root@server ~]# grep -e 'svm' /proc/cpuinfo
```

If there is no output make sure that virtualization extensions is enabled in BIOS. Verify that KVM modules are loaded in the kernel “it should be loaded by default”.

```
[root@server ~]# lsmod | grep kvm
```

Before starting, you will need the root account or non-root user with sudo privileges configured on your system and also make sure that your system is up-to-date.



The screenshot shows a terminal window titled "CentOS-9-Client - VMware Workstation". The terminal output is as follows:

```
[root@localhost ~]# yum update
Updating Subscription Management repositories.
Unable to read consumer identity

This system is not registered with an entitlement server. You can use "rhc" or "subscription-manager" to register.

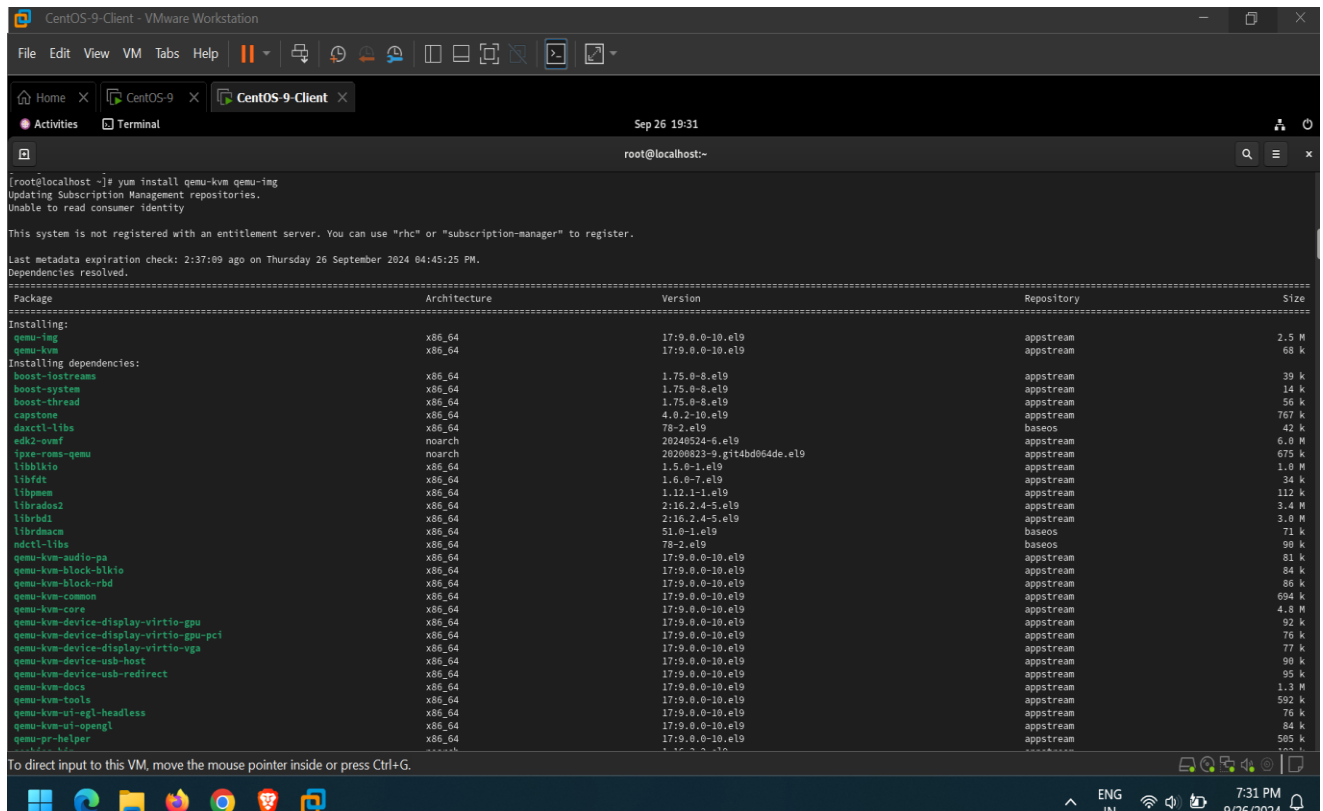
Last metadata expiration check: 2:36:39 ago on Thursday 26 September 2024 04:45:25 PM.
Dependencies resolved.
=====
Package                                Architecture    Version          Size
=====
Upgrading:
epel-next-release                       noarch          9-8.el9          7.8 k
epel-release                            noarch          9-8.el9          18 k
=====
Transaction Summary
=====
Upgrade 2 Packages

Total download size: 26 k
Is this ok [y/N]: y
Downloading Packages:
(1/2): epel-next-release-9-8.el9.noarch.rpm                28 kB/s | 7.8 kB  00:00
(2/2): epel-release-9-8.el9.noarch.rpm                     51 kB/s | 18 kB  00:00
-----
Total                                                    34 kB/s | 26 kB  00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing      : epel-release-9-8.el9.noarch                                1/1
  Upgrading      : epel-release-9-8.el9.noarch                                1/4
  Running scriptlet: epel-release-9-8.el9.noarch                                1/4
  Upgrading      : epel-next-release-9-8.el9.noarch                          2/4
  Cleanup        : epel-next-release-9-7.el9.noarch                          3/4
  Cleanup        : epel-release-9-7.el9.noarch                               4/4
  Running scriptlet: epel-release-9-7.el9.noarch                               4/4
  Verifying      : epel-next-release-9-8.el9.noarch                          1/4
  Verifying      : epel-next-release-9-7.el9.noarch                          2/4
  Verifying      : epel-release-9-8.el9.noarch                               3/4
  Verifying      : epel-release-9-7.el9.noarch                               4/4
Installed products updated.

Upgraded:
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.
```

Step 1: KVM Installation and Deployment

1. We will install qemu-kvm and qemu-img packages at first. These packages provide the user-level KVM and disk image manager.



```
[root@localhost ~]# yum install qemu-kvm qemu-img
Updating Subscription Management repositories.
Unable to read consumer identity

This system is not registered with an entitlement server. You can use "rhcm" or "subscription-manager" to register.

Last metadata expiration check: 2:37:09 ago on Thursday 26 September 2024 04:45:25 PM.
Dependencies resolved.
=====
Package                                Architecture      Version           Repository        Size
-----
Installing:
qemu-img                               x86_64            17.9.0-10.el9    appstream         2.5 M
qemu-kvm                               x86_64            17.9.0-10.el9    appstream         68 k
Installing dependencies:
boost-iostreams                        x86_64            1.75.0-8.el9     appstream         39 k
boost-system                           x86_64            1.75.0-8.el9     appstream         14 k
boost-thread                            x86_64            1.75.0-8.el9     appstream         56 k
capstone                               x86_64            4.0.2-10.el9     appstream         767 k
daxctl-libs                            x86_64            78-2.el9         baseos            42 k
edk2-ovmf                              noarch            20240524-6.el9   appstream         6.0 M
ipxe-roms-qemu                         noarch            20200823-9.git4bd064de.el9 appstream         675 k
libblkid                               x86_64            1.5.0-1.el9      appstream         1.9 M
libfdt                                  x86_64            1.0.6-7.el9      appstream         34 k
libfmem                                 x86_64            1.12.1-1.el9     appstream         112 k
librados2                              x86_64            2:16.2.4-5.el9   appstream         3.4 M
librbd1                                x86_64            2:16.2.4-5.el9   appstream         3.0 M
librbdapi                              x86_64            51.0-1.el9       appstream         71 k
ndctl-libs                             x86_64            78-2.el9         baseos            98 k
qemu-kvm-audio-pa                      x86_64            17.9.0-10.el9    appstream         81 k
qemu-kvm-block-blkio                   x86_64            17.9.0-10.el9    appstream         84 k
qemu-kvm-block-rbd                     x86_64            17.9.0-10.el9    appstream         86 k
qemu-kvm-common                        x86_64            17.9.0-10.el9    appstream         694 k
qemu-kvm-core                          x86_64            17.9.0-10.el9    appstream         4.9 M
qemu-kvm-device-display-virtio-gpu     x86_64            17.9.0-10.el9    appstream         92 k
qemu-kvm-device-display-virtio-gpu-pci x86_64            17.9.0-10.el9    appstream         76 k
qemu-kvm-device-display-virtio-vga     x86_64            17.9.0-10.el9    appstream         77 k
qemu-kvm-device-usb-host               x86_64            17.9.0-10.el9    appstream         98 k
qemu-kvm-device-usb-redirect           x86_64            17.9.0-10.el9    appstream         95 k
qemu-kvm-docs                          x86_64            17.9.0-10.el9    appstream         1.3 M
qemu-kvm-tools                         x86_64            17.9.0-10.el9    appstream         592 k
qemu-kvm-ui-egl-headless               x86_64            17.9.0-10.el9    appstream         76 k
qemu-kvm-ui-opengl                     x86_64            17.9.0-10.el9    appstream         84 k
qemu-pr-helper                         x86_64            17.9.0-10.el9    appstream         505 k
=====
```

2. Now, you have the minimum requirement to deploy virtual platform on your host, but we also still have useful tools to administrate our platform such as:

1. virt-manager provides a GUI tool to administrate your virtual machines.
2. libvirt-client provides a CL tool to administrate your virtual environment this tool called virsh.
3. virt-install provides the command “virt-install” to create your virtual machines from CLI.
4. libvirt provides the server and host side libraries for interacting with hypervisors and host systems.

Let's install these above tools using the following command.

```
root@localhost ~#
root@localhost ~# yum install virt-manager libvirt libvirt-client
Updating Subscription Management repositories.
Unable to read consumer identity

This system is not registered with an entitlement server. You can use "rhc" or "subscription-manager" to register.

Last metadata expiration check: 2:38:26 ago on Thursday 26 September 2024 04:45:25 PM.
Dependencies resolved.
=====
Package                               Architecture      Version            Repository         Size
-----
Installing:
libvirt                               x86_64            10.5.0-5.el9       appstream          22 k
libvirt-client                        x86_64            10.5.0-5.el9       appstream          448 k
virt-manager                          noarch            4.1.0-5.el9        appstream          585 k
Installing dependencies:
gnutls-dane                          x86_64            3.8.3-4.el9        appstream          22 k
gnutls-utils                         x86_64            3.8.3-4.el9        appstream          293 k
gsproxy                              x86_64            0.8.4-7.el9        baseos             110 k
gtk-vnc2                             x86_64            1.3.0-2.el9        appstream          41 k
gvc                                  x86_64            1.3.0-2.el9        appstream          102 k
libburn                             x86_64            1.5.4-4.el9        appstream          173 k
libev                                x86_64            4.33-6.el9         baseos             52 k
libisoburn                          x86_64            1.5.4-4.el9        appstream          416 k
libiso9660                          x86_64            1.5.4-4.el9        appstream          222 k
libimdb                             x86_64            1.20.2-2.el9       appstream          171 k
libnfsidmap                         x86_64            1:1.5.4-27.el9     baseos             61 k
libnfs                                x86_64            0.9.1-3.20211126git1ff6f43.el9 appstream          185 k
libvirt-libs                         x86_64            10.5.0-5.el9       baseos             14 k
libvirt-client-guest                x86_64            10.5.0-5.el9       appstream          47 k
libvirt-daemon                      x86_64            10.5.0-5.el9       appstream          213 k
libvirt-daemon-common               x86_64            10.5.0-5.el9       appstream          140 k
libvirt-daemon-config-network       x86_64            10.5.0-5.el9       appstream          29 k
libvirt-daemon-config-nwfilter      x86_64            10.5.0-5.el9       appstream          41 k
libvirt-daemon-driver-interface     x86_64            10.5.0-5.el9       appstream          219 k
libvirt-daemon-driver-network       x86_64            10.5.0-5.el9       appstream          272 k
libvirt-daemon-driver-nodedev       x86_64            10.5.0-5.el9       appstream          243 k
libvirt-daemon-driver-nwfilter      x86_64            10.5.0-5.el9       appstream          256 k
libvirt-daemon-driver-qemu          x86_64            10.5.0-5.el9       appstream          800 k
libvirt-daemon-driver-secret        x86_64            10.5.0-5.el9       appstream          216 k
libvirt-daemon-driver-storage       x86_64            10.5.0-5.el9       appstream          22 k
libvirt-daemon-driver-storage-core  x86_64            10.5.0-5.el9       appstream          277 k
libvirt-daemon-driver-storage-lscsi x86_64            10.5.0-5.el9       appstream          37 k
libvirt-daemon-driver-storage-iscsi x86_64            10.5.0-5.el9       appstream          34 k
=====
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.
```

3. For RHEL/CentOS7 users, also still having additional package groups such as: Virtualization Client, Virtualization Platform and Virtualization Tools to install.

```
root@localhost ~#
root@localhost ~# yum groupinstall virtualization-client virtualization-platform virtualization-tools
Updating Subscription Management repositories.
Unable to read consumer identity

This system is not registered with an entitlement server. You can use "rhc" or "subscription-manager" to register.

Last metadata expiration check: 4:05:44 ago on Thursday 26 September 2024 04:45:25 PM.
Dependencies resolved.
=====
Package                               Architecture      Version            Repository         Size
-----
Installing group/module packages:
libguestfs                          x86_64            1:1.50.2-1.el9     appstream          1.1 M
virt-install                         noarch            4.1.0-5.el9        appstream          42 k
virt-top                            x86_64            1.1.1-9.el9        appstream          715 k
virt-viewer                         x86_64            11.0-1.el9         appstream          296 k
virt-who                            noarch            1.31.28-1.el9      appstream          448 k
virtio-win                          noarch            1.9.15-4.el9       appstream          157 M
Installing dependencies:
augeas-libs                         x86_64            1.14.1-2.el9       appstream          424 k
dhcpcd-libs                        x86_64            12:4.4.2-19.b1.el9 baseos             790 k
dhcpcd-common                      noarch            12:4.4.2-19.b1.el9 baseos             129 k
hiredis                             x86_64            1.6-1.el9          appstream          43 k
hivex-libs                         x86_64            1.3.24-1.el9       appstream          42 k
ipcalc                             x86_64            1.0.0-5.el9        baseos             42 k
libguestfs-appliance               x86_64            1:1.50.2-1.el9     appstream          2.2 M
nfs-utils                          x86_64            4.0.26-4.el9       baseos             222 k
supermin                           x86_64            5.3.3-1.el9        appstream          758 k
syslinux                           x86_64            6.04-0.20.el9      baseos             571 k
syslinux-extlinux                  x86_64            6.04-0.20.el9      baseos             131 k
syslinux-extlinux-nonlinux         noarch            6.04-0.20.el9      baseos             395 k
syslinux-nonlinux                  noarch            6.04-0.20.el9      baseos             571 k
Installing weak dependencies:
geolitez-city                      noarch            20191217-6.el9     appstream          23 M
geolitez-country                   noarch            20191217-6.el9     appstream          1.6 M
Installing Groups:
Virtualization Client
Virtualization Platform
Virtualization Tools
=====
Transaction Summary
-----
Install 21 Packages
Total download size: 100 M
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.
```

The virtualization daemon which manage all of the platform is “libvirtd”. Let’s restart it.

```
[root@localhost ~]#  
[root@localhost ~]# systemctl restart libvirtd  
[root@localhost ~]#
```

After restarting the daemon, then check its status by running following command.

```
[root@localhost ~]# systemctl status libvirtd  
● libvirtd.service - libvirt legacy monolithic daemon  
   Loaded: loaded (/usr/lib/systemd/system/libvirtd.service; enabled; preset: disabled)  
   Active: active (running) since Thu 2024-09-26 20:56:52 IST; 21s ago  
TriggeredBy: ● libvirtd-admin.socket  
              ● libvirtd.socket  
              ● libvirtd-ro.socket  
   Docs: man:libvirtd(8)  
         https://libvirt.org/  
  Main PID: 4071 (libvirtd)  
    Tasks: 22 (limit: 32768)  
  Memory: 50.5M  
     CPU: 1.154s  
   CGroup: /system.slice/libvirtd.service  
           └─3958 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default.conf --leasefile-ro --dhcp-script=/usr/libexec/libvirt_leaseshelper  
           └─3959 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default.conf --leasefile-ro --dhcp-script=/usr/libexec/libvirt_leaseshelper  
           └─4071 /usr/sbin/libvirtd --timeout 120
```

Now, let’s switch to the next section to create our virtual machines.

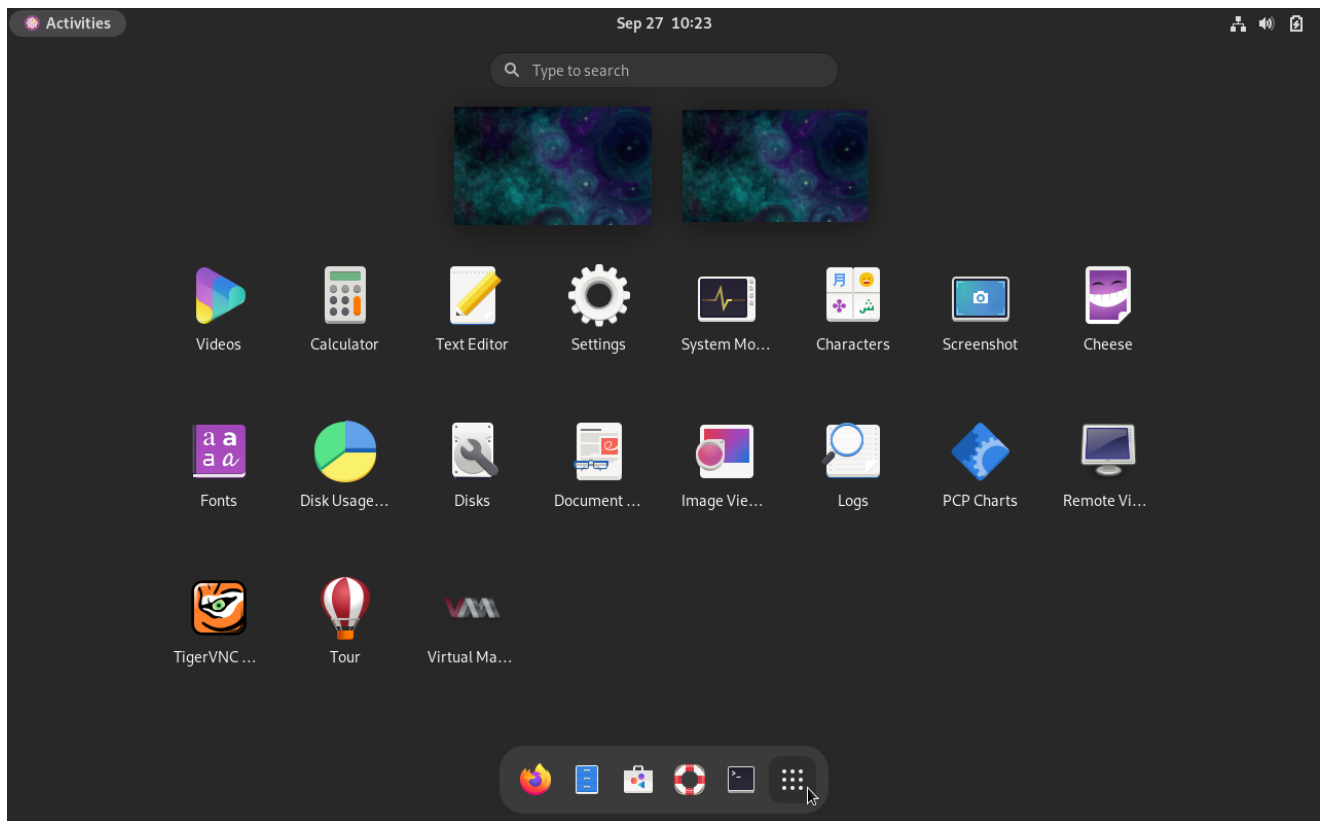
Step 2: Create VMs using KVM

As we mentioned early, we have some useful tools to manage our virtual platform and creating virtual machines. One of this tools called [virt-manager] which we use in the next section.

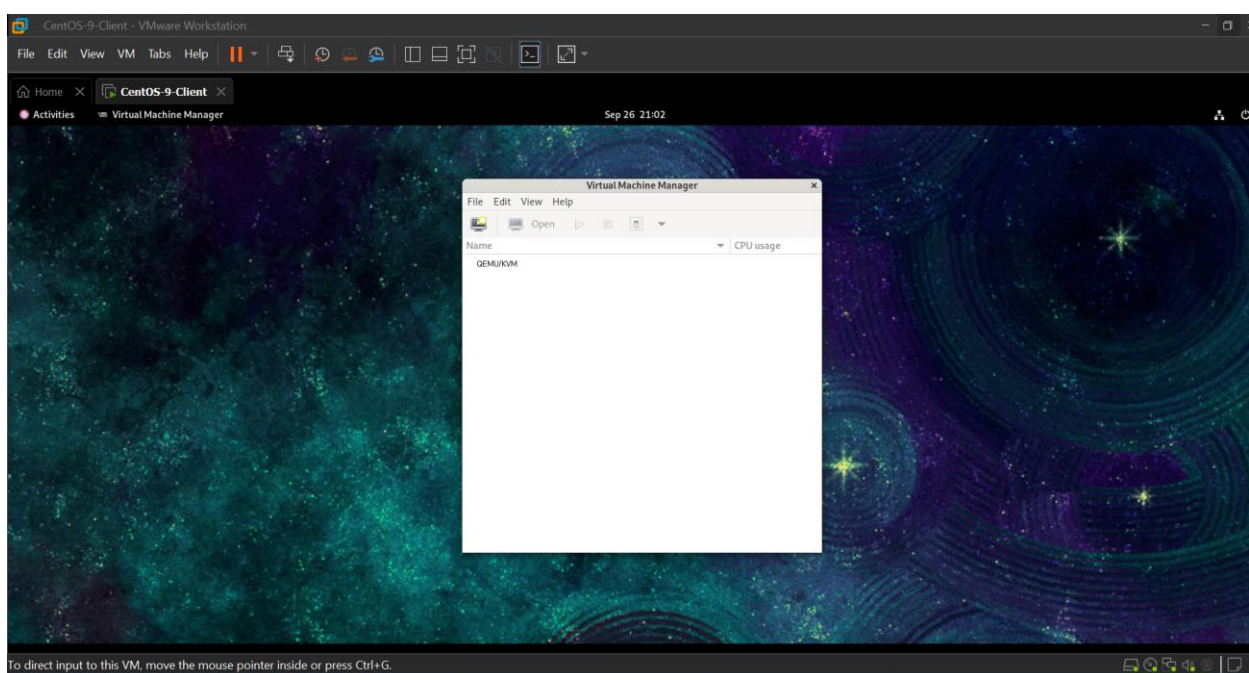
6. Although virt-manager is a GUI based tool, we also could launch/start it from terminal as well as from GUI.

```
[root@server ~]#virt-manager
```

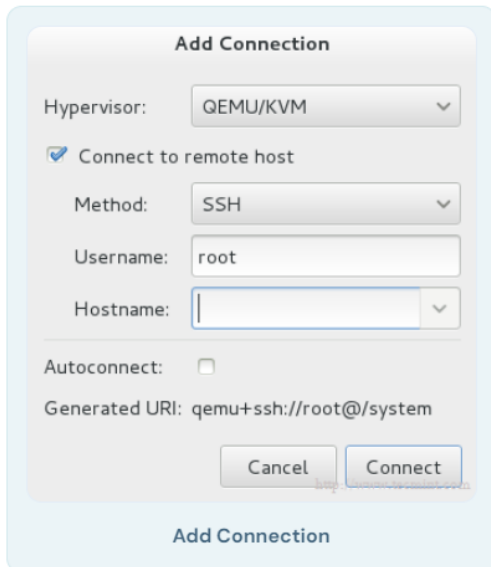

Using GNOME



7. After starting the tool, this window will appear.



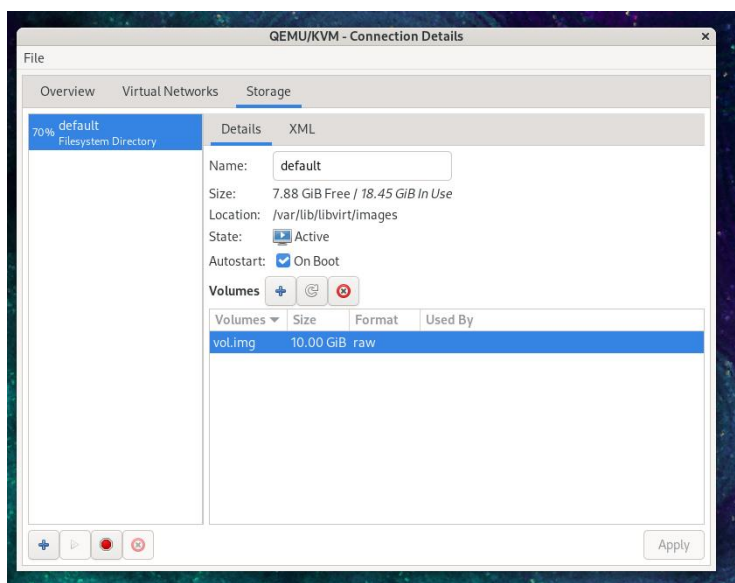
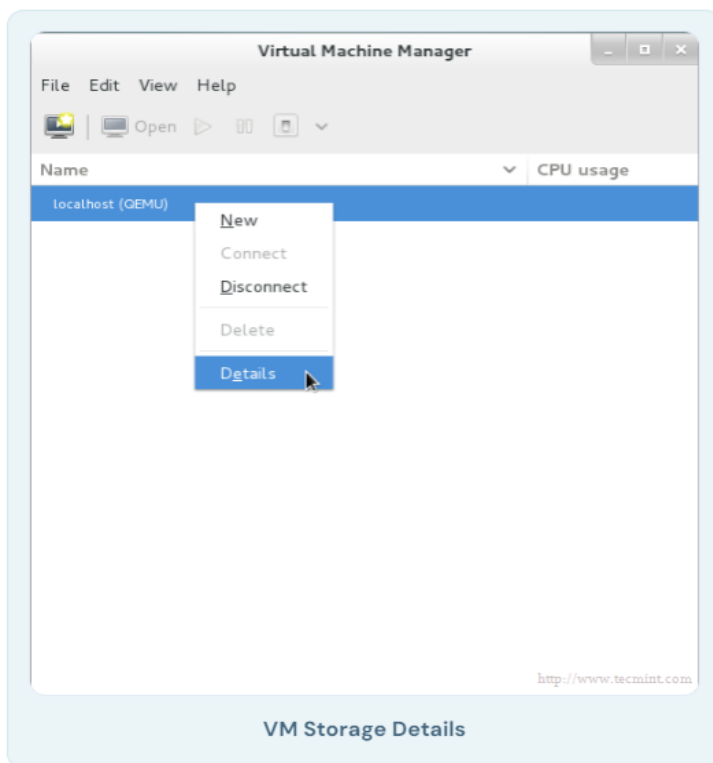
8. By default you will find manager is connected directly to localhost, fortunately you could use the same tool to manage another host remotely. From “File” tab, just select “Add Connection” and this window will appear.



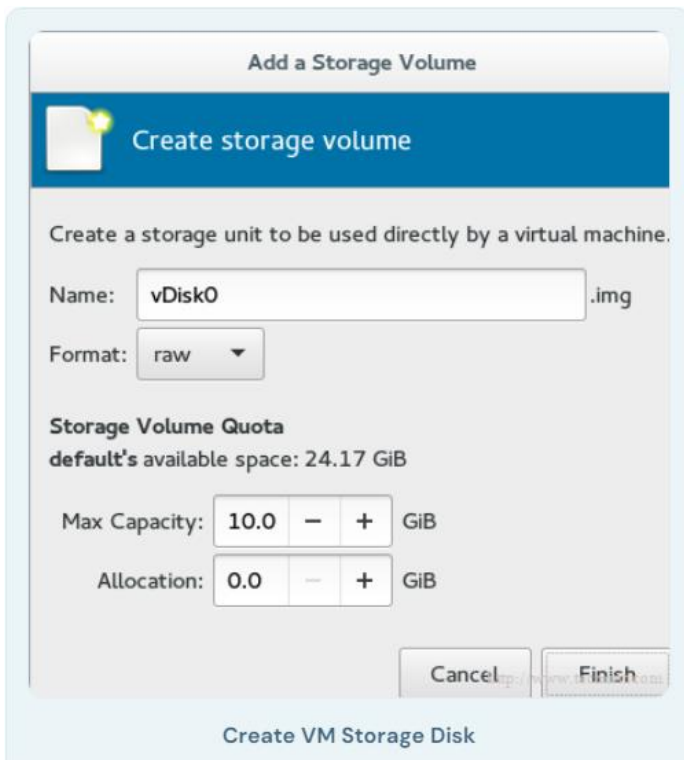
Check “Connect to remote host” option then provide Hostname/IP of the remote server. If you need establishing connection to the remote host at every time the manager starting, just check “Auto Connect” option.

9. Let's return to our localhost, before creating new virtual machine you should decide where will the files be stored?! In other words, you should create the Volume Disk (Virtual disk / Disk image) for your virtual machine.

By Right clicking on localhost and selecting "Details" and then select "Storage" tab.



10. Next, press “New Volume” button, then enter the name of your new virtual disk (Volume Disk) and enter the size which you want/need in the “Max Capacity” section.



Add a Storage Volume

Create storage volume

Create a storage unit to be used directly by a virtual machine.

Name: vDisk0 .img

Format: raw

Storage Volume Quota
default's available space: 24.17 GiB

Max Capacity: 10.0 - + GiB

Allocation: 0.0 - + GiB

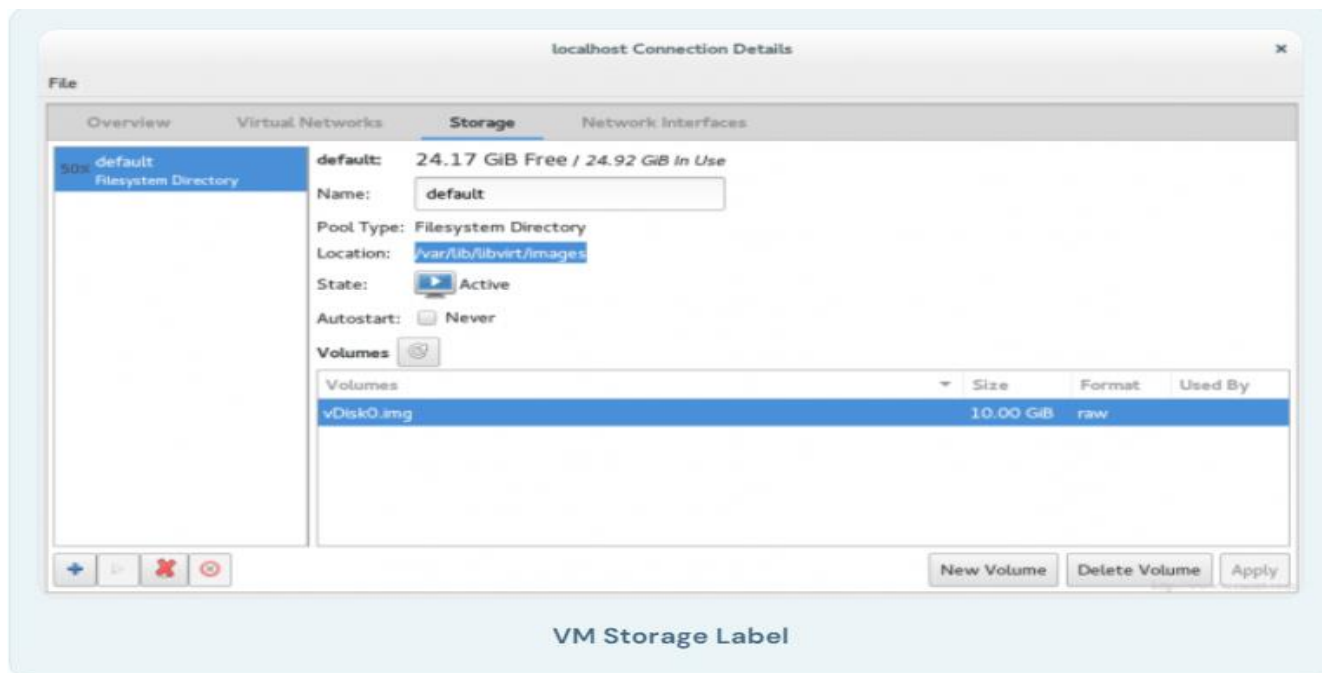
Cancel Finish

Create VM Storage Disk

The allocation size is the actual size for your disk which will be allocated immediately from your physical disk after finishing the steps.

For example, you created virtual disk with size 60G, but you have used actually only 20G, using this technology the allocated size from your physical hard disk will be 20G not 60G.

11. You will note that a label of the new Volume Disk has been appeared in the list.

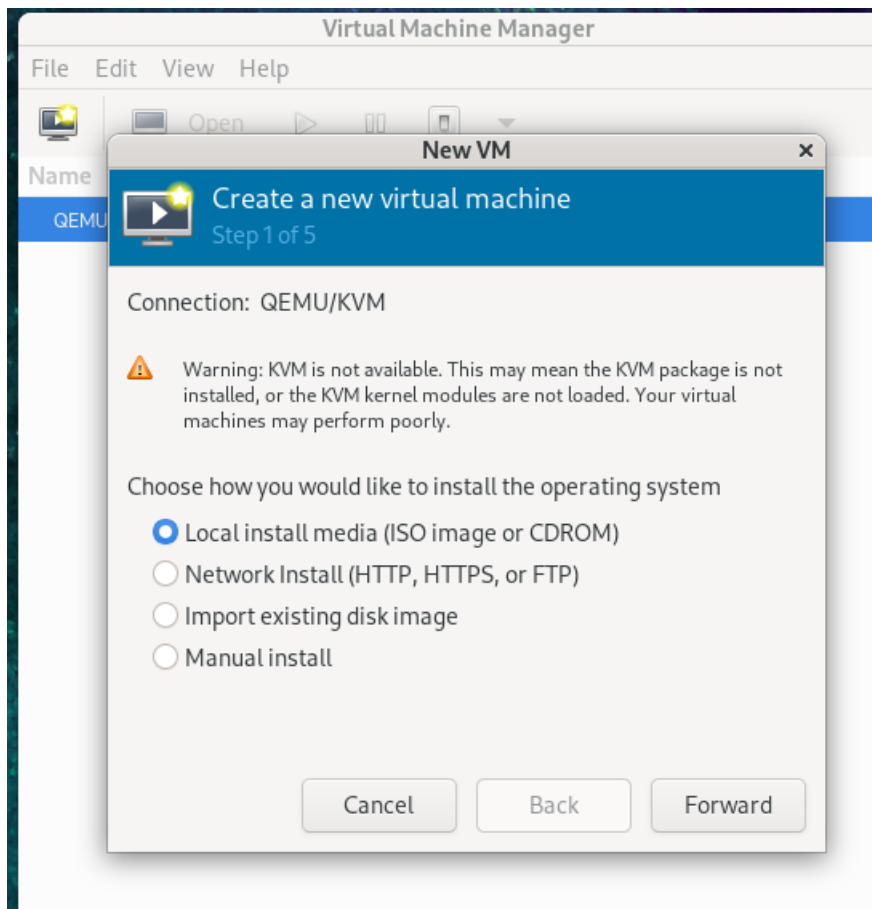


You should also notice the path of the new disk image (Volume Disk), by default it will be under /var/lib/libvirt/images, you can verify it using the following command.

```
[root@server Downloads]# ls -l /var/lib/libvirt/images
```

```
-rw-----. 1 root root 10737418240 Jan  3 16:47 vm1Storage.img
```

12. Now, we're ready to create our virtual machine. Let's hit the button "VM" in the main window, this wizard window will be appear.

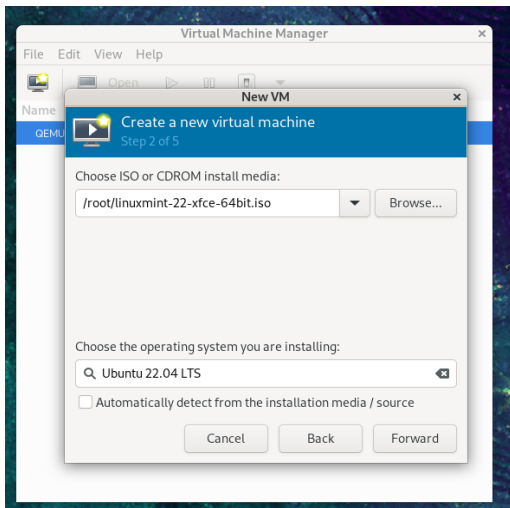


Select the installation method which you will use to create the virtual machine. For now we will use Local install media, later we will discuss the remaining methods.

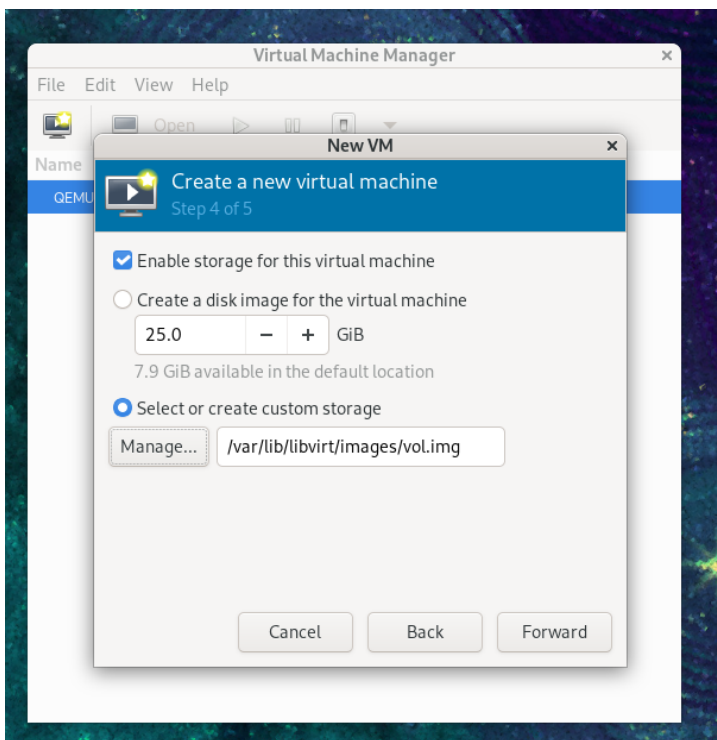
13. Now it's time to specify which Local install media to be used, we have two options:

1. From physical [CDROM/DVD].
2. From ISO image.

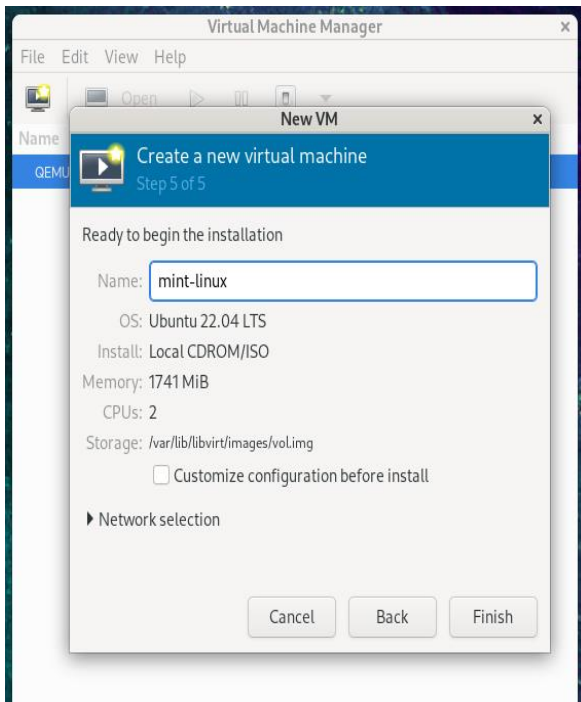
For our tutorial, let's use ISO image method, so you should provide the path of your ISO image.



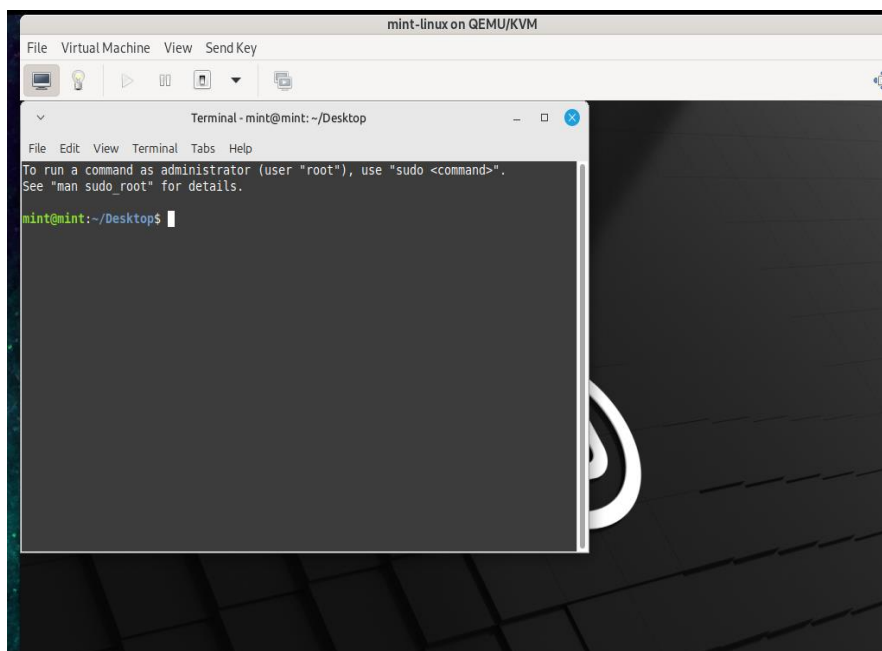
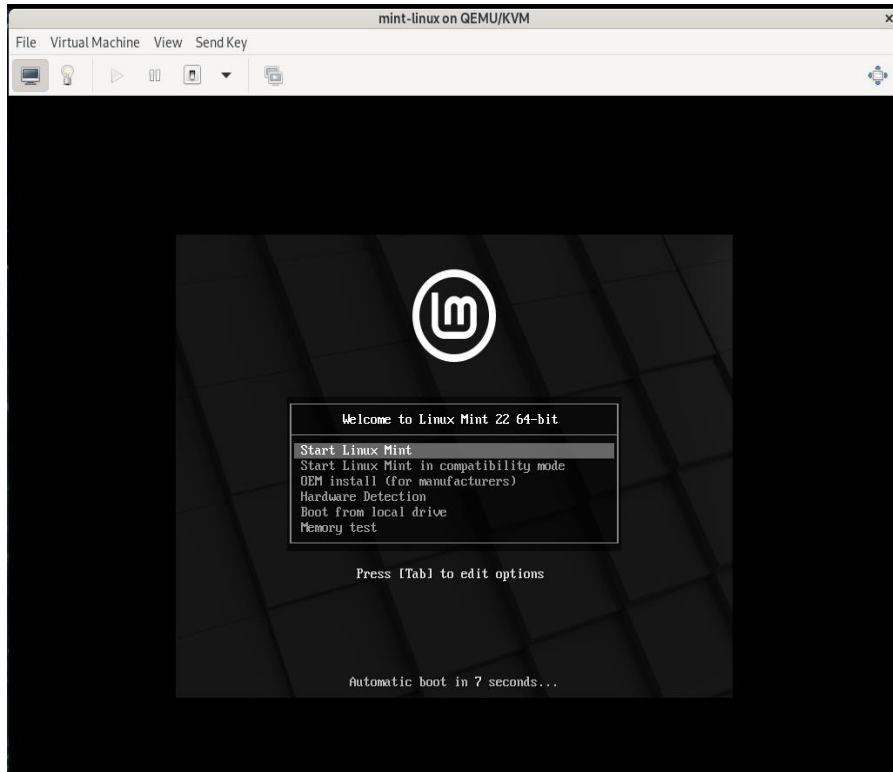
14. The storage has return back, we will use the virtual disk which we have created early to install virtual machine on it. It will be as shown.



15. The final step which ask you about the name of your virtual machine and another advanced options let's talk about it later.



If you like to change some configuration or doing some customization just check “Customize configuration before install” option. Then click finish and wait seconds, control console will appear for your Guest OS to manage it



Conclusion

The project successfully demonstrates the implementation of nested virtualization using Oracle Virtual Box and KVM on a CentOS guest machine. By enabling nested virtualization, we were able to run a hypervisor (KVM) within a Virtual Box VM, and subsequently create and manage virtual machines inside this nested environment. This setup provides a powerful framework for testing and experimenting with multi-layered virtualization environments, particularly for development, testing, and educational purposes.

Through careful configuration of system resources, network bridging, and virtualization tools, this project shows how nested virtualization can extend the capabilities of standard virtual machines, simulating complex infrastructures on a single physical host. The hands-on experience gained from this project serves as a solid foundation for understanding advanced concepts in virtualization and cloud technologies, with practical applications in cloud computing, infrastructure as code, and software-defined networking.