**Virtualization setup document:**

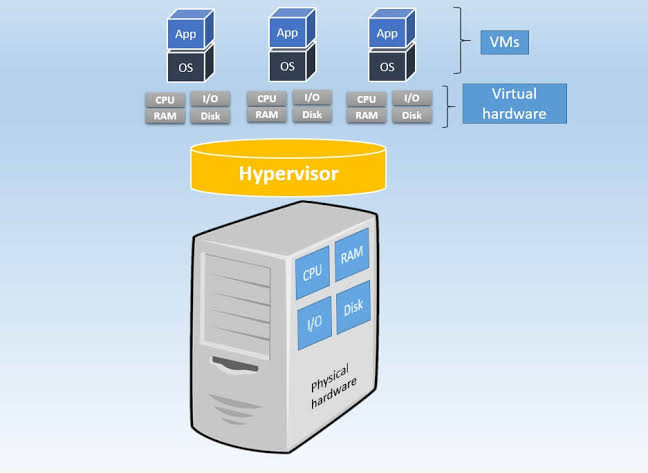
* **Chapret1: Virtualization setup with KVM on physical server.**
* **Chapter2: Physical server Configuration minimum requirements.**
* **Chapter3: Install the CentOS Base System on physical server.**
* **Chapter4: Configuration of yum repository.**
* **Chapter5: Install KVM Hypervisor on top of physical server.**
* **Chapter6: Configure Storage Volumes/Disk Images**
* **Chapter7:Creting a directory-based storage pool with virsh**
* **Chapter8: virsh commands cheat sheet to manage KVMguest virtual machines**
* **Chapter9: How to Add Memory, vCPU, Hard Disk to Linux KVM Virtual Machine**
* **Chapter 10: Snapshot and clone management for vm in kvm**
* **Chapter11: Convert Disk File From Raw To Qcow2.**

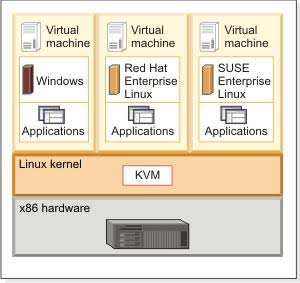
**Chapter1:**

**Virtualization setup with KVM on physical server.**

**Physical server** is a single-tenant platform with dedicated resources, a hypervisor can support multiple **virtual servers**, allowing multiple applications to run simultaneously and share **physical** hardware capacity amongst themselves

**KVM**: Kernel-based Virtual Machine (KVM) is an [open source](https://www.redhat.com/en/topics/open-source/what-is-open-source) [virtualization](https://www.redhat.com/en/topics/virtualization/what-is-virtualization) technology built into Linux. Specifically, KVM [hypervisor](https://www.redhat.com/en/topics/virtualization/what-is-a-hypervisor) that allows a host machine to run multiple, isolated virtual environments called guests or virtual machines (VMs).





**Chapter2:**

**Physical server Configuration minimum requirements:**

## **Hardware** **requirements**:

Cpu –20core cpus

Memory (RAM) --200gb

Hard disk —500gb,500gb,500gb

## **Software requirements**

Operating system (Linux –Centos7)

**Network requirements:**

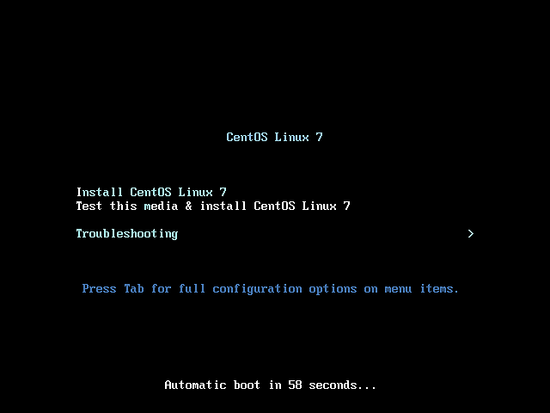
IP address type,Ports:

NIC type-(VMXNET3 (if the configuration server is a VMware VM))

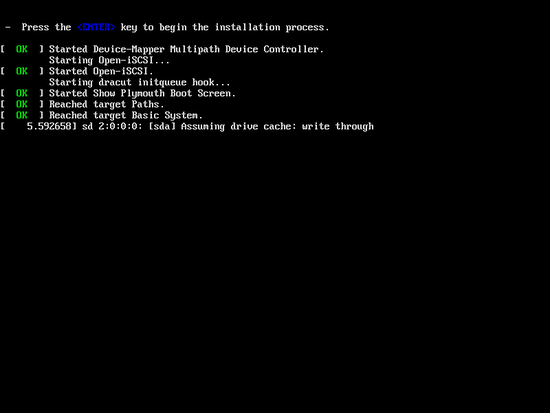
**Chapter3:**

## **Install the CentOS Base System on physical server.**

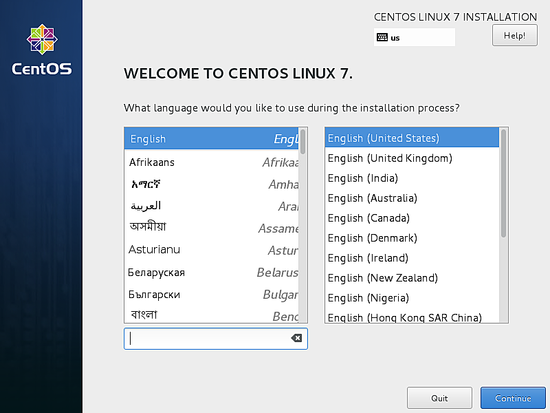
Boot from your CentOS 7 DVD. Select *Install CentOS 7.*

[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-1_.png)

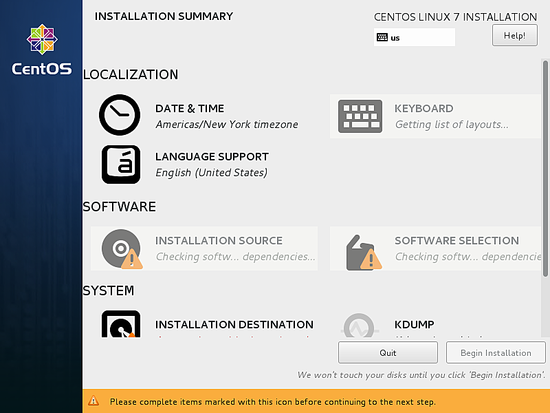
Next press *ENTER.*

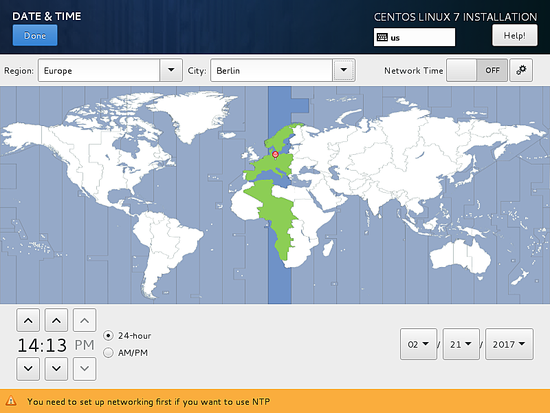
[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-2_.png)

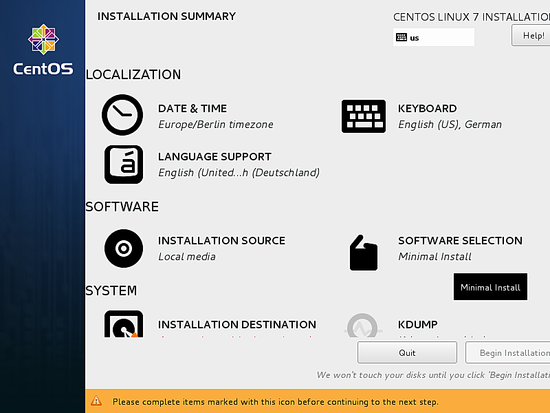
Choose the installation language. I am using *English* with *English United States* here, so I just press Continue:

[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-3_.png)

Next, we get the following screen:

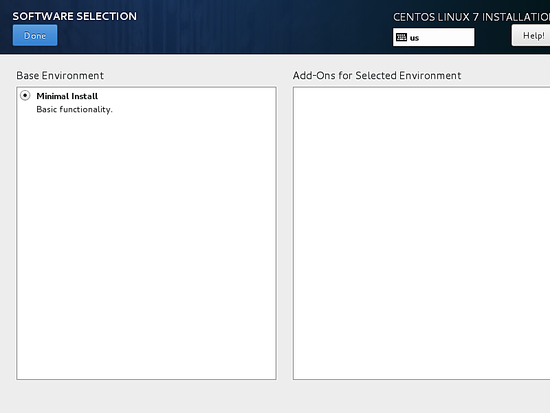
[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-4_.png)

[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-5_.png)

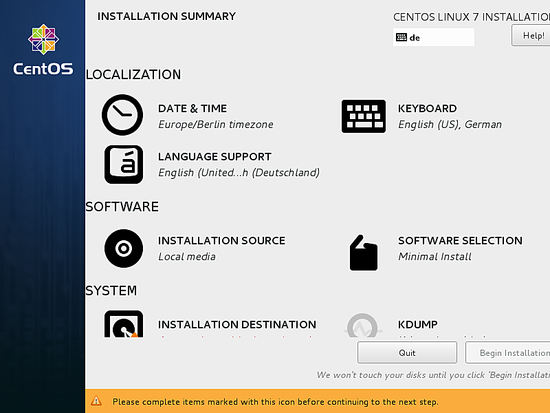
[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-14_.png)

Next, you will see that the installation source is "*Auto-detected installation media"*, if you have any other source of installation like a network install then you can set the path in the "*On the network"*  field with and without proxy from *Proxy Setup*. Additionally, we can add *Additional repositories* if needed. Then press *Done.*

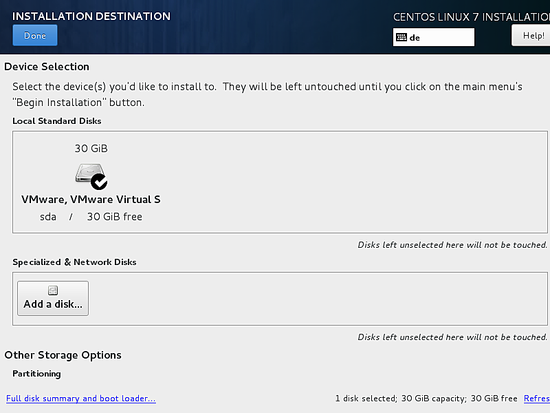
We can add more software packages if required (when you use the install DVD and not minimal installer). In my case, I need only a *Minimal Install(Basic Functionality)* setup to install software later with yum, so I will save the settings by pressing *Done*.

[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-16_.png)

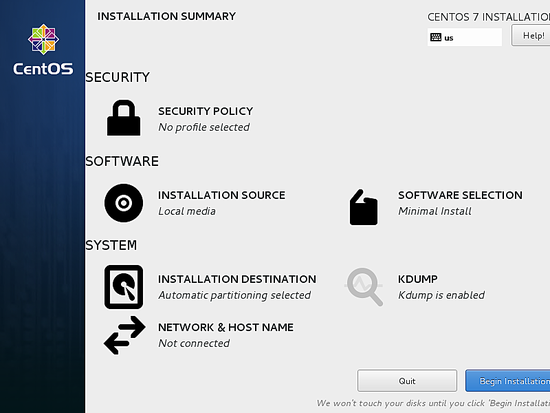
We move towards *SYSTEM* and select *INSTALLATION DESTINATION*.

[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-17_.png)

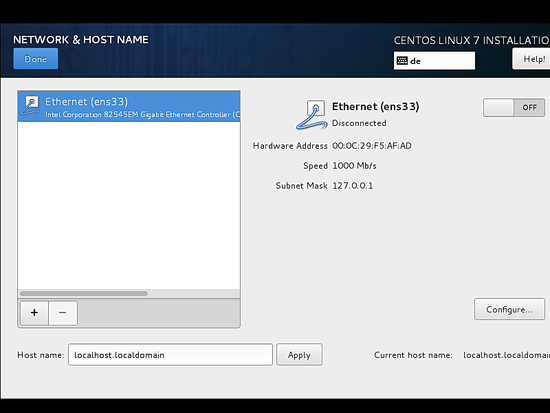
When you need a manual partitioning scheme then you can select *I will configure partitioning*, in my case I will prefer *Automatically configure partitioning* after that press *Done*.

[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-18_.png)

Next, we will customize our *NETWORK & HOSTNAME* by selecting that.

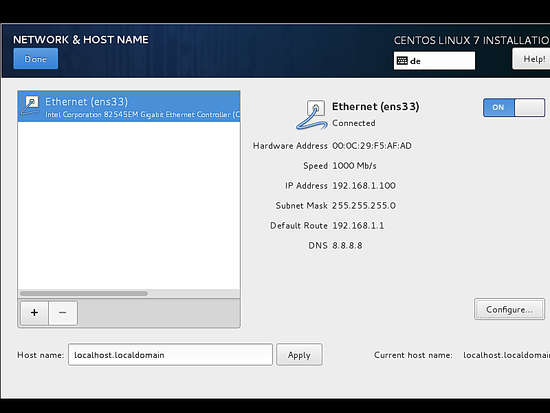
[](https://www.howtoforge.com/images/centos_72_server/big/centos_20.png)

I will use the hostname *server1.example.com* and customize the ethernet setup by pressing *Configure*.

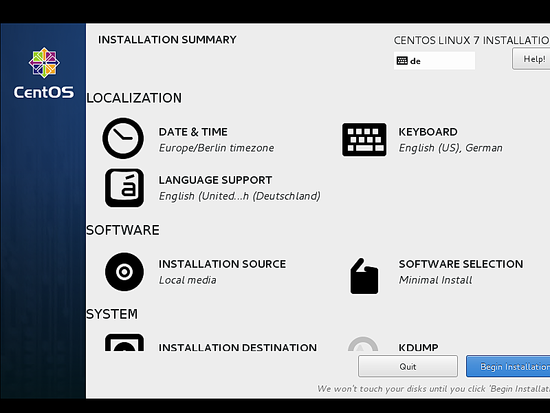
[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-19_.png)

The next screen will look like this, now we add the *IPv4 Settings* by selecting the same. If you have IPv6 then you add it in *IPv6 Settings*.

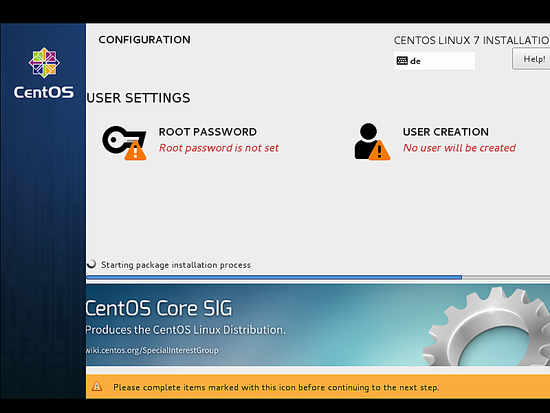
Next, we have to turn the connection *ON* as shown in the screenshot below. Further press *Done*.

[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-25_.png)

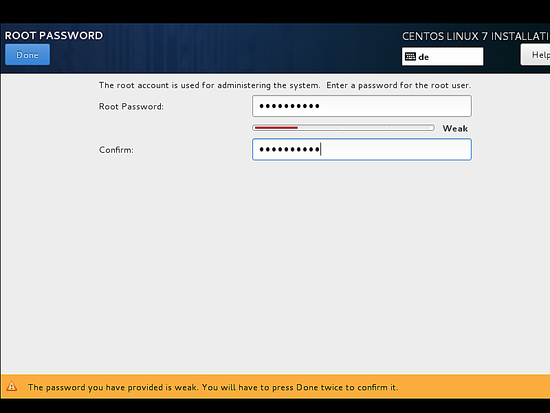
Now we are ready to start the installation process, you might want to crosscheck all the settings and then press *Begin Installation*.

[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-26_.png)

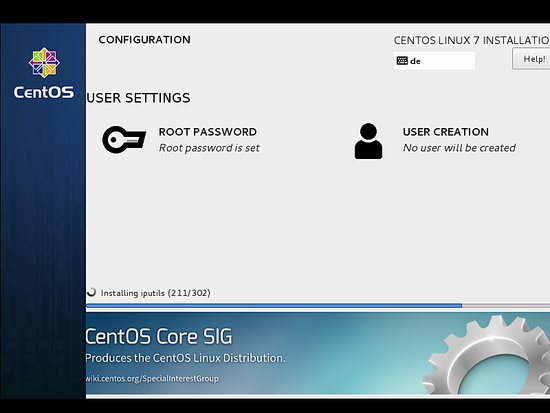
The installation process will start now and you get a small blue progress bar in the next windows. Now we have to set the *ROOT PASSWORD* and add a new non-root user in the *USER CREATION* option. I will first go for the root password.

[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-27_.png)

Enter a secure password of your choice and press *Done*

[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-28_.png)

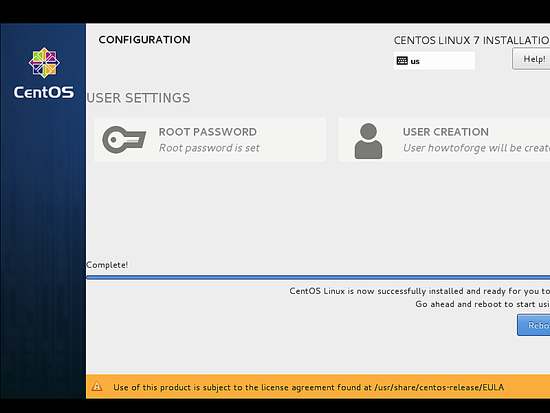
Next, we will go for *USER CREATION*.

[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-29_.png)

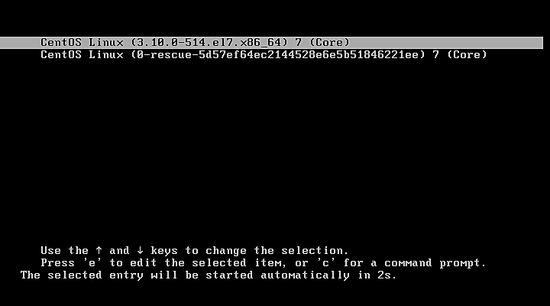
Next, I will create a non-root user, as in my case I used the *Full name”janakic" and* *Username* "janaki*"*, check the option *Require the password to use this account* and then press *Done*.

and wait for the completion of the setup.

After completion of the installation, it will ask to reboot the server, just press Finish configuration.

[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-31_1.png)

The server reboots and will request your username and password afterwards.

[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-32_.png)

Now we are ready to login with the user that we just created above or we can use the *root* credentials.

## First Login on CentOS

Login as root user to the server so we can do some final installation steps.

[](https://www.howtoforge.com/images/centos_7_3_server/big/CentOS-7.3-33_.png)

**Chapter4:**

**Configuration of yum repository:**

**Step1:** Mount the CD/DVD with any Dir. By using command as**,**

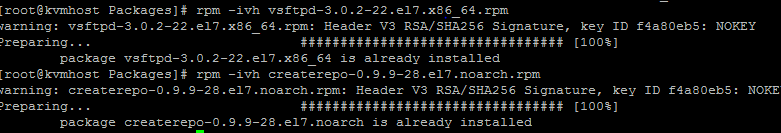
Mount /dev/sr0 /media



**Step2: Install the create repo packages and vsftpd packages along with dependences.By using RPM,**

Rpm –ivh /media/Packages/createrepo\*

Rpm –ivh /medi/Packages/vsftpa\*



**Step3**: **Create one dir as**,

Mkdir –p /var/ftp/pub/pkgdir

**Step4:** **Copy the files in side of that iso into pkgdir**

Cp /me dia/Packages/\* /var/ftp/pub/pkgdir &



**Step5:** Execute the createrepo –v /vr/ftp/pub/pkgdir

**Step6** **:Create one .repo file inside of /etc/yum.repo.d/ with config detils,**

Vi ll.repo

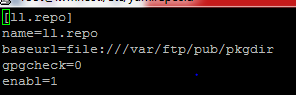
[ll.repo}

Nme=ll.reppo

Baseurl=file:///var/ftp/pub/pkgdir

gpgcheck=0

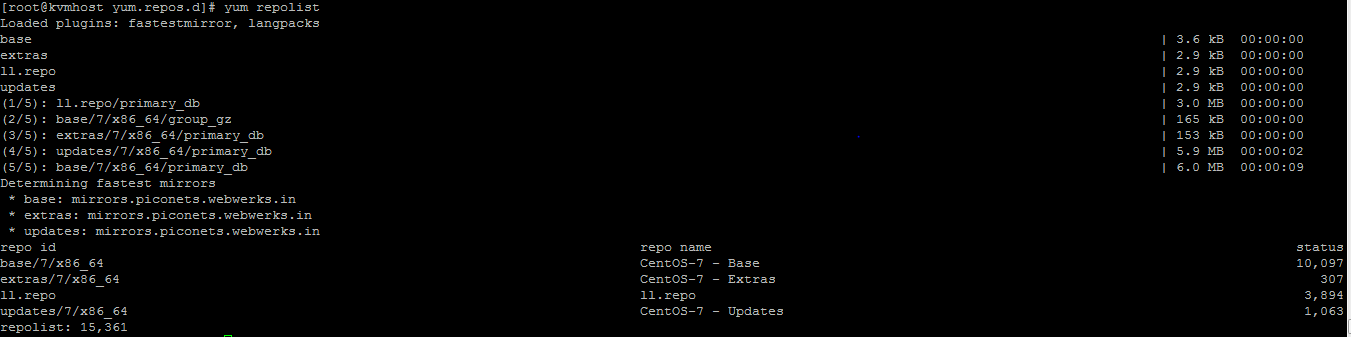
enable=1



**Step7:** **Clean all by yum command as,**

yum clean all

yum repolist (**Check the repolist**)

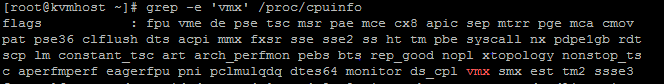


**Chapter5:**

# **Install KVM Hypervisor on top of physical server.**

KVM is an open source hardware virtualization software through which we can create and run multiple Linux based and windows based virtual machines simultaneously. KVM is known as Kernel based Virtual Machine because when we install KVM package then KVM module is loaded into the current kernel and turns our Linux machine into a hypervisor.

Before proceeding KVM installation, let’s check whether your system’s CPU supports Hardware Virtualization.



[Check Virtualization Support](https://www.tecmint.com/wp-content/uploads/2015/01/Check-Virtualization-Support.png)

*Check Virtualization Support*

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

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

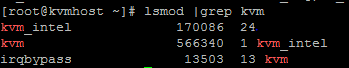
[Check CPU Virtualization Support](https://www.tecmint.com/wp-content/uploads/2015/01/Check-CPU-Virtualization-Support.png)

*Check CPU Virtualization Support*

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@kvmhost ~]# lsmod | grepkvm

The output should contains kvm\_intel for intel-based hosts or kvm\_amd for amd-based hosts.



*Check KVM Kernel Module*

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.

[root@kvmhost~]# yum update

Make sure that Selinux be in Permissive mode.

[root@kvmhost ~]# setenforce 0

### **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@kvmhost~]# yum install qemu-kvm qemu-img

**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:

a) virt-manager provides a GUI tool to administrate your virtual machines.

b)libvirt-client provides a CL tool to administrate your virtual environment this tool called virsh.

c)virt-install provides the command “virt-install” to create your virtual machines from CLI.

d)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@kvmhost ~]# yum install virt-manager libvirt libvirt-python libvirt-client

**4.** The virtualization daemon which manage all of the platform is “libvirtd”. lets restart it.

[root@kvmhost ~]#systemctl restart libvirtd

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

[root@kvmhost~]#systemctl status libvirtd

##### Sample Output

libvirtd.service - Virtualization daemon

Loaded: loaded (/usr/lib/systemd/system/libvirtd.service; enabled)

Active: active (running) since Mon 2014-12-29 15:48:46 EET; 14s ago

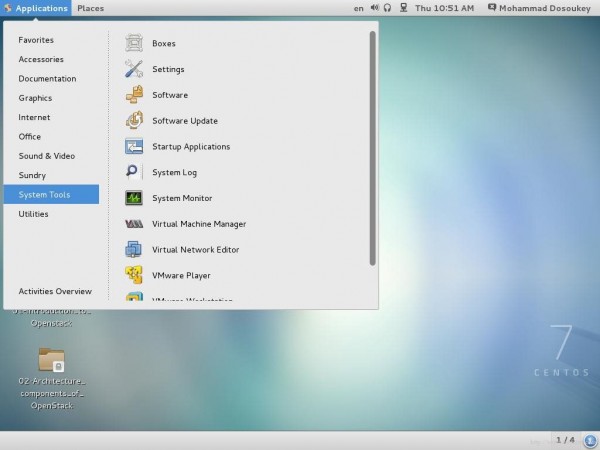
Main PID: 25701 (libvirtd)

### **Step 2: Create VMs using KVM with GUI**

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.

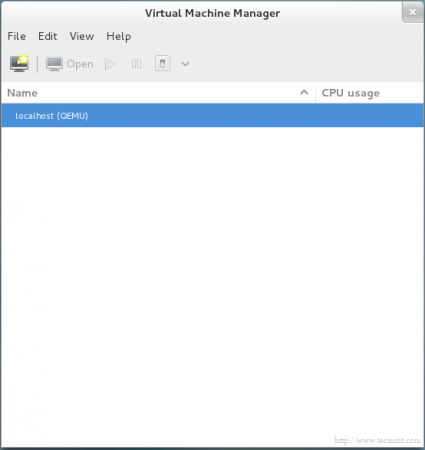
Although virt-manager is a GUI based tool, we also could launch/start it from terminal as well as from GUI.

[root@server ~]#virt-manager

[](https://www.tecmint.com/wp-content/uploads/2015/01/Start-Virtual-Manager-in-GNOME-Classic.jpeg)

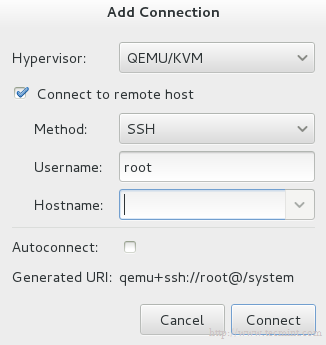
*Start Virtual Manager in GNOME Classic*

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

[](https://www.tecmint.com/wp-content/uploads/2015/01/Start-Virtual-Manager-Window.png)

*Start Virtual Manager Window*

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

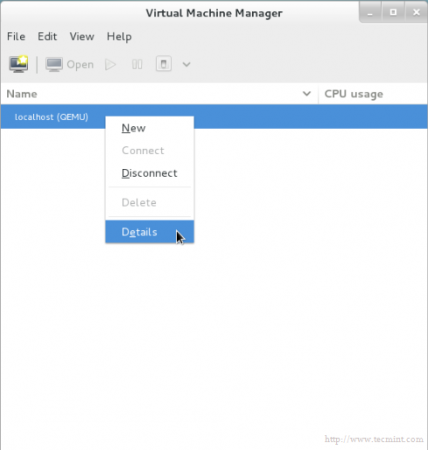
[](https://www.tecmint.com/wp-content/uploads/2015/01/Add-Connection.png)

*Add Connection*

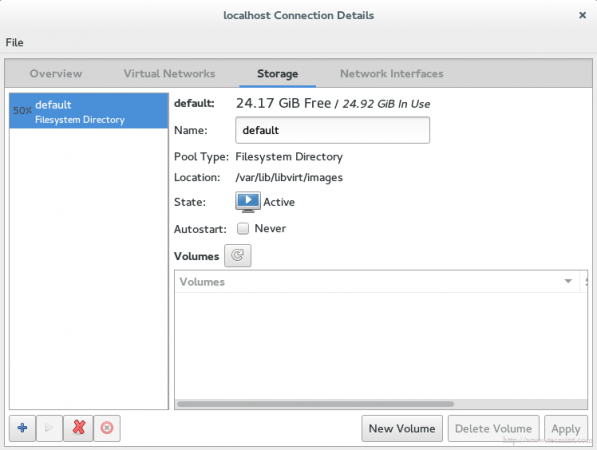
Check “Connect to remote host” option then provide Hostname/IP of the remote server. If you need establishing connection to the remote host at every time the manager starting, just check “Auto Connect” option.

**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.

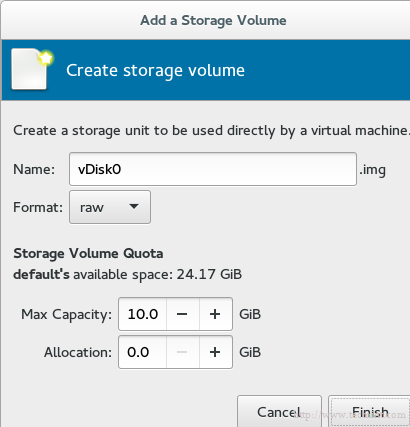
[](https://www.tecmint.com/wp-content/uploads/2015/01/VM-Storage-Details.png)

*VM Storage Details*

[](https://www.tecmint.com/wp-content/uploads/2015/01/VM-Storage-Volume.png)

*VM Storage Volume*

**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.

[](https://www.tecmint.com/wp-content/uploads/2015/01/Create-VM-Storage-Disk.png)

*Create VM Storage Disk*

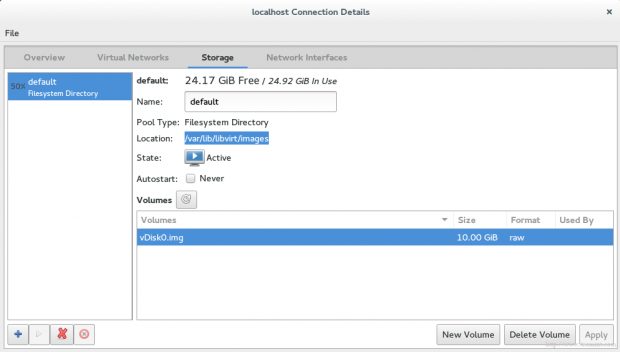
The allocation size is the actual size for your disk which will be allocated immediately from your physical disk after finishing the steps.

**Note**: This is an important technology in storage administration field which called “thin provision”. It used to allocate the used storage size only, NOT all of available size.

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.

In another words the allocated physical size will by dynamically allocated depending on the actual used size. You could find more information in details at [VMWare vStorage Thin Provisioning](https://www.vmware.com/files/pdf/VMware-vStorage-Thin-Provisioning-DS-EN.pdf).

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

[](https://www.tecmint.com/wp-content/uploads/2015/01/VM-Storage-Label.png)

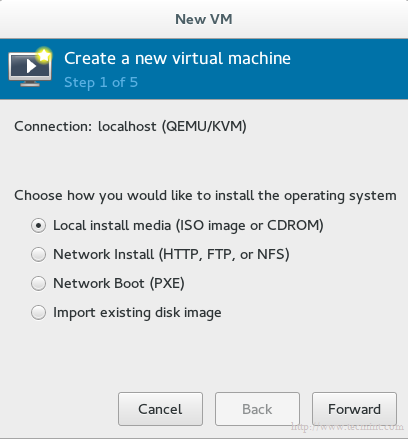
*VM Storage Label*

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.

[](https://www.tecmint.com/wp-content/uploads/2015/01/Create-New-Virtual-Machine.png)

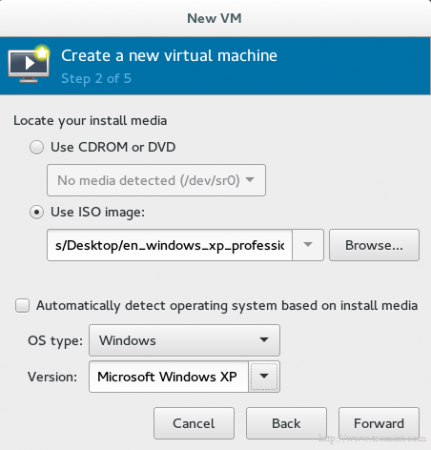
*Create New Virtual Machine*

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 its time to specify which Local install media to be used, we have two options:

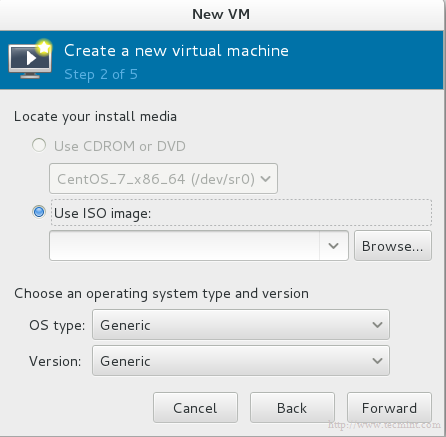
1. From physical [CDROM/DVD].
2. From ISO image.

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

[](https://www.tecmint.com/wp-content/uploads/2015/01/Select-Installation-Media.png)

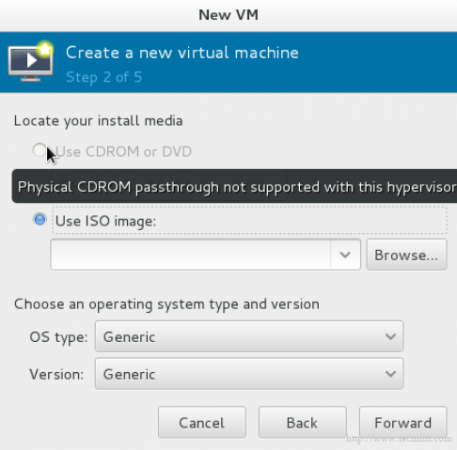
*Select Installation Media*

**Important**: Unfortunately there’s a really silly bug for whom use RHEL/CentOS7. This bug prevents you from installation using physical [CDROM/DVD], you will find the option is grayed like this.

[](https://www.tecmint.com/wp-content/uploads/2015/01/Disabled-CD-DVD-Rom.png)

*Disabled CD DVD Rom in KVM*

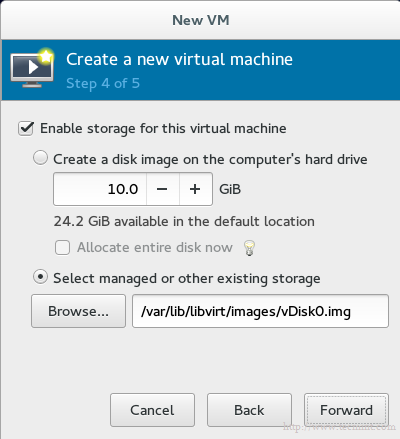
And if you hold your cursor on it, this error message will appear.

[](https://www.tecmint.com/wp-content/uploads/2015/01/CD-DVD-Not-Supported.png)

*CD DVD Not Supported in KVM*

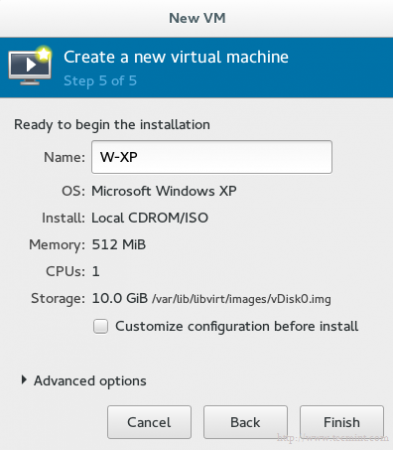
Until now there is no official/direct solution for this bug, you could find more information about this but [here](https://bugzilla.redhat.com/show_bug.cgi?id=1072610).

**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.

[](https://www.tecmint.com/wp-content/uploads/2015/01/Enable-Storage-for-Virtual-Machine.png)

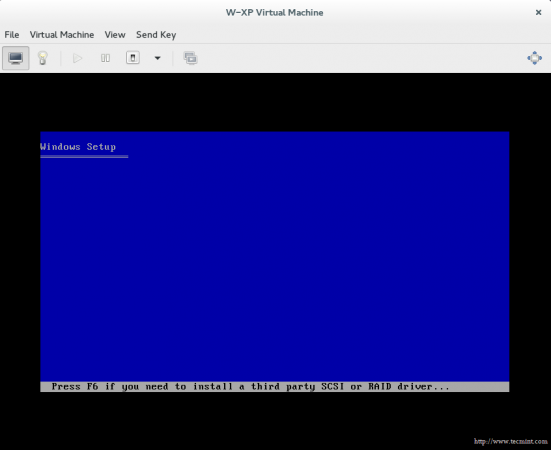
*Enable Storage for Virtual Machine*

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

[](https://www.tecmint.com/wp-content/uploads/2015/01/Enter-Name-of-Virtual-Machine.png)

*Enter Name of Virtual Machine*

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

[](https://www.tecmint.com/wp-content/uploads/2015/01/Virtual-Machine-Installation.png)

**Chapter6:**

**Configure Storage Volumes/Disk Images and vm creation by basic command .**

Check that KVM host will support for which iso by,

Osinfo-query os

Now it is disk image’s turn, using qemu-img to create new disk image from Spool1. For more details about qemy-img, use the man page.

# man qemu-img

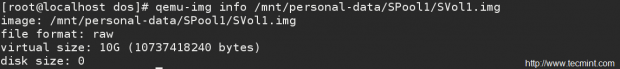
We should specify the qemu-img command “create, check,….etc”, disk image format, the path of disk image you want to create and the size.

# qemu-img create -f raw /mnt/personal-data/SPool1/SVol1.img 10G

[Create KVM Storage Volumes](https://www.tecmint.com/wp-content/uploads/2015/02/Create-Storage-Volumes.png)

*Create Storage Volume*

 By using qemu-img command info, you could get information about your new disk image.

[](https://www.tecmint.com/wp-content/uploads/2015/02/Check-Storage-Volume-Information.png)

*Check Storage Volume Information*

Warning: Never use qemu-img to modify images in use by a running virtual machine or any other process; this may destroy the image.

Now its time to create virtual machines in the next step.

Step 3: Create Virtual Machines

10. Now with the last and latest part, we will create virtual machines using virt-istall. The virt-install is a command line tool for creating new KVM virtual machines using the “libvirt” hypervisor management library. For more details about it, use:

# man virt-install

To create new KVM virtual machine, you need to use the following command with all the details like shown in the below.

1. Name: Virtual Machine’s name.
2. Disk Location: Location of disk image.
3. Graphics : How to connect to VM “Usually be SPICE”.
4. vcpu : Number of virtual CPU’s.
5. ram : Amount of allocated memory in megabytes.
6. Location : Specify the installation source path.
7. Network : Specify the virtual network “Usually be vibr00 bridge”.

**virt-install --network bridge:br0 --name lamp1server --os-variant=centos7.0 --ram=1024 --vcpus=1 --disk path=/var/lib/libvirt/images/lamp1server.qcow2,format=qcow2,bus=virtio,size=30 --graphics none --location=/opt/CentOS-7.0-1406-x86\_64-DVD.iso --extra-args="console=ttyS0,115200" --check all=off**

**Chapter7:**

**CREATING A DIRECTORY-BASED STORAGE POOL WITH VIRSH**

1. **Create the storage pool definition**

Use the virsh pool-define-as command to define a new storage pool. There are two options required for creating directory-based storage pools:

* + The name of the storage pool.

This example uses the name *guest\_images*. All further virsh commands used in this example use this name.

* + The path to a file system directory for storing guest image files. If this directory does not exist, virsh will create it.

This example uses the */guest\_images* directory.

# virsh pool-define-as *guest\_images* dir - - - - "*/guest\_images*"

Pool guest\_images defined

1. **Verify the storage pool is listed**

Verify the storage pool object is created correctly and the state reports it as inactive.

# virsh pool-list --all

Name State Autostart

-----------------------------------------

default active yes

guest\_images inactive no

1. **Create the local directory**

Use the virsh pool-build command to build the directory-based storage pool for the directory *guest\_images* (for example), as shown:

# virsh pool-build *guest\_images*

Pool guest\_images built

# ls -la /*guest\_images*

total 8

drwx------. 2 root root 4096 May 30 02:44 .

dr-xr-xr-x. 26 root root 4096 May 30 02:44 ..

# virsh pool-list --all

Name State Autostart

-----------------------------------------

default active yes

guest\_images inactive no

1. **Start the storage pool**

Use the virsh command pool-start to enable a directory storage pool, thereby allowing allowing volumes of the pool to be used as guest disk images.

# virsh pool-start *guest\_images*

Pool guest\_images started

# virsh pool-list --all

Name State Autostart

-----------------------------------------

default active yes

guest\_images active no

1. **Turn on autostart**

Turn on *autostart* for the storage pool. Autostart configures the libvirtd service to start the storage pool when the service starts.

# virsh pool-autostart*guest\_images*

Pool guest\_images marked as autostarted

# virsh pool-list --all

Name State Autostart

-----------------------------------------

default active yes

guest\_images active yes

1. **Verify the storage pool configuration**

Verify the storage pool was created correctly, the size is reported correctly, and the state is reported as running. If you want the pool to be accessible even if the guest virtual machine is not running, make sure that Persistent is reported as yes. If you want the pool to start automatically when the service starts, make sure that Autostart is reported as yes.

# virsh pool-info *guest\_images*

Name: guest\_images

UUID: 779081bf-7a82-107b-2874-a19a9c51d24c

State: running

Persistent: yes

Autostart: yes

Capacity: 49.22 GB

Allocation: 12.80 GB

Available: 36.41 GB

# ls -la /guest\_images

total 8

drwx------. 2 root root 4096 May 30 02:44 .

dr-xr-xr-x. 26 root root 4096 May 30 02:44 ..

#

By default, libvirt-based commands use the directory /var/lib/libvirt/images on a virtualization host as an initial file system directory storage pool. You can easily create a new storage pool by using the virsh pool-create-as command.

As we discussed in LAB setup –

I have a disk /dev/sdb (40G) which we will use storage pool. I will be create a LVM VG with name ‘lab\_kvm\_storage ‘ and LV with name ‘lab\_kvm\_lv’ then creating a xfs filesystem on the LV

. # vgcreate lab\_kvm\_storage /dev/sdb

# lvcreate -l +100%FREE -n lab\_kvm\_lv lab\_kvm\_storage

# mkfs.xfs /dev/mapper/lab\_kvm\_storage-lab\_kvm\_lv

Off course we will add the fstab entry for auto-mount on every system boot

. /dev/mapper/lab\_kvm\_storage-lab\_kvm\_lv /var/lib/libvirt/images xfs defaults 0 0

Mount the /var/lib/libvirt/images as a new filesystem

# mount –a Now let’s create a storage pool and start it. Also we will enable auto-start so pool will be available e on every system boot.

# virsh pool-define-as lab\_kvm\_storagepool --type dir --target /var/lib/libvirt/images

# virsh pool-autostart lab\_kvm\_storagepool

# virsh pool-start lab\_kvm\_storagepool

# virsh pool-list To see detailed information about a pool , you can run the below command.

# virsh pool-list --all –-details

#virsh pool-info lab\_kvm\_storagepool

# **virsh commands cheat sheet to manage KVM guest virtual machines**

In this virsh commands cheatsheet, I’ll show you most used virsh commands to manage Guest Virtual Machines running on KVM or Xen Hypervisor.

The basic structure of most virsh usage is:

virsh [OPTION]... <command> <domain> [ARG]...

#### Virsh display node information:

This is the first item on our virsh commands cheatsheet. This displays the host node information and the machines that support the virtualization process.

# virsh nodeinfo

CPU model: x86\_64

CPU(s): 8

CPU frequency: 2200 MHz

CPU socket(s): 1

Core(s) per socket: 4

Thread(s) per core: 2

NUMA cell(s): 1

Memory size: 12238908 KiB

#### Virsh list all domains

To list both inactive and active domains, use the command:

# virsh list --all

Id Name State

----------------------------------------------------

- admin shut off

- cloudstack shut off

- hyperv shut off

#### List only active domains

To list only active domains with virsh command, use:

# virsh list

Id Name State

----------------------------------------------------

#### Virsh start vm

This is an example on how to use virsh command to start a guest virtual machine. We’re going to start test domain displayed above

# virsh start test

Domain test started

# virsh list

Id Name State

----------------------------------------------------

3 test running

#### Virsh autostart vm

To set a vm to start automatically on system startup, do:

# virsh autostart test

Domain test marked as autostarted

# virsh dominfo test

Id: 9

Name: test

UUID: a943ed42-ba62-4270-a41d-7f81e793d754

OS Type: hvm

State: running

CPU(s): 2

CPU time: 144.6s

Max memory: 2048 KiB

Used memory: 2048 KiB

Persistent: yes

Autostart: enable

Managed save: no

Security model: none

Security DOI: 0

Keep an eye on the option Autostart: enable.

#### Virsh autostart disable

To disable autostart feature for a vm:

$ virsh autostart --disable test

Domain test unmarked as autostarted

$ virsh dominfo test

Id: -

Name: test

UUID: a943ed42-ba62-4270-a41d-7f81e793d754

OS Type: hvm

State: shut off

CPU(s): 2

Max memory: 2048 KiB

Used memory: 2048 KiB

Persistent: yes

Autostart: disable

Managed save: no

Security model: none

Security DOI: 0

#### Virsh stop vm, virsh shutdown vm

To shutdown a running vm gracefully use:

$ virsh shutdown test

Domain test is being shutdown

$ virsh list

Id Name State

----------------------------------------------------

#### Virsh force shutdown vm

You can do a forceful shutdown of active domain using the command:

$ virsh destroy test

#### Virsh stop all running vms

In case you would like to shutdown all running domains, just issue the command below:

$ for i in `sudo virsh list | grep running | awk '{print $2}'` do

virsh shutdown $i

done

#### Virsh reboot vm

To restart a vm named test, the command used is:

# virsh reboot test

**Chapter 8:**

**How to Add Memory, vCPU, Hard Disk to Linux KVM Virtual Machine**

This will explain how to perform the following Linux KVM VM activities:

1. Add Memory to VM
2. Add vCPU to VM
3. Add Disk to VM
4. Save VM Configuration
5. Delete a VM

### **1. Add Memory to Virtual Machine**

To add additional memory to your VM, you should do the following:

* Shutdown your VM
* Edit the VM file and increase the value of maximum memory allocated to this VM
* Restart the VM
* Use virsh setmem to set the memory upto the maximum memory allocated for this VM.

In this example, let us increase the memory of myRHELVM1’s VM from 2GB to 4GB.

First, shutdown the VM using virsh shutdown as shown below:

# virsh shutdown myRHELVM1

Domain myRHELVM1 is being shutdown

Next, edit the VM using virsh edit:

# virsh edit myRHELVM1

Look for the below line and change the value for memory to the following. In my example, earlier it was 2097152:

<memory unit='KiB'>4194304</memory>

Please note that the above value is in KB. After making the change, save and exit:

# virsh edit myRHELVM1

Domain myRHELVM1 XML configuration edited.

Restart the VM with the updated configuration file. Now you will see the max memory increased from 2G to 4G.

You can now dynamically modify the VM memory upto the 4G max limit.

Create the Domain XML file using virsh create

# virsh create /etc/libvirt/qemu/myRHELVM1.xml

Domain myRHELVM1 created from /etc/libvirt/qemu/myRHELVM1.xml

View the available Memory for this domain. As you see below, even though the maximum available memory is 4GB, this domain only has 2GB (Used memory).

# virsh dominfo myRHELVM1 | grep memory

Max memory: 4194304 KiB

Used memory: 2097152 KiB

Set the memory for this domain to 4GB using virsh setmem as shown below:

# virsh setmem myRHELVM1 4194304

Now, the following indicates that we’ve allocated 4GB (Used memory) to this domain.

# virsh dominfo myRHELVM1 | grep memory

Max memory: 4194304 KiB

Used memory: 4194304 KiB

### 2. **Add VCPU to VM**

To increase the virtual CPU that is allocated to the VM, do virsh edit, and change the vcpu parameter as explained below.

In this example, let us increase the memory of myRHELVM1’s VM from 2GB to 4GB.

First, shutdown the VM using virsh shutdown as shown below:

# virsh shutdown myRHELVM1

Domain myRHELVM1 is being shutdown

Next, edit the VM using virsh edit:

# virsh edit myRHELVM1

Look for the below line and change the value for vcpu to the following. In my example, earlier it was 2.

<vcpu placement='static'>4</vcpu>

Create the Domain XML file using virsh create

# virsh create /etc/libvirt/qemu/myRHELVM1.xml

Domain myRHELVM1 created from /etc/libvirt/qemu/myRHELVM1.xml

View the virtual CPUs allocated to this domain as shown below. This indicates that we’ve increased the vCPU from 2 to 4.

# virsh dominfo myRHELVM1 | grep -i cpu

CPU(s): 4

CPU time: 21.0s

### 3. Add Disk to VM

In this example, we have only two virtual disks (vda1 and vda2) on this VM.

# fdisk -l | grep vd

Disk /dev/vda: 10.7 GB, 10737418240 bytes

/dev/vda1 \* 3 1018 512000 83 Linux

/dev/vda2 1018 20806 9972736 8e Linux LVM

There are two steps involved in creating and attaching a new storage device to Linux KVM guest VM:

* First, create a virtual disk image
* Attach the virtual disk image to the VM

Let us create one more virtual disk and attach it to our VM. For this, we first need to create a disk image file using qemu-img create command as shown below.

In the following example, we are creating a virtual disk image with 7GB of size. The disk images are typically located under /var/lib/libvirt/images/ directory.

# cd /var/lib/libvirt/images/

# qemu-img create -f raw myRHELVM1-disk2.img 7G

Formatting 'myRHELVM1-disk2.img', fmt=raw size=7516192768

To attach the newly created disk image, use the virsh attach-disk command as shown below:

# virsh attach-disk myRHELVM1 --source /var/lib/libvirt/images/myRHELVM1-disk2.img --target vdb --persistent

Disk attached successfully

The above virsh attach-disk command has the following parameters:

* myRHELVM1 The name of the VM
* –source The full path of the source disk image. This is the one that we created using qemu-image command above. i.e: myRHELVM1-disk2.img
* –target This is the device mount point. In this example, we want to attach the given disk image as /dev/vdb. Please note that we don’t really need to specify /dev. It is enough if you just specify vdb.
* –persistent indicates that the disk that attached to the VM will be persistent.

As you see below, the new /dev/vdb is now available on the VM.

# fdisk -l | grep vd

Disk /dev/vda: 10.7 GB, 10737418240 bytes

/dev/vda1 \* 3 1018 512000 83 Linux

/dev/vda2 1018 20806 9972736 8e Linux LVM

Disk /dev/vdb: 7516 MB, 7516192768 bytes

Now, you can partition the /dev/vdb device, and create multiple partitions /dev/vdb1, /dev/vdb2, etc, and mount it to the VM. Use [fdisk to create the partitions](https://www.thegeekstuff.com/2010/09/linux-fdisk/) as we explained earlier.

Similarly to detach a disk from the guest VM, you can use the below command. But be careful to specify the correct vd\* otherwise you may end-up removing wrong device.

# virsh detach-disk myRHELVM1 vdb

Disk detached successfully

### **4. Save Virtual Machine Configuration**

If you make lot of changes to your VM, it is recommended that you save the configurations.

Use the virsh dumpxml file to take a backup and save the configuration information of your VM as shown below.

# virsh dumpxml myRHELVM1 > myrhelvm1.xml

# ls myrhelvm1.xml

myrhelvm1.xml

Once you have the configuration file in the XML format, you can always recreate your guest VM from this XML file, using virsh create command as shown below:

virsh create myrhelvm1.xml

### **5. Delete KVM Virtual Machine**

If you’ve created multiple VMs for testing purpose, and like to delete them, you should do the following three steps:

* Shutdown the VM
* Destroy the VM (and undefine it)
* Remove the Disk Image File

In this example, let us delete myRHELVM2 VM. First, shutdown this VM:

# virsh shutdown myRHELVM2

Domain myRHELVM2 is being shutdown

Next, destory this VM as shown below:

# virsh destroy myRHELVM2

Domain myRHELVM2 destroyed

Apart from destroying it, you should also undefine the VM as shown below:

# virsh undefine myRHELVM2

Domain myRHELVM2 has been undefined

Finally, remove any disk image file that you’ve created for this VM from the /var/lib/libvirt/images directory:  
Now you can remove the disk img file under /var/lib/libvirt/images

rm /var/lib/libvirt/images/myRHELVM2-disk1.img

rm /var/lib/libvirt/images/myRHELVM2-disk2.img

**chapter 10:**

**Snapshot management for vm in kvm**

Note : VM needs to upon running.

List the current snapshots

# virsh snapshot-list testvm1

Create a Snapshot

# virsh snapshot-create-as --domain testvm1 --name "snapshot\_name" --description “description"

# virsh snapshot-list testvm1 -----🡪To check the details of a snapshot

# virsh snapshot-info --domain testvm1 --current To revert to a snapshot ( snapshot restore)

# virsh shutdown --domain testvm1

To revert the snap:

# virsh snapshot-revert --domain testvm1 --snapshotname "snapshot\_name" --running

To delete a snapshot:

# virsh snapshot-delete --domain guest\_vm --snapshotname "snapshot\_name”

**How to clone existing KVM virtual machine images on Linux**

Note: VM should be in shutdown state.  
# virt-clone --original {Domain-Vm-Name-Here} --auto-clone  
OR  
# virt-clone --original {Domain-Vm-Name-Here} --name {New-Domain-Vm-Name-Here} --auto-clone  
OR  
# virt-clone --original {Domain-Vm-Name-Here} \  
--name {New-Domain-Vm-Name-Here} --file {/var/lib/libvirt/images/File.Name.here}

