HARVARD
Extension School

Week 2
Understanding and setting up the working environment

Contents

In this lecture

Linux environment: In this lecture students will learn how to work in a Linux environment, install modules and packages, use linux commands, and write scripts using bash. By the end of this lecture, students will learn how to install the Apache web-server on ubuntu linux.

- Linux Commands/basics
  - Packages managers (apt, yum/dnf)
  - File system commands
  - System status and network status
- Bash scripting
- Authentication and Connection to a remote environment: console, ssh, rdp
- Simple Apache web server setup

What is linux

- Open source Unix like Operating System released by Linus Torvalds 1991
- An operating system (OS) is the software one uses for all interactions with the computer’s hardware. Other examples of OSs are Windows and Mac OSX (which itself is mostly unix-like).
- Linux mainly consist of a Kernel and some System programs
- Most of the servers now uses Linux, some estimate 40% of the web servers.One third of Azure instances are Linux! Self-driving car and cell phones are all linux.

Why Linux

- Open Source so it is free
- Small in size (Try to install windows on old computer and lets see!)
- Community support (Do you hire a windows consultant?)
- Reliable (How many times did you reboot your windows?)

Who uses Linux

- Supercomputers -- almost exclusively run Linux
- Linux servers form the backbone of internet
- Cloud platforms (AWS, Google cloud, even Azure) are mostly based on linux for building linux servers, which are much cheaper than their windows counterparts (bonus, azure's network is linux managed https://www.wired.com/2015/09/microsoft-using-linux-run-cloud/)
- Linux - essential for data science, machine learning, and deep learning
- Home projects (including IoT)
- MacOS command line (CLI) runs a unix variant: BSDe. (see also: https://www.lifewire.com/mac-os-x-is-not-linux-distribution-2204744)

Installing and Booting Linux

1. Pick the linux flavor:
    - CentOS
    - Fedora
    - Ubuntu
    - OpenSUSE
    - Debian
2. Machine (Physical or VM) + ISO
3. We will use VirtualBox (open source virtualization product) to create Virtual machines to install linux.

VirtualBox

https://www.virtualbox.org

Install a Virtual machine on Virtualbox

- Download the OS iso, for example ubuntu desktop:

  https://www.ubuntu.com/download/desktop

- Installation Demo:

Boot Sequence

- Basic Input/Output System - BIOS
  - BIOS Loads and execute the Master Boot Record - MBR.
- Master Boot Record - MBR
  - About 512 bytes located at the first sector of the bootable disk(/dev/hda, /dev/sda).
  - MBR loads and executes the the Grand Unified Bootloader - GRUB.
- Grand Unified Bootloader - GRUB2
  - To choose form multiple kernel images installed on the system.
  - GRUP located on /boot/grub/grub.conf.
  - GRUB loads and executes Kernel and initrd images (that contains executable like the SHELL).

Boot Sequence (Cont.)

- Kernel
  - kernels are located in the /boot directory, along with an initial RAM disk image (initramfs), and device maps of the hard drives.
  - Once Kernel extract it self it load and execute the systemd (old SysV init).‘ps -ef | grep systemd.
  - At the end of the boot process, initramfs is unmounted and the real file system is mounted and init run on the file system.
- Systemd
  - At first, systemd mounts the filesystems as defined by /etc/fstab including swap file (what is that?)
  - /etc/systemd/system/default.target -> graphical.target (runlevel 5 in the old SystemV init).
  - For a server, the default is more likely to be the multi-user.target which is like runlevel 3 in SystemV.
  - The emergency.target is similar to single user mode.

Booting Process

https://medium.com/@Seavievv/booting-process-in-centos7-e1f4a817d32b

BIOS
- Perform **POST**
- Loads **MBR**

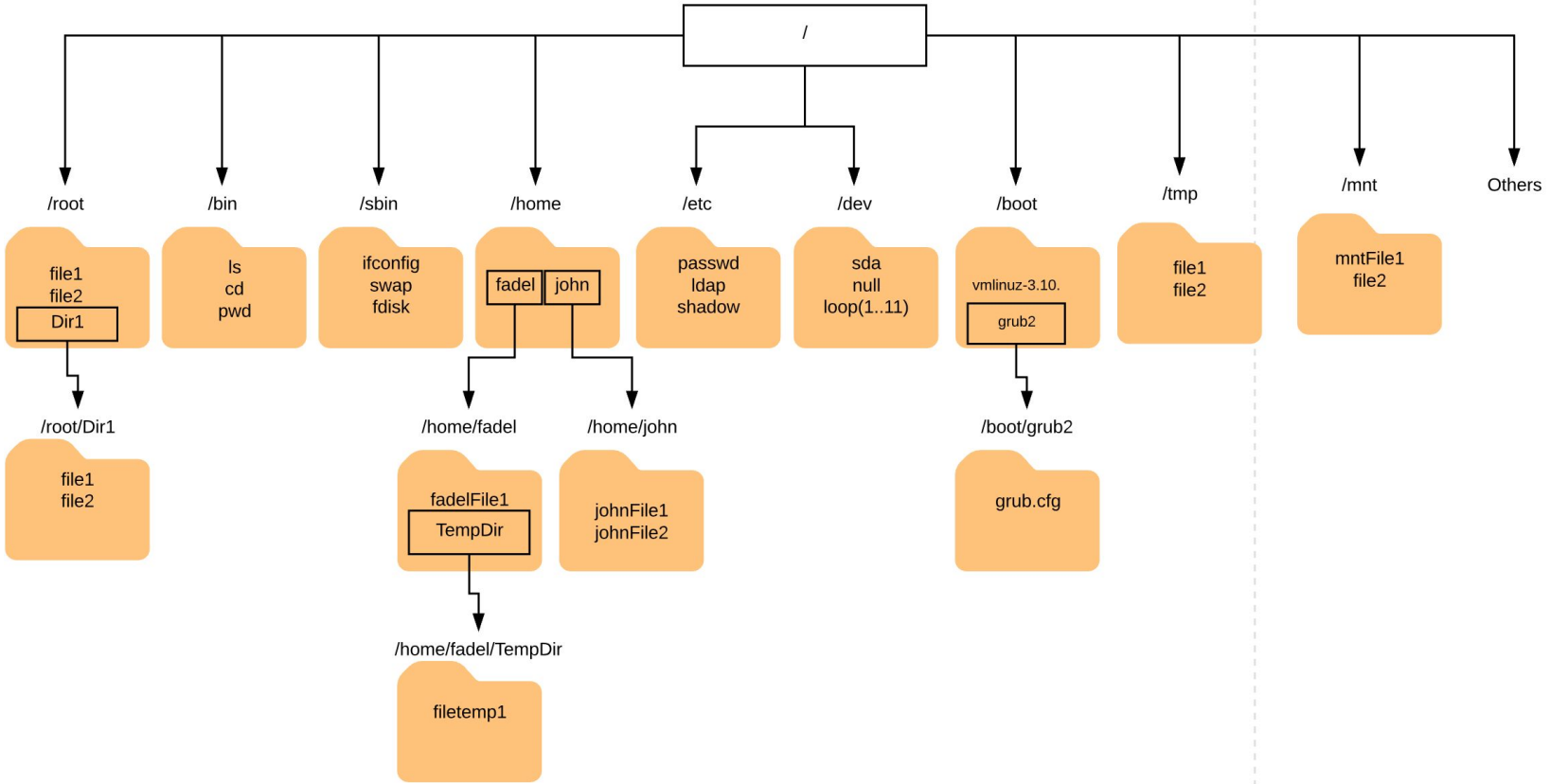
MBR
- Loads **GRUB2** Boot Loader

GRUB2
- Loads the **vmlinuz** kernel image
- Extracts the contents of **initramfs** image

KERNEL
- Loads necessary driver modules from initrd image
- Starts systems 1st process - **systemd**

SYSTEMD
- Reads configuration files from the **/etc/systemd** direcotry
- Reads file linked by **/etc/systemd/system/default.target**
- Brings the system to the state defined by the system target

File System


/
/root
/bin
/sbin
/home
/etc
/dev
/boot
/tmp
/mnt
Others
file1
file2
Dir1
ls
cd
pwd
ifconfig
swap
fdisk
fadel
john
passwd
ldap
shadow
sda
null
loop(1..11)
vmlinuz-3.10.
grub2
file1
file2
mntFile1
file2
/root/Dir1
file1
file2
/home/fadel
/home/john
fadelFile1
TempDir
johnFile1
johnFile2
/home/fadel/TempDir
filetemp1
/boot/grub2
grub.cfg

/ : All files and folders start from here
/root : The root user home directory.
/boot – Boot Loader Files: Boot loader related files. grub files, vmlinuz, initramfs are located under /boot
/bin – User Binaries : Contains binary executable commands used by users like pwd, ls, ps, etc
/sbin – System Binaries : contains binary executable commands used by system administrators for maintenance purpose like fdisk, ifconfig , swap
/etc – Configuration Files : Contains configuration files required by the system and the software stack. For example: /etc/resolv.conf, /etc/logrotate.conf , /etc/httpd/, /etc/apache2/ , /etc/puppet.
/dev – Device Files: Device files that include terminal devices, usb, or any device attached to the system. For example: /dev/tty1, /dev/sda
/proc – Process Information: It is a virtual file system contains information about running processes and the resources on the system.For example: /proc/uptime , /proc/{pid} directory contains information about the process with that particular pid.

/var – Variable Files : files that are expected to grow like log file , databases , packages , mail , temporary files . /var/log/, /var/mail/, /var/lib/ , /var/tmp/.
/tmp – Temporary Files : Temporary files created by system and users that might be deleted when system is rebooted.
/usr – User Programs : Contains binaries (users : /usr/bin like scp, admins: /usr/sbin like useradd), libraries /usr/lib, /usr/lib64) for the binaries in /usr/bin and /usr/sbin. /usr/local contains users programs installed from source. For example, apache installation from source goes under /usr/local/apache2
/home – Home Directories: Store the home directories for users. For example: /home/fadel, /home/peter
/lib – System Libraries: Library files for the binaries under /bin and /sbin. Library filenames usually ld* or lib*.so.* . For example: ld-2.2.1.so, libncurses.so.6.0
/opt – Optional add-on Applications : add-on applications from individual vendors should be installed under either /opt/ or /opt/ sub-directory.
/mnt – Mount Directory : Temporary mount directory.
/media – Removable Media Devices: Temporary mount directory for removable devices. For examples, /media/cdrom for CD-ROM; /media/cdrecorder for CD writer
/srv – Service Data: specific services related data. For example, /srv/mysql contains mysql related data.

ISO and VM Image

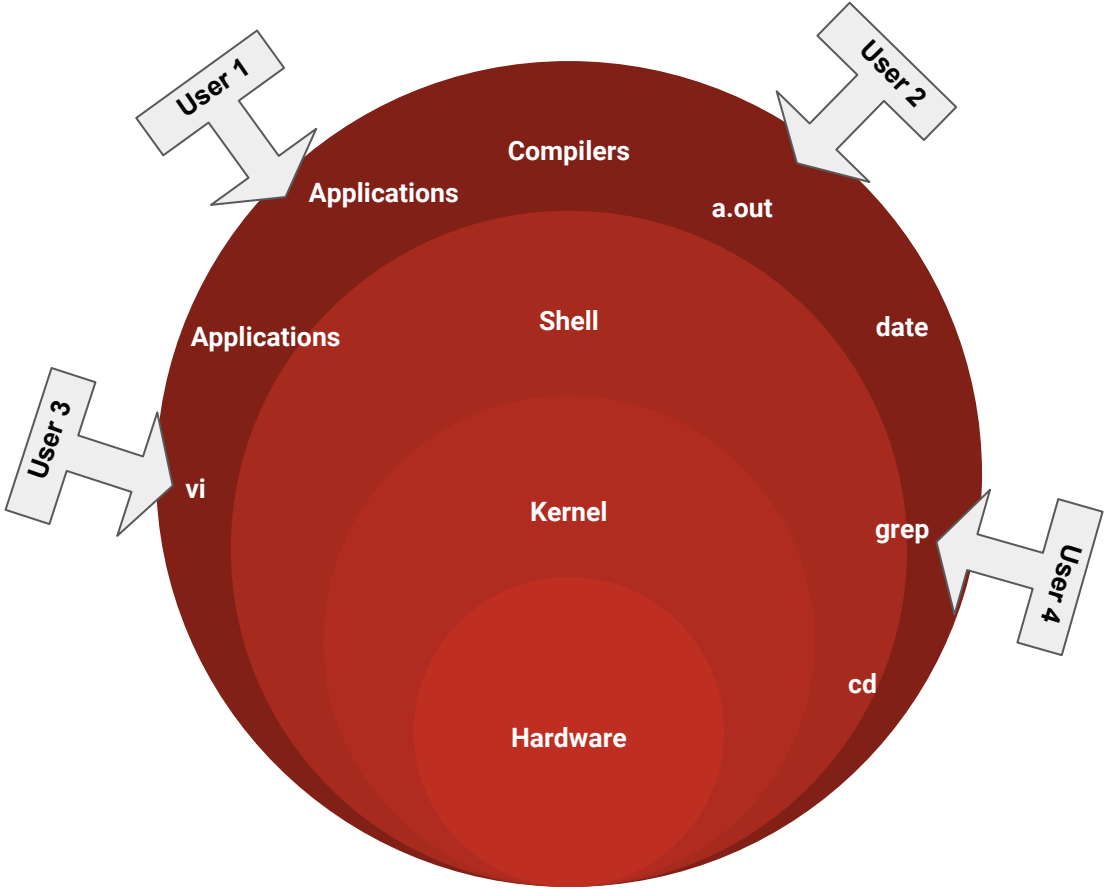
- So What is the difference between the image of the VM and the ISO we create the VM from ?

ISO
ISO is a single file that have the parts required to install an OS on the VM disk

VM
VM image is a fully installed OS and Software Stack on a filesystem.

- Why it is important to differentiate between the iso and the image ?
- Can we migrate the Image or the ISO to AWS?

Linux Shell


User 1
User 2
User 3
User 4
Compilers
Applications
a.out
Shell
Applications
date
vi
Kernel
grep
cd
Hardware

Linux CLI on the shell

```
List files ..
>ls
Listing, longer form with more details..
>ls -l
Finding more info on commands via “man”
> ls -l /
lrwxrwxrwx.   1 root root    7 May 14 12:14 bin -> usr/bin
dr-xr-xr-x.   5 root root 4096 Sep  8 13:16 boot
drwxr-xr-x.  22 root root 3520 Aug 20 09:49 dev
drwxr-xr-x. 145 root root 8192 Sep  8 13:11 etc
drwxr-xr-x.   3 root root   19 Apr 11 00:59 home
.
```

```
Which user am I using?:
$> whoami
What shell are we in?:
$> echo $SHELL
Where am I? (see directory tree below)
$> pwd
Go to a particular directory (relative path vs full path)
$> cd <directory_name>
Change to directory one level up..
$> cd ..
Change back to home directory ..
$> cd
Print something
$> echo “Hello world”
Hello world
List running process
$> ps
$> ps -ef
Print file content
$> cat file1.txt
Edit file
$> vi file1
```

```
What is the path of a certain command
$> which ls
What is the PATH variable
$> echo $PATH
/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/home/fadel/.local/bin:/home/fadel/bin
Where is those variable PATH and SHELL
$> env
How can I create directory ?
$> mkdir newDir
How can I create file
$> touch fileName
How can I add data (like Hello World) to a file with name fileName
$> echo “Hello World” >>  fileName
$> cat fileName
How I can remove a file or a directory
$> rm fileName
$> rm -r Dir
Hello world
Can I be root?
$> su -
$> sudo su -
```

```
What is my ethernet and IP config
$> ifconfig
$> /sbin/ifconfig
What is my routing table
$> route
What is my disk space
$> df
Can I mount a disk ?
$> mount /dev/sdb
Can I just mount ? mount what
No
Attach a Dist
Create file system on the disk
Then mount the disk
$> mkfs.ext4 /dev/sdb
What about when I reboot the system , is the mount still there ?
No.
What I should do ?
$> echo “/dev/sdb /mnt ext4 0 0” >> /etc/fstab
What is the files and folder size ?
$> du
$> du -sh /tmp
```

More Linux Commands

```
grep (egrep), cut
$> grep firas FileName.txt
$> cut -d ‘,’ -f 1 FileName.txt

pipe (|)
$> ls /tmp | grep fadel
$> cat FileName.txt | grep “firas”

redirection (>,>>,<)
$> ls /tmp > /tmp/lsTmp.txt
$> cat FileName.txt >> CollectFilesContent.txt

Wildcards
$> ls /tmp/*.iso

variables (explain $)
$> var=”Firas”
$> echo $var
```

Bash Scripting

```
$> vim myDirFiles.sh
#!/bin/bash
#A primitive script to make directories and copy files
mkdir test && cd test
touch test0.txt test1.txt test2.txt test3.txt test4.txt
mkdir dir1 dir2 dir3 dir4
mv test1.txt dir1
mv test2.txt dir2
mv test3.txt dir3
mv test4.txt dir4
echo “This is a directory with some folders and files” > README
echo “the list is ” >> README
ls >> README
tree >> README
echo "All Done!"
CTRL+C + qw!
$> bash myDirFiles.sh
```

```
$> cat > myDirFiles.sh << EOF
#!/bin/bash
mkdir test && cd test
touch test0.txt
for i in {1..4}
do
touch test$i.txt
mkdir dir$i
mv test$i.txt dir$i
done
echo “This is a directory with some folders and files” > README
echo “the list is ” >> README
ls >> README
tree >> README
echo "All Done!"
EOF
$> bash myDirFiles.sh
$> . myDirFiles.sh
$> source myDirFiles.sh
```

OS specific Topics

1. File permissions
2. processes (top, ps, ps -ef, kill (signals), nohup, nice, etc.)
3. Memory (free, swap)
4. Archive and compression: tar, gzip
5. File type and file copy: file, dd
6. File system types: xfs, ext3, ext4, fat, fstab
7. Export command (bash), setenv (csh, tcsh)
8. Cron jobs
9. Users (useradd, passwd, sudo)
10. Small kernel, modules to add and remove during run time (ala drivers)

Authentication and Connection

Linux/Unix Console

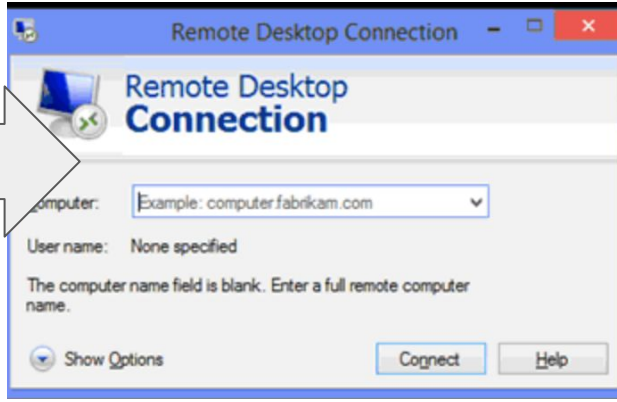
SSH

```
$> ssh-keygen
$> ssh-add
$> ssh username@server
$> ssh username@server -A
```

RDP(Windows)

Starts Remote Desktop Connection (RDP)

```
mstsc /v:server /admin
```



Package management

- CentOS
- Fedora
- Ubuntu

CentOS

Source Code:

- Compile and run the installation manually
- Dependance ?
  - Compiler for the source
  - Package dependency for the installed package

RPM: RedHat package manager

- Source archive(s) (e.g. .tar.gz, .tar.bz2) are included in SRPMs.
- Automatic build-time dependency evaluation.

Yum: Yellowdog updater, modified

- Automatic updates, package and dependency management, on RPM-based distributions

CentOS, Install apache from the source code

```
$>  curl -O  http://www-us.apache.org/dist//httpd/httpd-2.4.34.tar.gz
$>  tar xzf httpd-2.4.34.tar.gz
$>  cd httpd-2.4.34
$>  ./configure --prefix=/opt/
$>  make && $ make install
$>  vim /opt/conf/httpd.conf
$>  /opt/bin/httpd  start
```

CentOS, Install apache from the RPM

```
$>  wget https://rpmfind.net/linux/centos/7.5.1804/os/x86_64/Packages/httpd-2.4.6-80.el7.centos.x86_64.rpm
$> rpm -U httpd-2.4.6-80.el7.centos.x86_64.rpm

#Create RPM from source
$> wget http://mirrors.ibiblio.org/apache//httpd/httpd-2.4.34.tar.bz2
$> yum install rpmbuild zlib-devel libselinux-devel libuuid-devel apr-devel apr-util-devel pcre-devel
openldap-devel lua-devel libxml2-devel openssl-devel
$> rpmbuild -tb httpd-2.4.34.tar.bz2
$> echo “I am waiting for long compilation process : and there was an error”
```

CentOS, Install apache using YUM

```
Yum repositories path /etc/yum.repos.d/

Client Config:
/etc/yum.repos.d/CentOS-Base.repo

$> cat /etc/yum.repos.d/CentOS-Base.repo
[base]
name=CentOS-$releasever - Base
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=os&infra=$infra
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
$> yum install httpd
```

Fedora and Ubuntu

Installing from the source
Same as CentOS.

Fedora:

```
$> dnf install httpd
$> yum install httpd
```

Ubuntu:

```
$> apt-get install apache2
```

Apache Configuration

CentOS:

/etc/httpd

Ubuntu:

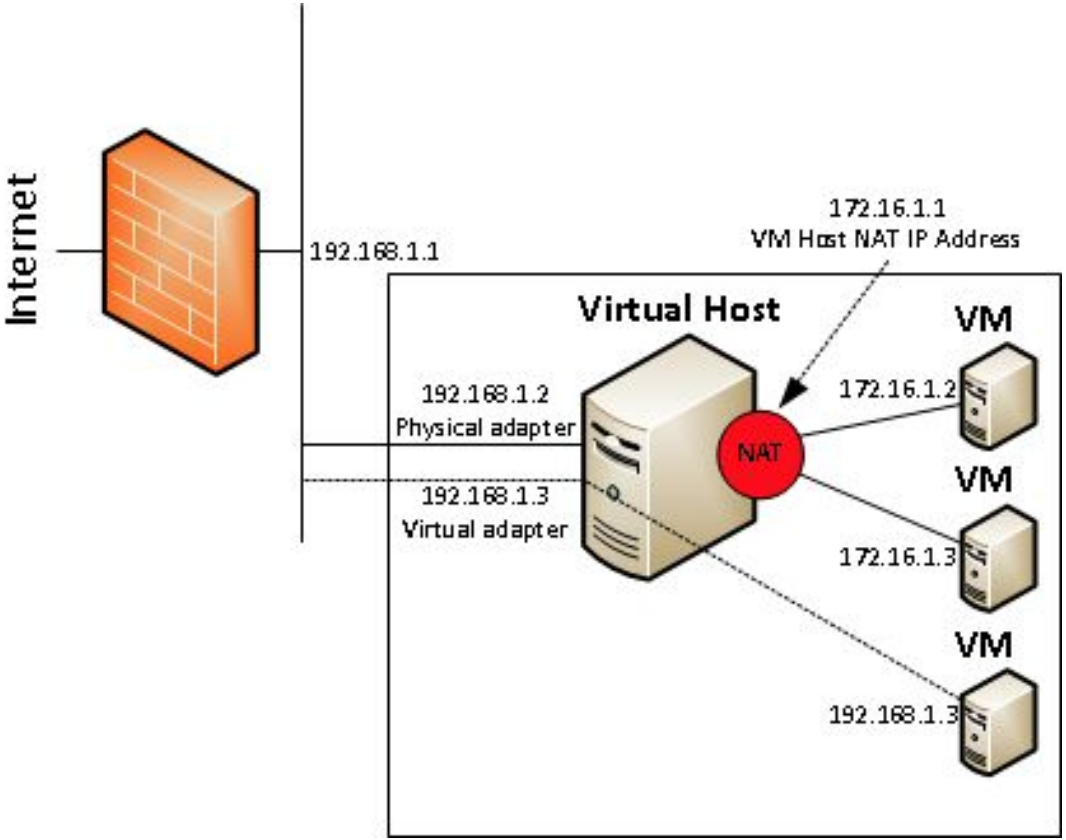
/etc/apache2

Fedora:

/etc/httpd

VirtualBox Networking and CLI

VirtualBox Networking



VirtualBox CLI

```
curl -O  http://mirrors.seas.harvard.edu/centos/7/isos/x86_64/CentOS-7-x86_64-Minimal-1804.iso

VM=e91
VBoxManage createhd --filename $VM.vdi --size 32768
   #Medium created. UUID: b78361f6-d07e-40a5-83e5-ce5e44c739f4

# List hdds
VBoxManage list hdds

# To delete hdds
   #UUID=b78361f6-d07e-40a5-83e5-ce5e44c739f4
   #vboxmanage closemedium disk $UUID --delete

# List OSs
VBoxManage list ostypes

# Register a vm
VBoxManage createvm --name $VM --ostype "RedHat_64" --register
#Virtual machine 'e91' is created and registered.
#UUID: 2b669e45-580b-4e13-b496-eef17a7700a2


ls -l  /Users/faras/VirtualBox\ VMs/e91/
   #e91.vbox
cat /Users/faras/VirtualBox\ VMs/e91/e91.vbox
```

```
# If Bridge networking is used
ifconfig bridge0 create
ifconfig bridge0 up addm en0 addm en1

#NAT networking
VBoxManage natnetwork add --netname natnet1 --network "192.168.15.0/24" --enable --dhcp on
VBoxManage list natnetworks

# Lets look at e91 vm setting from the VirtualBox GUI
# GUI


# Add controller
VBoxManage storagectl $VM --name "SATA Controller" --add sata --controller IntelAHCI

# Attach the desk created earlier to the controller
VBoxManage storageattach $VM --storagectl "SATA Controller" --port 0 --device 0 --type hdd --medium $VM.vdi


#Add an IDE controller with a DVD drive attached, and the install ISO inserted into the drive
VBoxManage storagectl $VM --name "IDE Controller" --add ide
VBoxManage storageattach $VM --storagectl "IDE Controller" --port 0 --device 0 --type dvddrive --medium
CentOS-7-x86_64-Minimal-1804.iso


# Check the storage again from the GUI, new device with iso is attached
#
```

```
# I/O APIC The Intel I/O Advanced Programmable Interrupt Controller
VBoxManage modifyvm $VM --ioapic on

# First boot is the iso dvd
VBoxManage modifyvm $VM --boot1 dvd --boot2 disk --boot3 none --boot4 none

# setup memory
VBoxManage modifyvm $VM --memory 1024 --vram 128

# network
VBoxManage modifyvm $VM --nic1 bridged --bridgeadapter1 bridge0

# Show the machine info
VBoxManage showvminfo $VM

# Start the machine
VBoxHeadless --startvm $VM
   #Oracle VM VirtualBox Headless Interface 5.2.16
   #(C) 2008-2018 Oracle Corporation
   #All rights reserved.

# Go to the console of the VM on the GUI
# Install CentOS

# If Everything went fine, remove the installation iso
VBoxManage storageattach $VM --storagectl "IDE Controller" --port 0 --device 0 --type dvddrive --medium none
VBoxManage storagectl $VM --name "IDE Controller" --remove
# Reorder the boot sequence
VBoxManage modifyvm $VM --boot1 disk --boot2 none --boot3 none --boot4 none
```

```
#vboxmanage or VBoxManage

# Snapshot and restore

#Snapshot
VBoxManage snapshot $VM take $VM.snap1

# List snapshots of the virtual machine
vboxmanage  snapshot $VM list
   #   Name: e91.snap1 (UUID: 560f6b82-541d-4a40-91a6-c304d302a18f) *


#Restore
VBoxManage snapshot $VM restore $VM.snap1
   #Restoring snapshot 'e91.snap1' (560f6b82-541d-4a40-91a6-c304d302a18f)
   #0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%


# Delete a snapshot
vboxmanage  snapshot $VM delete $VM.snap1
```