

# **WELCOME TO: MODULE 7**

## **NETWORKING, SERVICES AND SYSTEM UPDATES**

# Internet Access to VM

- Open **Virtualbox Manager**
- Select the machine you cannot get internet on in the left pane
- Click the **Settings** button in the top menu
- Click **Network** in the left pane in the settings window
- Switched to **Bridged Adaptor** in the **Attached to** drop-down menu
- Hit **OK** to save your changes
- Start your VM

# Network Components

- IP
  - Subnet mask
  - Gateway
  - Static vs. DHCP
- 
- Interface
  - Interface MAC.

# Network Files and Commands

- Interface Detection
- Assigning an IP address
- Interface configuration files
  - /etc/nsswitch.conf
  - /etc/hostname
  - /etc/sysconfig/network
  - /etc/sysconfig/network-scripts/ifcfg-nic
  - /etc/resolv.conf
- Network Commands
  - **ping**
  - **ifconfig**
  - **ifup** or **ifdown**
  - **netstat**
  - **tcpdump**

# NIC Information

NIC = Network Interface Card

**Example:**

```
ethtool enp0s3
```



Other NICs

**lo** = The loopback device is a special interface that your computer uses to communicate with itself. It is used mainly for diagnostics and troubleshooting, and to connect to servers running on the local machine

**virb0** = The virbr0, or "Virtual Bridge 0" interface is used for NAT (Network Address Translation). Virtual environments sometimes use it to connect to the outside network

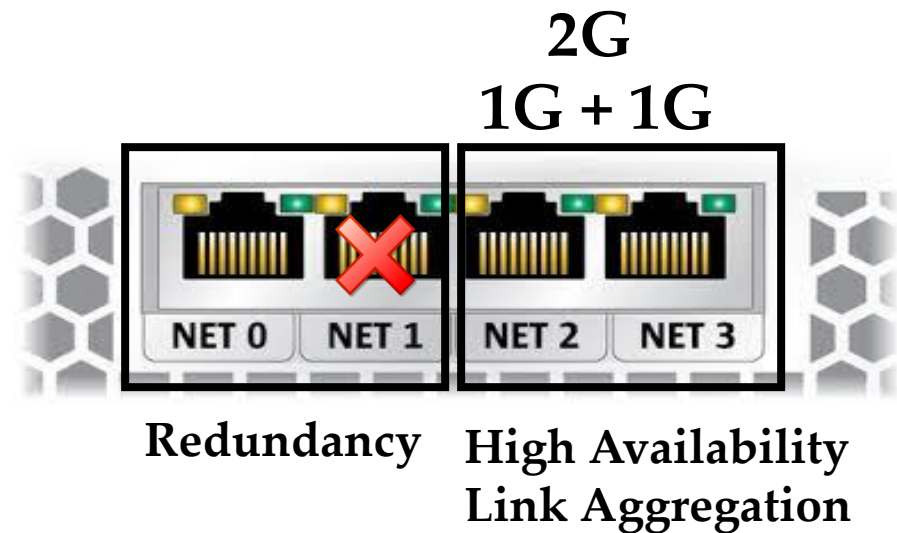
# NIC Bonding

NIC = Network Interface Card (PC or laptop)



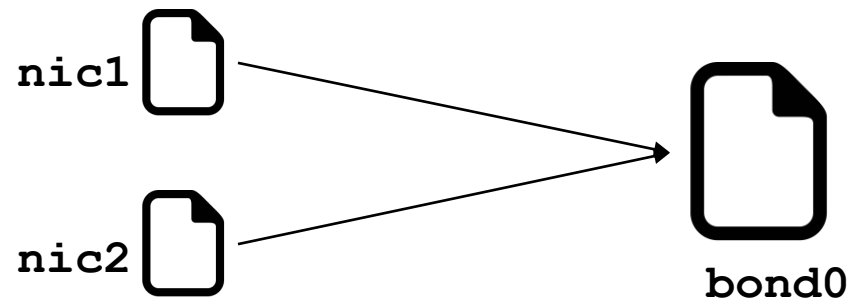
NIC(Network Interface Card) bonding is also known as Network bonding. It can be defined as the aggregation or combination of multiple NIC into a single bond interface.

It's main purpose is to provide high availability and redundancy



# NIC Bonding Procedure

- `modprobe bonding`
- `modinfo bonding`
- Create `/etc/sysconfig/network-scripts/ifcfg-bond0`
- Edit `/etc/sysconfig/network-scripts/ethernet1`
- Edit `/etc/sysconfig/network-scripts/ethernet2`



- Restart network = `systemctl restart network`

# New Network Utilities

What we will learn in this lecture...

- Getting started with **NetworkManager**
- Network configuration methods
  - `nmtui`
  - `nmcli`
  - `nm-connection-editor`
  - **GNOME Settings.**



# New Network Utilities

## ✓ Getting started with NetworkManager

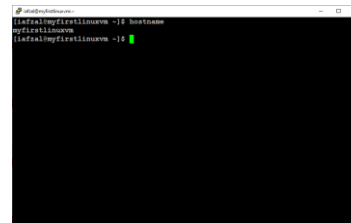
- NetworkManager is a service that provides set of tools designed specifically to make it easier to manage the networking configuration on Linux systems and is the default network management service on RHEL 8
- It makes network management easier
- It provides easy setup of connection to the user
- NetworkManager offers management through different tools such as **GUI**, **nmtui**, and **nmcli**.

# New Network Utilities

## ✓ Network configuration methods

- **nmcli** – Short for network manager command line interface. This tool is useful when access to a graphical environment is not available and can also be used within scripts to make network configuration changes
- **nmtui** – Short for network manager text user interface. This tool can be run within any terminal window and allows changes to be made by making menu selections and entering data
- **nm-connection-editor** - A full graphical management tool providing access to most of the NetworkManager configuration options. It can only be accessed through the desktop or console
- **GNOME Settings** - The network screen of the GNOME desktop settings application allows basic network management tasks to be performed

- Let's practice in our Linux machine...



# **System Updates and Repos**

- yum (CentOS), apt-get (other Linux)
- rpm (Redhat Package Manager)

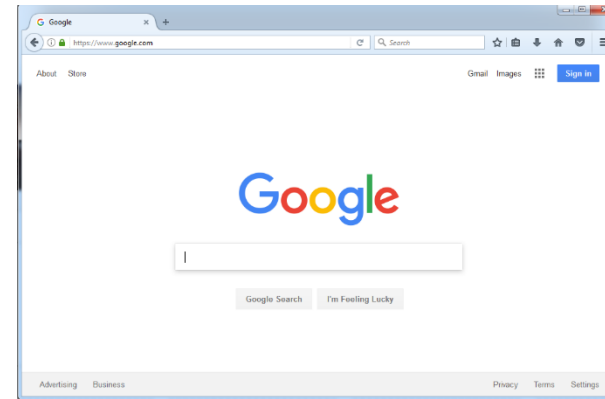
# Advance Package Management

- Installing packages
- Upgrading
- Deleting
- View package details information
- Identify source or location information
- Packages configuration files



# Download Files or Apps

- Example of Windows browser



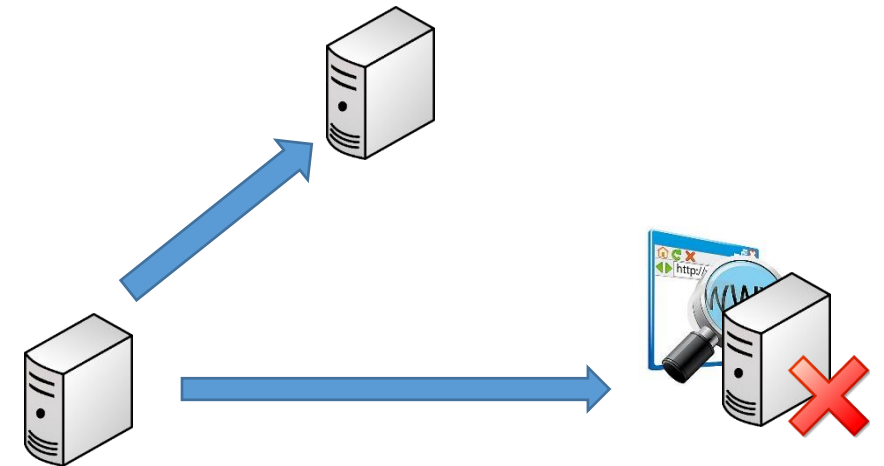
- Linux = **wget**

- Example in Linux:

**wget** <http://website.com/filename>

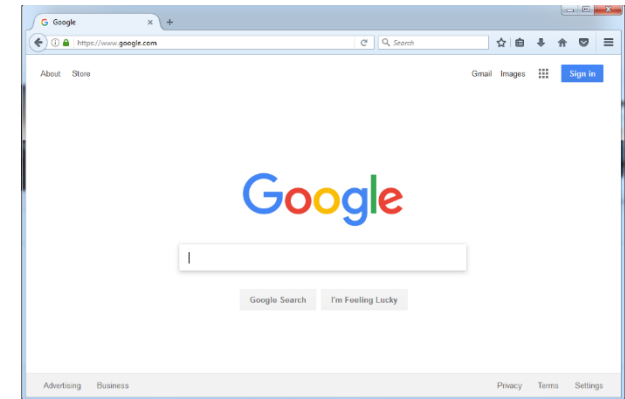
- **Why???**

Most of the servers in corporate environment do **NOT** have internet access



# curl and ping Commands

- Example of Windows browser



- Linux = `curl`

- **Linux** = `ping`

- Example in Linux:

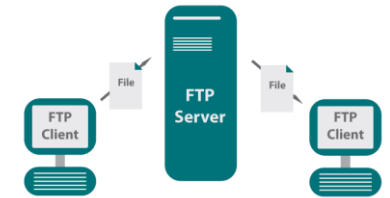
```
curl http://website.com/filename
```

```
curl -O http://website.com/filename
```

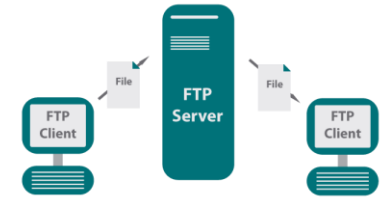
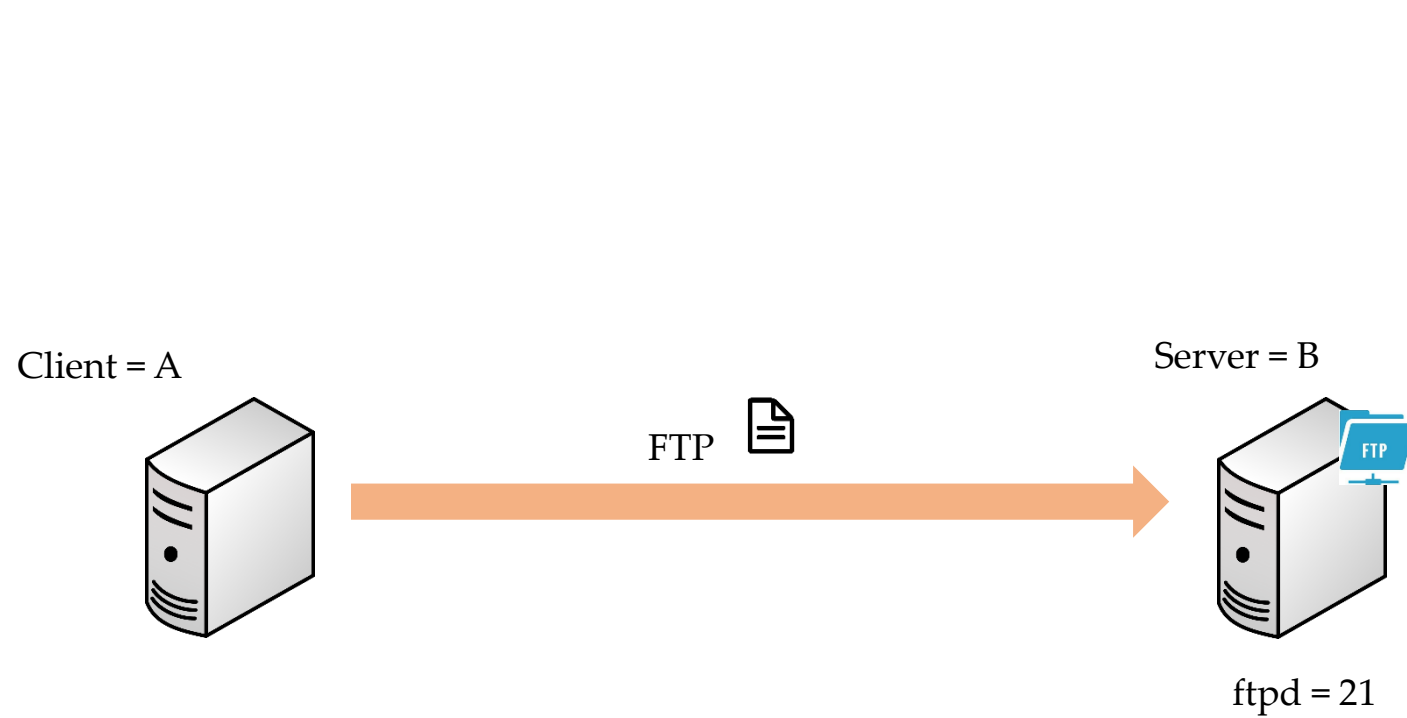
```
ping www.google.com
```

# FTP – File Transfer Protocol

- The File Transfer Protocol is a standard network protocol used for the transfer of computer files between a client and server on a computer network. FTP is built on a client-server model architecture using separate control and data connections between the client and the server. (*Wikipedia*)
- Protocol = Set of rules used by computers to communicate
- Default FTP Port = 21
- For this lecture we need 2 Linux machines
  - **Client** = **MyFirstLinuxVM**
  - **Server** = **LinuxCentOS7**



# FTP – File Transfer Protocol

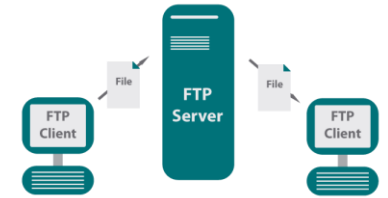




# FTP – File Transfer Protocol

- Install and Configure FTP on the remote server

- `# Become root`
- `# rpm -qa | grep ftp`
- `# ping www.google.com`
- `# yum install vsftpd`
- `# vi /etc/vsftpd/vsftpd.conf` *(make a copy first)*
- Find the following lines and make the changes as shown below:
  - `## Disable anonymous login ##`
    - `anonymous_enable=NO`
  - `## Uncomment ##`
    - `ascii_upload_enable=YES`
    - `ascii_download_enable=YES`
  - `## Uncomment - Enter your Welcome message - This is optional ##`
    - `ftpd_banner>Welcome to UNIXMEN FTP service.`
  - `## Add at the end of this file ##`
    - `use_localtime=YES`
- `# systemctl start vsftpd`
- `# systemctl enable vsftpd`
- `# systemctl stop firewalld`
- `# systemctl disable firewalld`
- `# useradd iafzal` *(if the user does not exist).*



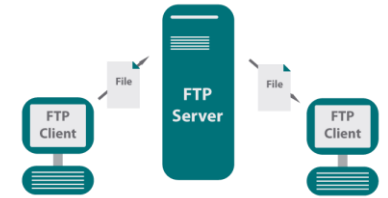
# FTP – File Transfer Protocol

- **Install FTP client on the client server**

- `# Become root`
- `# yum install ftp`
- `# su - iaafzal`
- `$ touch kruger`

- **Commands to transfer file to the FTP server:**

- [ftp 192.168.1.x](#)
- Enter username and password
- `bi`
- `hash`
- `put kruger`
- `bye.`



# SCP – Secure Copy Protocol

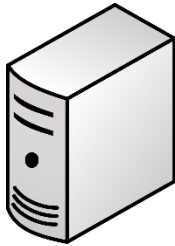
- The Secure Copy Protocol or “SCP” helps to transfer computer files securely from a local to a remote host. It is somewhat similar to the File Transfer Protocol “FTP”, but it adds security and authentication
- Protocol = Set of rules used by computers to communicate
- Default SCP Port = 22 (same as SSH)
- For this lecture we need 2 Linux machines
  - **Client = MyFirstLinuxVM**
  - **Server = LinuxCentOS7**



# SCP – Secure Copy



Client = A

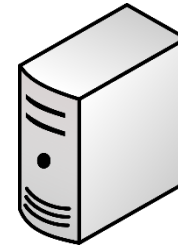


ssh

scp



Server = B



sshd = 22

# SCP – Secure Copy

- **SCP commands to transfer file to the remote server:**
  - Login as yourself (iafzal)
  - touch jack
  - scp jack iafzal@192.168.1.x:/home/iafzal
  - Enter username and password

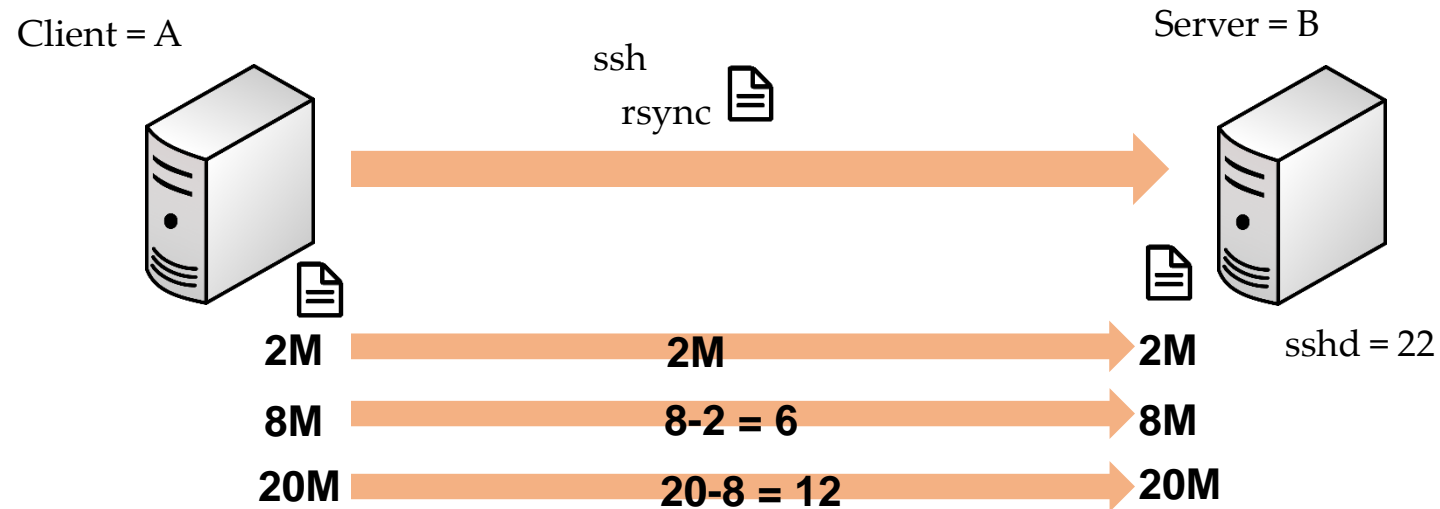


# rsync – Remote Synchronization

- **rsync** is a utility for efficiently transferring and synchronizing files within the same computer or to a remote computer by comparing the modification times and sizes of files
- rsync is a lot faster than ftp or scp
- This utility is mostly used to backup the files and directories from one server to another
- Default rsync Port = 22 (same as SSH)
- For this lecture we need 2 Linux machines
  - **Client = MyFirstLinuxVM**
  - **Server = LinuxCentOS7**



# rsync – Remote Synchronization



# rsync – Remote Synchronization



- **Basic syntax of rsync command**
  - `# rsync options source destination`
- **Install rsync in your Linux machine** *(check if it already exists)*
  - `# yum install rsync` *(On CentOS/Redhat based systems)*
  - `# apt-get install rsync` *(On Ubuntu/Debian based systems)*
- **rsync a file on a local machine**
  - `$ tar cvf backup.tar .` *(tar the entire home directory (/home/iafzal))*
  - `$ mkdir /tmp/backups`
  - `$ rsync -zvh backup.tar /tmp/backups/`
- **rsync a directory on a local machine**
  - `$ rsync -azvh /home/iafzal /tmp/backups/`
- **rsync a file to a remote machine**
  - `$ mkdir /tmp/backups` *(create /tmp/backups dir on remote server)*
  - `$ rsync -avz backup.tar iafzal@192.168.1.x:/tmp/backups`
- **rsync a file from a remote machine**
  - `$ touch serverfile`
  - `$ rsync -avzh iafzal@192.168.1.x:/home/iafzal/serverfile /tmp/backups`





# System Upgrade/Patch Management

- Two type of upgrades

Major version = 5, 6, 7

Minor version = 7.3 to 7.4


Major version = yum  mmand

Minor version = yum update 

Example:

**yum update -y**

**yum update vs. upgrade**

**upgrade** = delete packages 

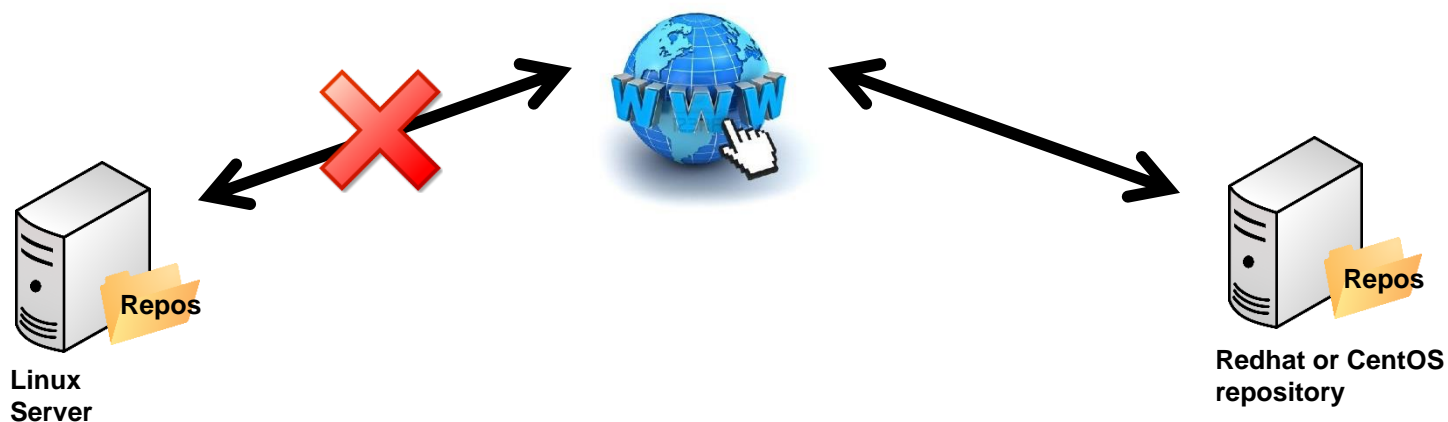
**update** = preserve



# CREATE LOCAL REPOSITORY FROM DVD



- What is local repository?



- Command  
`createrepo`

# SSH AND TELNET

- Telnet = Un-secured connection between computers
- SSH = Secured
- Two type of packages for most of the services
  - Client package
  - Server package



# SSH without a Password

- SSH is a secure way to login from host A to host B
- Repetitive tasks require login without a password

What we will learn...

- How to generate SSH keys on the server
- Add SSH keys to the client
- Verify by logging through SSH.



# DNS = Domain Name System

- Purpose?

Hostname to IP	(A Record)
IP to Hostname	(PTR Record)
Hostname to Hostname	(CNAME Record)

- Files

`/etc/named.conf`  
`/var/named`

- Service

`systemctl restart named`

# Download, Install and Configure DNS

- Create a snapshot of your virtual machine
- Setup:
  - Master DNS
  - Secondary or Slave DNS
  - Client
- Domain Name = lab.local
- IP address = My local IP address on enp0s3
- Install DNS package
  - `yum install bind bind-utils -y`
- Configure DNS (Summary)
  - Modify `/etc/named.conf`
  - Create two zone files (`forward.lab` and `reverse.lab`)
  - Modify DNS file permissions and start the service
- Revert back to snapshot

# HOSTNAME/IP LOOKUP

- Commands used for DNS lookup
  - **nslookup**
  - **dig**

# NTP

- Purpose?

Time synchronization

- File

`/etc/ntp.conf`

- Service

`systemctl restart ntpd`

- Command

`ntpq`



# chronyd

- Purpose? = Time synchronization
- Package name = chronyd
- Configuration file = /etc/chronyd.conf
- Log file = /var/log/chrony
- Service = systemctl start/restart chronyd
- Program command = chronyd.

# New System Utility Command (timedatectl)

- The **timedatectl** command is a new utility for RHEL/CentOS 7/8 based distributions, which comes as a part of the systemd system and service manager
- It is a replacement for old traditional **date** command
- The **timedatectl** command shows/change date, time, and timezone
- It synchronizes the time with NTP server as well
  - You can either use **chronyd** or **ntpd** and make the ntp setting in **timedatectl** as **yes**
  - Or you can use **systemd-timesyncd** daemon to synchronize time which is a replacement for ntpd and chronyd

## ***Please note:***

*Redhat/CentOS does not provide this daemon in its standard repo. You will have to download it separately.*

# New System Utility Command (timedatectl)

## Lab exercise:

- To check time status
  - `timedatectl`
- To view all available time zones
  - `timedatectl list-timezones`
- To set the time zone
  - `timedatectl set-timezone "America/New_York"`
- To set date
  - `timedatectl set-time YYYY-MM-DD`
- To set date and time
  - `timedatectl set-time '2015-11-20 16:14:50'`
- To start automatic time synchronization with a remote NTP server
  - `timedatectl set-ntp true.`

# Sendmail - OLD

- Purpose?

Send and receive emails

- Files

`/etc/mail/sendmail.mc`

`/etc/mail/sendmail.cf`

`/etc/mail`

- Service

`systemctl restart sendmail`

- Command

`mail -s "subject line" email@mydomain.com`

# Sendmail

- **Sendmail** is a program in Linux operating systems that allows systems administrator to send email from the Linux system
- It uses SMTP (Simple Mail Transfer Protocol)
- SMTP port = 25
- It attempts to deliver the mail to the intended recipient immediately and, if the recipient is not present, it queues messages for later delivery.



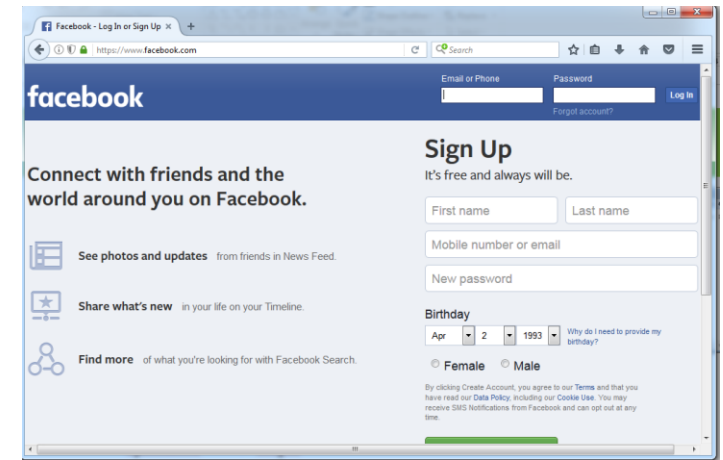
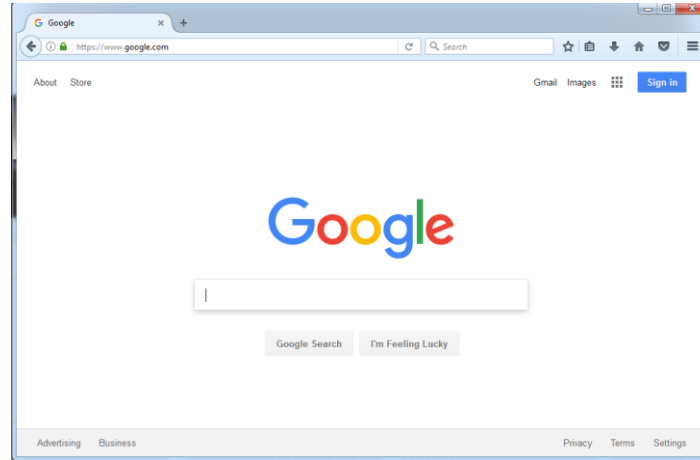
# Sendmail

- Sendmail installation and configuration
  - # `su -` (*Login as root*)
  - # `rpm -qa | grep sendmail` (*verify if it is already installed*)
  - # `yum install sendmail sendmail-cf`
  - # `vi /etc/mail/sendmail.mc`
  - # `systemctl start sendmail`
  - # `systemctl enable sendmail`
  - # `systemctl stop firewalld`
  - # `systemctl disable firewalld`



# Web Server (httpd)

- Purpose = Serve webpages



- Service or Package name = **httpd**
- Files = **/etc/httpd/conf/httpd.conf**  
= **/var/www/html/index.html**
- Service  

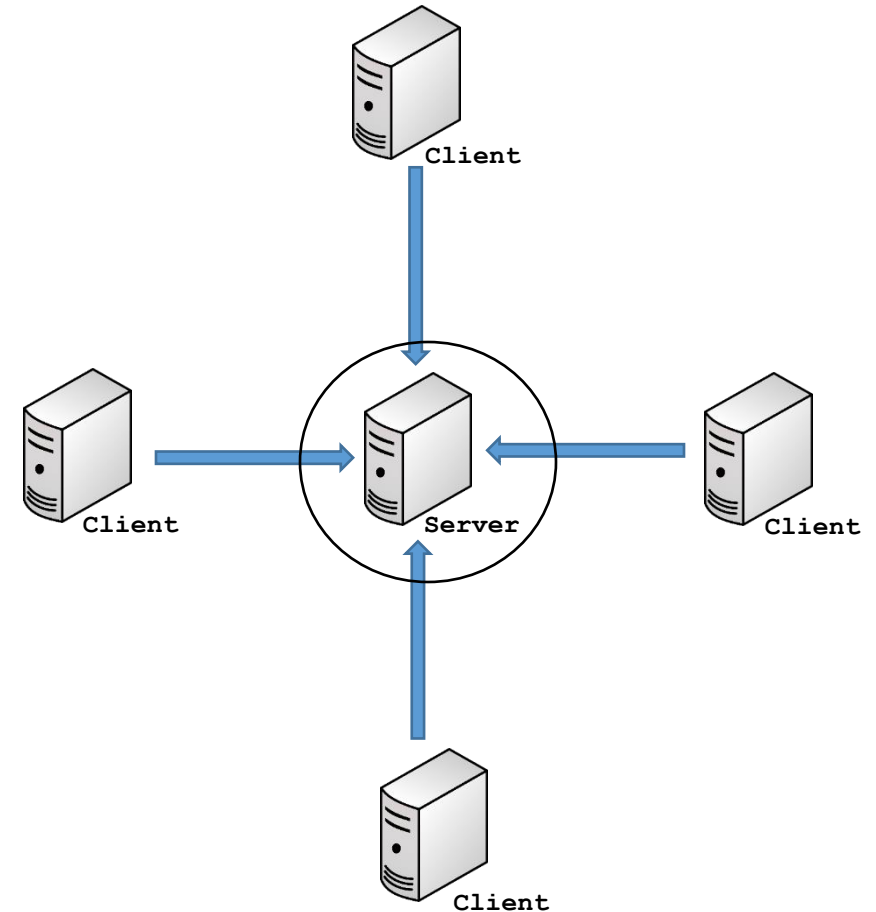
```
systemctl restart httpd
```

```
systemctl enable httpd
```
- Log Files = **/var/log/httpd/**

# CENTRAL LOGGER (RSYSLOG)

- Purpose = Generate logs or collect logs from other servers
- Service or package name = **rsyslog**
- Configuration file= **/etc/syslog.conf**
- Service

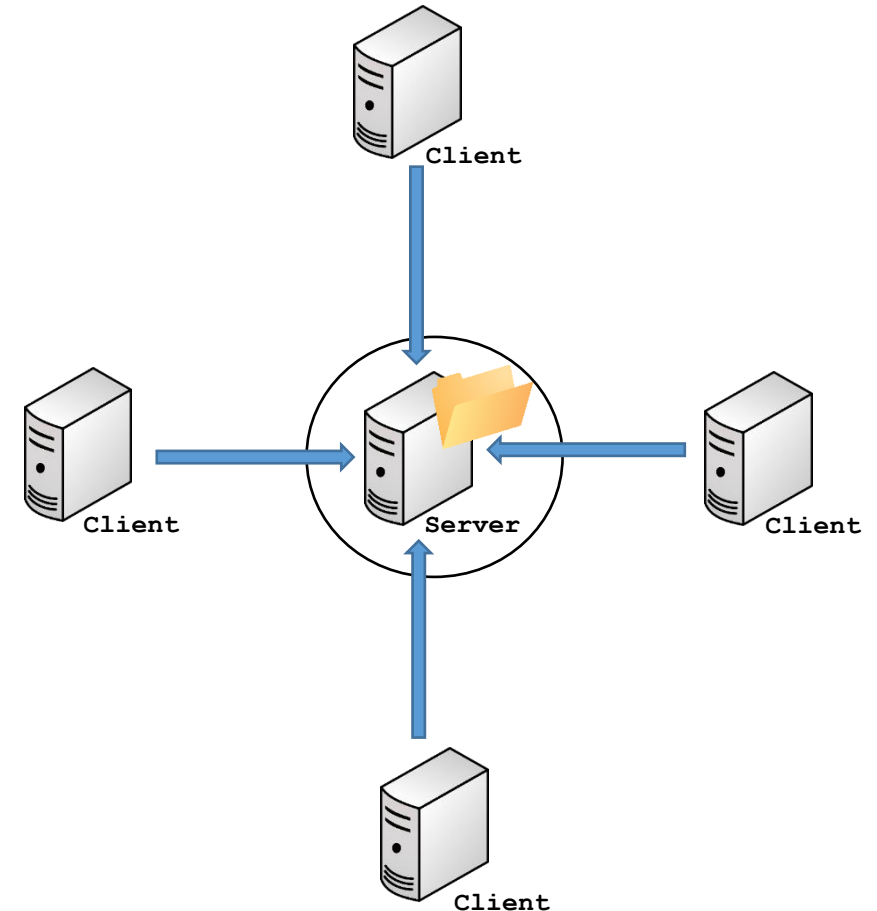
```
systemctl restart rsyslog  
systemctl enable rsyslog
```





# NETWORK FILE SYSTEM (NFS)

- Purpose = Share files or directories (filesystem)
- Service or package name = **nfs-utils**
- Configuration file =  
**/etc/fstab, /etc/exports,  
/etc/sysconfig/nfs**
- Service  
**systemctl restart nfs-server**  
**systemctl enable nfs-server**



# LINUX OS HARDENING



- **User Account**
- **Remove un-wanted packages**
- **Stop un-used Services**
- **Check on Listening Ports**
- **Secure SSH Configuration**
- **Enable Firewall (iptables/firewalld)**
- **Enable SELinux**
- **Change Listening Services Port Numbers**
- **Keep your OS up to date (security patching)**

# OpenLDAP Installation

- **What is OpenLDAP?**
- **OpenLDAP Service**
  - `slapd`
- **Start or stop the service**
  - `systemctl start slapd`
  - `systemctl enable slapd`
- **Configuration Files**
  - `/etc/openldap/slapd.d`

# Trace Network Traffic (traceroute)

- The traceroute command is used in Linux to map the journey that a packet of information undertakes from its source to its destination. One use for traceroute is to locate when data loss occurs throughout a network, which could signify a node that's down.
- Because each hop in the record reflects a new server or router between the originating PC and the intended target, reviewing the results of a traceroute scan also lets you identify slow points that may adversely affect your network traffic.

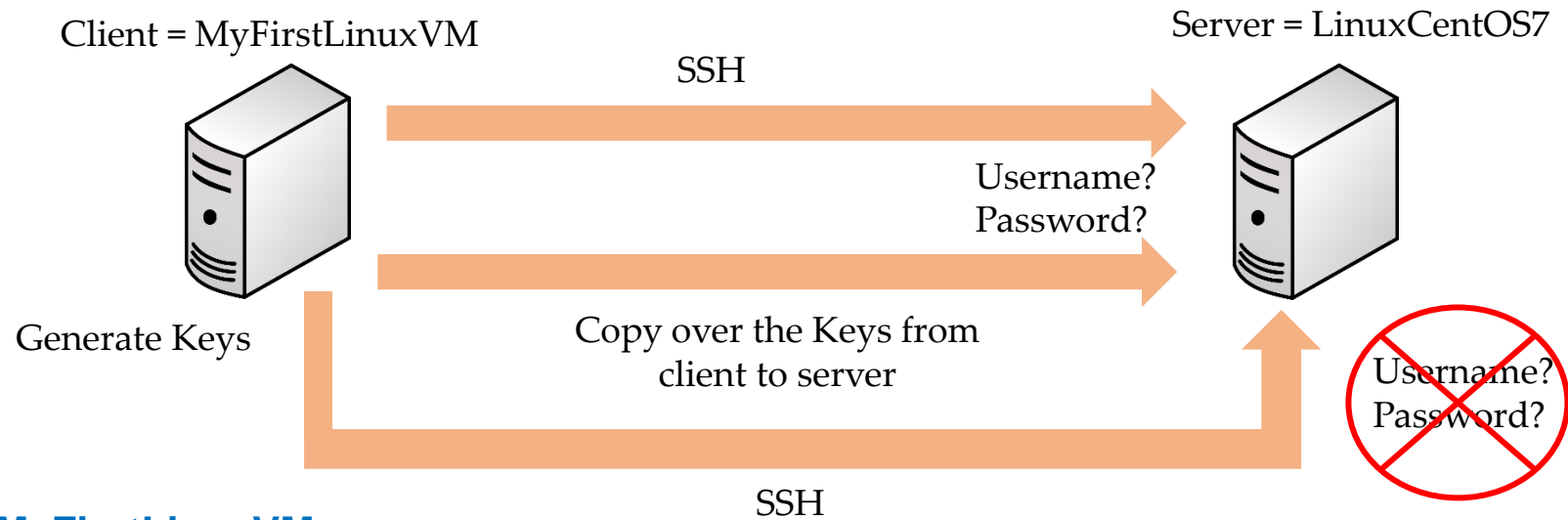
- Example

```
# traceroute www.google.com
```

# Access Remote Server without Password (SSH-Keys)

- Two reasons to access a remote machine
  - Repetitive logins
  - Automation through scripts
- Keys are generated at user level
  - iafzal
  - root

# Access Remote Server without Password (SSH-Keys)



## Client = MyFirstLinuxVM

**Step 1** — Generate the Key

```
# ssh-keygen
```

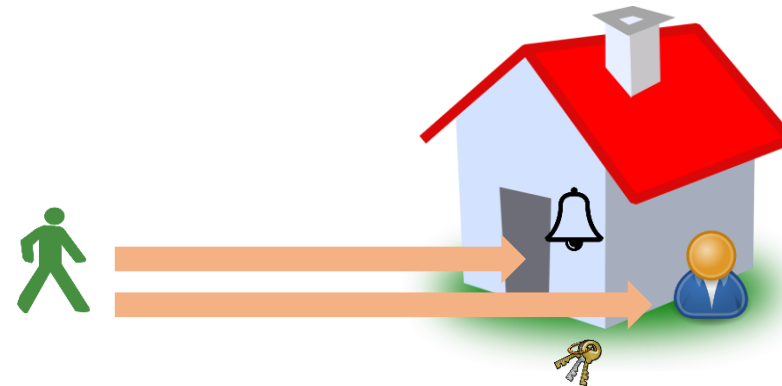
**Step 2** — Copy the Key to the server

```
# ssh-copy-id root@192.168.1.x
```

**Step 3** — Login from client to server

```
# ssh root@192.168.1.x
```

```
# ssh -l root 192.168.1.x
```



# Cockpit

- Cockpit is a server administration tool sponsored by Red Hat, focused on providing a modern-looking and user-friendly interface to manage and administer servers
- Cockpit is the easy-to-use, integrated, glanceable, and open web-based interface for your servers
- The application is available in most of the Linux distributions such as, CentOS, Redhat, Ubuntu and Fedora
- It is installed in Redhat 8 by default and it is optional in version 7
- It can monitor system resources, add or remove accounts, monitor system usage, shut down the system and perform quite a few other tasks all through a very accessible web connection



# Install, Configure and Manage Cockpit

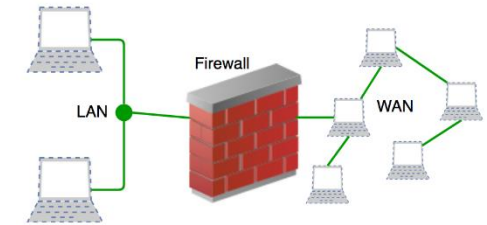
- Check for network connectivity
  - **ping www.google.com**
- Install cockpit package as root
  - **yum/dnf install cockpit -y** *(For RH or CentOS)*
  - **apt-get install cockpit** *(For Ubuntu)*
- Start and enable the service
  - **systemctl start|enable cockpit**
- Check the status of the service
  - **systemctl status cockpit**
- Access the web-interface
  - **https://192.168.1.x:9090**



# Introduction to Firewall

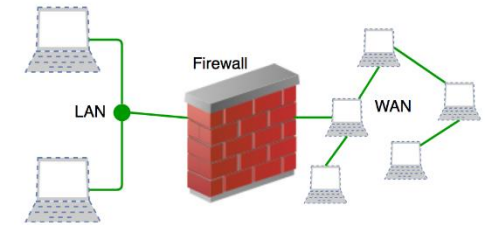
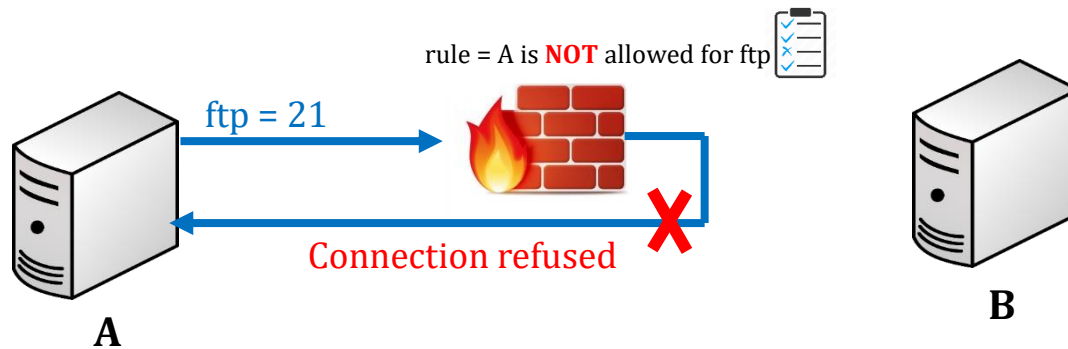
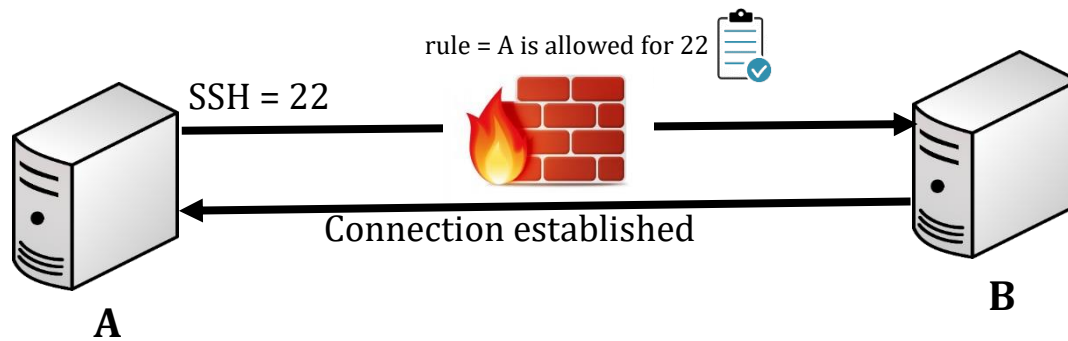
1/2

- What is Firewall
  - A wall that prevents the spread of fire
  - When data moves in and out of a server its packet information is tested against the firewall rules to see if it should be allowed or not
  - In simple words, a firewall is like a watchman, a bouncer, or a shield that has a set of rules given and based on that rule they decide who can enter and leave
- There are 2 type of firewalls in IT
  - Software = Runs on operating system
  - Hardware = A dedicated appliance with firewall software



# Introduction to Firewall

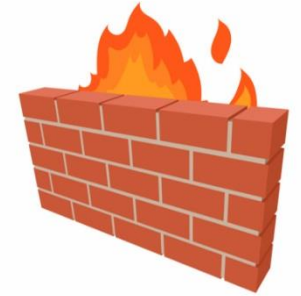
2/2



# Firewall (*firewalld*)

1/2

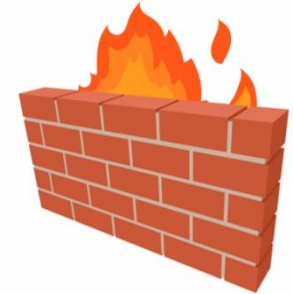
- Firewalld works the same way as iptables but of course it has its own commands
  - **firewall-cmd**
- It has a few pre-defined service rules that are very easy to turn on and off
  - **Services such as: NFS, NTP, HTTPD etc.**
- Firewalld also has the following:
  - **Table**
  - **Chains**
  - **Rules**
  - **Targets**



# Firewall (*firewalld*)

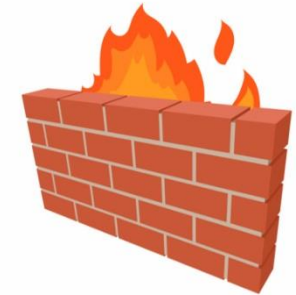
2/2

- You can run one or the other
  - iptables or firewalld
- Make sure iptables is stopped, disabled and mask
  - `systemctl stop iptables`
  - `systemctl disable iptables`
  - `systemctl mask iptables`
- Now check if firewalld package is installed
  - `rpm -qa | grep firewalld`
- Start firewalld
  - `systemctl start/enable firewalld`
- Check the rule of firewalld
  - `firewall-cmd --list-all`
- Get the listing of all services firewalld is aware of:
  - `firewall-cmd --get-services`
- To make firewalld re-read the configuration added
  - `firewall-cmd --reload`

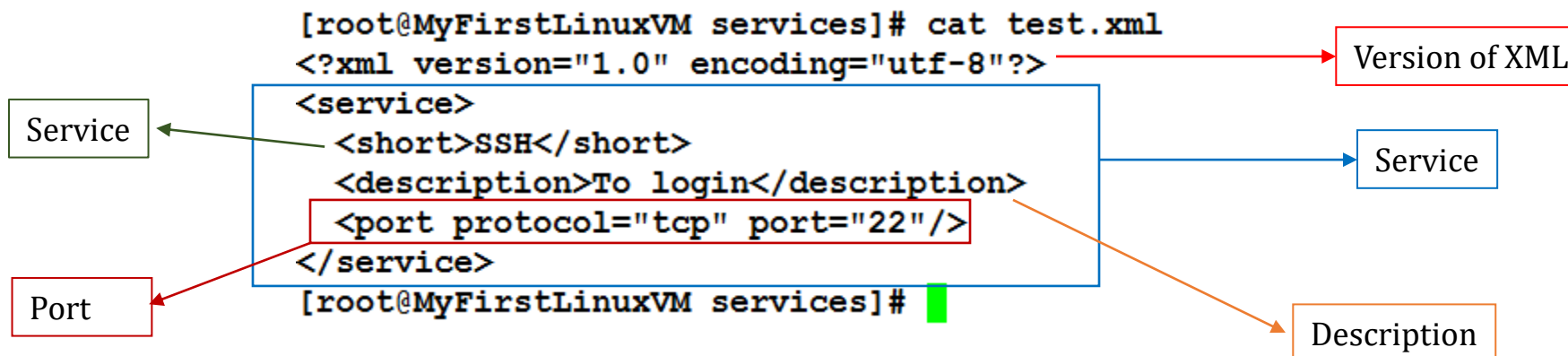


# Firewall *(firewalld - Practical Examples)*

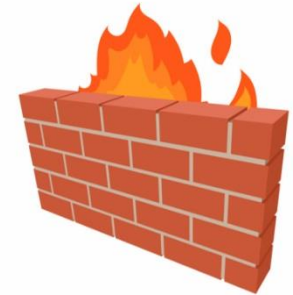
1/3

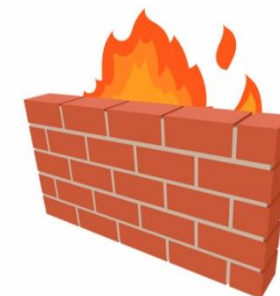


- The firewalld has multiple zone, to get a list of all zones
  - `firewall-cmd --get-zones`
- To get a list of active zones
  - `firewall-cmd --get-active-zones`
- To get firewall rules for public zone
  - `firewall-cmd --zone=public --list-all`
  - OR
  - `firewall-cmd --list-all`
- All services are pre-defined by firewalld. What if you want to add a 3<sup>rd</sup> party service
  - `/usr/lib/firewalld/services/allservices.xml`
  - Simply cp any .xml file and change the service and port number



- To add a service (http)
  - `firewall-cmd --add-service=http`
- To remove a service
  - `firewall-cmd --remove-service=http`
- To reload the firewalld configuration
  - `firewall-cmd --reload`
- To add or remove a service permanently
  - `firewall-cmd --add-service=http --permanent`
  - `firewall-cmd --remove-service=http --permanent`
- To add a service that is not pre-defined by firewalld
  - `/usr/lib/firewalld/services/allservices.xml`
  - Simply cp any .xml file sap.xml and change the service and port number (32)
  - `systemctl restart firewalld`
  - `firewall-cmd --get-services` (to verify new service)
  - `Firewall-cmd --add-service=sap`





- To add a port
  - `firewall-cmd --add-port=1110/tcp`
- To remove a port
  - `firewall-cmd --remove-port=1110/tcp`
- To reject incoming traffic from an IP address
  - `firewall-cmd --add-rich-rule='rule family="ipv4" source address="192.168.0.25" reject'`
- To block and unblock ICMP incoming traffic
  - `firewall-cmd --add-icmp-block-inversion`
  - `firewall-cmd --remove-icmp-block-inversion`
- To block outgoing traffic to a specific website/IP address
  - `host -t a www.facebook.com = find IP address`
  - `firewall-cmd --direct --add-rule ipv4 filter OUTPUT 0 -d 31.13.71.36 -j DROP`

# Tune System Performance

1/8

Linux system comes fine tuned by default when you install, however there are a few tweaks that can be done based on system performance and application requirements

In this lesson we will learn...

- Optimize system performance by selecting a tuning profile managed by the **tuned** daemon
- Prioritize or de-prioritize specific processes with the **nice** and **renice** commands



# Tune System Performance

2/8

## What is tuned?

- Tuned pronounced as tune-d
- **Tune** is for system tuning and **d** is for daemon
- It is **systemd** service that is used to tune Linux system performance
- It is installed in CentOS/Redhat version 7 and 8 by default
- **tuned** package name is **tuned**
- The **tuned** service comes with pre-defined profiles and settings (*List of profile will be discussed in the next page*)
- Based on selected profile the **tuned** service automatically adjust system to get the best performance. E.g. **tuned** will adjust networking if you are downloading a large file or it will adjust IO settings if it detects high storage read/write
- The tuned daemon applies system settings when the service starts or upon selection of a new tuning profile.

# Tune System Performance

(**tuned** profiles)

3/8

Tuned profile	Purpose
balanced	deal for systems that require a compromise between power saving and performance
desktop	Derived from the balanced profile. Provides faster response of interactive applications
Throughput-performance	Tunes the system for maximum throughput
Latency-performance	Ideal for server systems that require low latency at the expense of power consumption
network-latency	Derived from the latency-performance profile. It enables additional network tuning parameters to provide low network latency
Network-throughput	Derived from the throughput-performance profile. Additional network tuning parameters are applied for maximum network throughput
powersave	Tunes the system for maximum power saving
oracle	Optimized for Oracle database loads based on the throughput-performance profile
virtual-guest	Tunes the system for maximum performance if it runs on a virtual machine
virtual-host	Tunes the system for maximum performance if it acts as a host for virtual machines

# Tune System Performance

4/8

- Check if tuned package has been installed  
`rpm -qa | grep tuned`
- Install tuned package if NOT installed already  
`yum install tuned`
- Check **tuned** service status  
`systemctl status|enable|start tuned`  
`systemctl enable tuned` *(To enable at boot time)*
- Command to change setting for tuned daemon  
`tuned-adm`
- To check which profile is active  
`tuned-adm active`
- To list available profiles  
`tuned-adm list.`

# Tune System Performance

5/8

- To change to desired profile  
`tuned-adm profile profile-name`
- Check for tuned recommendation  
`tuned-adm recommend`
- Turn off tuned setting daemon  
`tuned-adm off`
- Change profile through web console  
Login to <https://192.168.1.x:9090>  
Overview → Configuration → Performance profile

# Tune System Performance

(**nice**/**renice**)

6/8

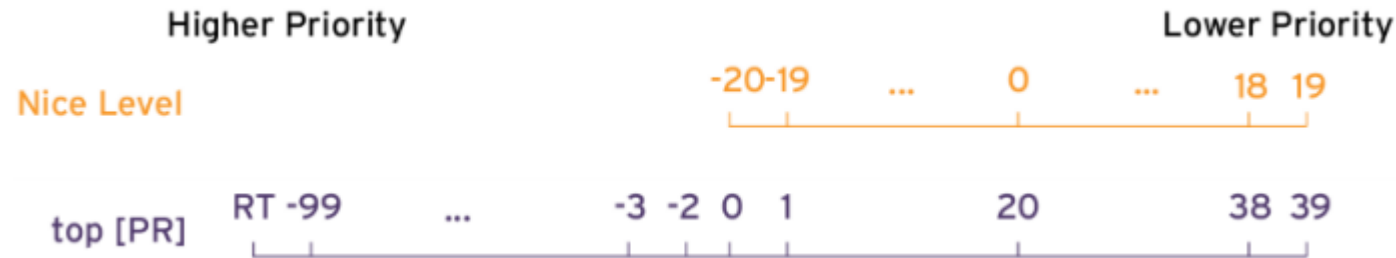
- Another way of keeping your system fine-tuned is by prioritizing processes through **nice** and **renice** command
- If a server has 1 CPU then it can execute 1 computation/process at a time as they come in (*first come first served*) while other processes must wait
- With **nice** and **renice** commands we can make the system to give preference to certain processes than others
- This priority can be set at 40 different levels
- The nice level values range from -20 (highest priority) to 19 (lowest priority) and by default, processes inherit their nice level from their parent, which is usually 0.

# Tune System Performance

(nice/renice)

7/8

- To check process priority  
**top**



Nice value is a user-space and priority PR is the process's actual priority that use by Linux kernel. In Linux system priorities are 0 to 139 in which 0 to 99 for real time and 100 to 139 for users

- Process priority can be viewed through ps command as well with the right options  
**\$ ps axo pid,comm,nice,cls --sort=-nice**

# Tune System Performance

(nice/renice)

8/8

- To set the process priority  
`nice -n # process-name`  
e.g. `nice -n -15 top`
- To change the process priority  
`renice -n # process-name`  
e.g. `renice -n 12 PID.`

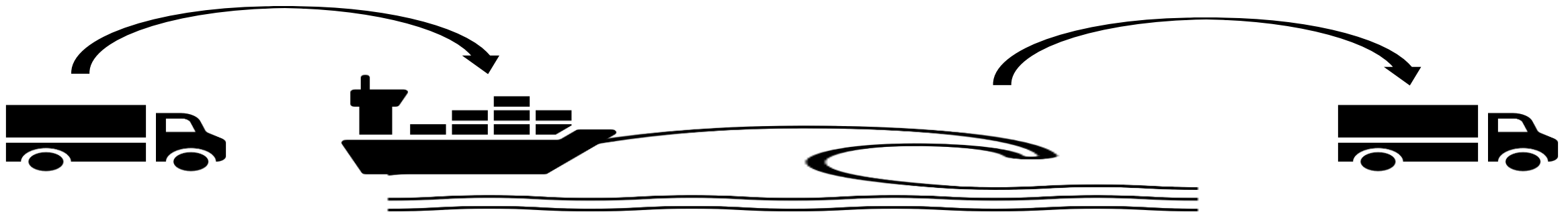
# Run Containers

## What is a Container?

- The term container and the concept came from the shipping container



- These containers are shipped from city to city and country to country
- No matter which part of the world you go to, you will find these containers with the exact same measurements... **YOU KNOW WHY???**
- Because around the world all docks, trucks, ships and warehouses are built to easily transport and store them

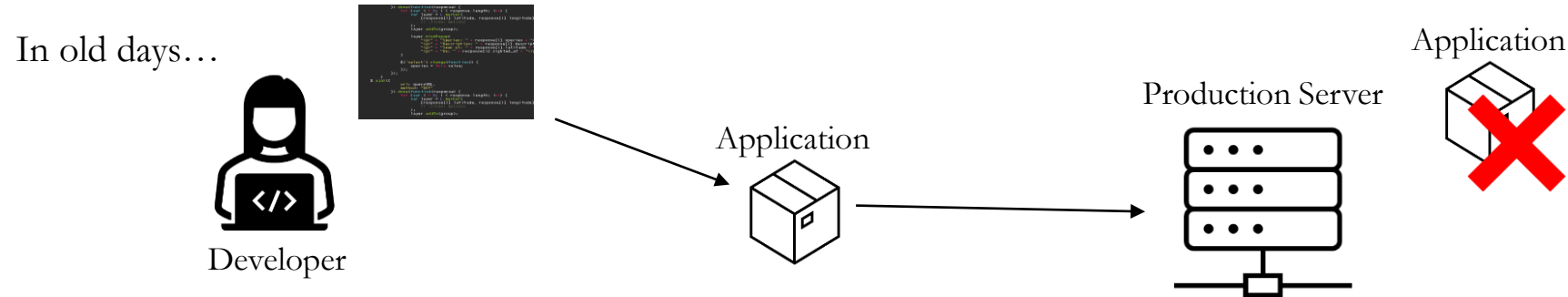




# Run Containers

## What is a Container?

Now when we are talking about containers in IT we are fulfilling somewhat similar purpose



- Then came the container technology which allowed developers or programmer to test and build applications on any computer just by putting it in a container (*bundled in with the software code, libraries and configuration files*) and then run on another computer regardless of its architecture
- You can move the application anywhere without moving its OS just like moving the actual physical container anywhere that would fit on any dockyard, truck, ship or warehouse
- An OS can run single or multiple containers at the same time

# Run Containers

## What is a Container?

Now when we are talking about containers in IT we are fulfilling somewhat similar purpose

### **Please Note:**

**Container technology is mostly used by developers or programmers who write codes to build applications**

**As a system administrator your job is to install, configure and manage them.**

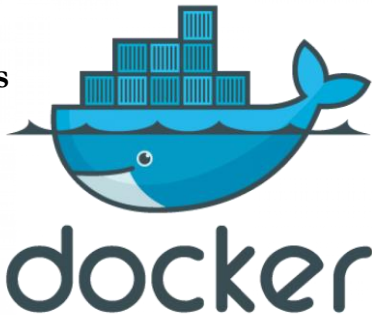
You can move the application anywhere without moving its OS just like moving the actual physical container anywhere that would fit on any dockyard, truck, ship or warehouse

- An OS can run single or multiple containers at the same time

# Run Containers

## What are the Container Software?

Developed by:  
Solomon Hykes



Released on:  
March 20<sup>th</sup> 2013

- Docker is the software used to create and manage containers
- Just like any other package, docker can be installed on your Linux system and its service or daemon can be controlled through native Linux service management tool

Developed by:  
 Red Hat



Released on:  
August 2018

podman

- Podman is an alternative to docker
- Docker is not supported in RHEL 8
- It is daemon less, open source, Linux-native tool designed to develop, manage, and run containers.

# Run Containers

## Getting Familiar with Redhat Container Technology

Red Hat provides a set of command-line tools that can operate without a container engine, these include:

- **podman** - for directly managing pods and container images (run, stop, start, ps, attach, etc.)
- **buildah** - for building, pushing, and signing container images
- **skopeo** - for copying, inspecting, deleting, and signing images
- **runc** - for providing container run and build features to podman and buildah
- **crun** - an optional runtime that can be configured and gives greater flexibility, control, and security for rootless containers.

## Getting Familiar with podman Container Technology

When you hear about containers then you should know the following terms as well

- **images** – containers can be created through images and containers can be converted to images
- **pods** – Group of containers deployed together on the host. In the podman logo there are 3 seals grouped together as a pod.



podman

# Run Containers

## Building, Running and Managing Containers



To install podman

- **yum/dnf install podman -y**
- **yum install docker -y** *(For dockers)*

Creating alias to docker

- **alias docker=podman**

Check podman version

- **podman -v**

Getting help

- **podman --help** or **man podman**

Check podman environment and registry/repository information

- **podman info** *(If you are trying to load a container image, then it will look at the local machine and then go through each registry by the order listed)*

To search a specific image in repository.

- **podman search httpd**

# Run Containers

## Building, Running and Managing Containers



To list any previously downloaded podman images

- **podman images**

To download available images

- **podman pull docker.io/library/httpd**
- **podman images** *(Check downloaded image status)*

To list podman running containers

- **podman ps**

To run a downloaded httpd containers

- **podman run -dt -p 8080:80/tcp docker.io/library/httpd**  
*(d=detach, t=get the tty shell, p=port)*
- **podman ps**      *or Check httpd through web browser*

To view podman logs.

- **podman logs -l**

# Run Containers

## Building, Running and Managing Containers



To stop a running container

- **podman stop con-name** *(con-name from podman ps command)*
- **podman ps** *(To list running containers)*

To run a multiple containers of httpd by changing the port #

- **podman run -dt -p 8081:80/tcp docker.io/library/httpd**
- **podman run -dt -p 8082:80/tcp docker.io/library/httpd**
- **podman ps**

To stop and start a previously running container

- **podman stop|start con-name**

To create a new container from the downloaded image

- **podman create --name httpd-con docker.io/library/httpd**

To start the newly created container.

- **podman start httpd-con**

# Run Containers

## Building, Running and Managing Containers



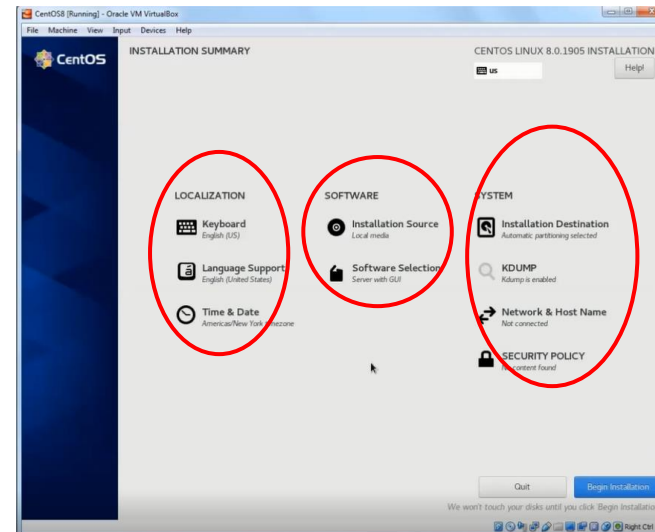
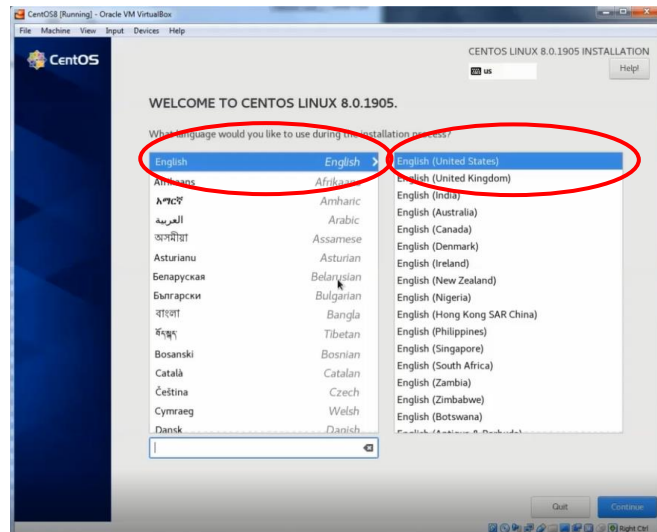
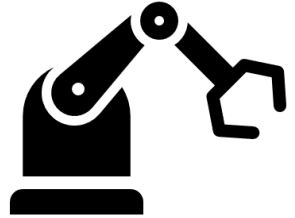
### Manage containers through systemd

- First you have to generate a unit file
  - `podman generate systemd --new --files --name httpd-con`
- Copy it to systemd directory
  - `cp /root/container-httpd.service /etc/systemd/system`
- Enable the service
  - `systemctl enable container-httpd-con.service`
- Start the service.
  - `systemctl start container-httpd-con.service`



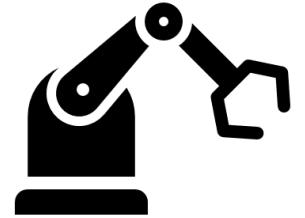
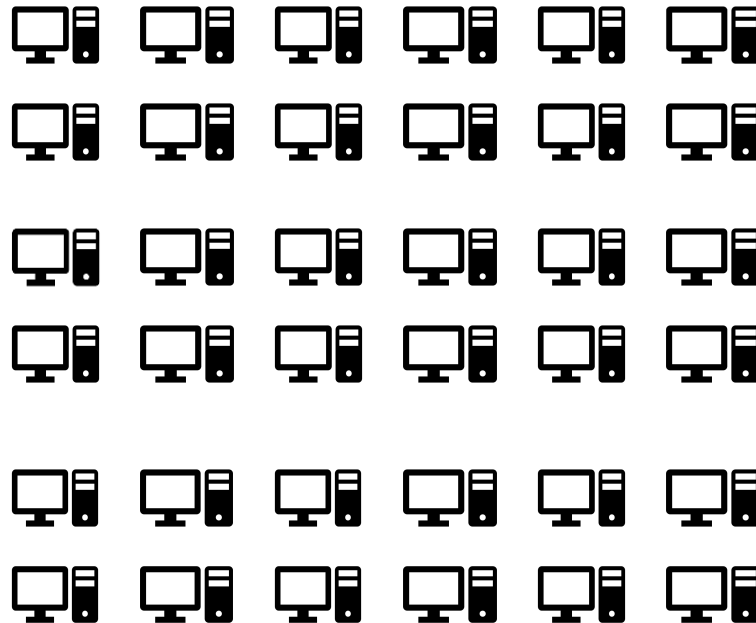
# Kickstart (Automate Linux Installation)

- Kickstart is a method to automate the Linux installation without the need for any intervention from the user
- With the help of kickstart you can automate questions that are asked during the installation. e.g.
  - Language and time zone
  - How the drives should be partitioned
  - Which packages should be installed etc.

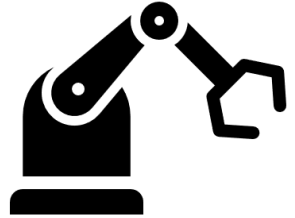


# Kickstart (Automate Linux Installation)

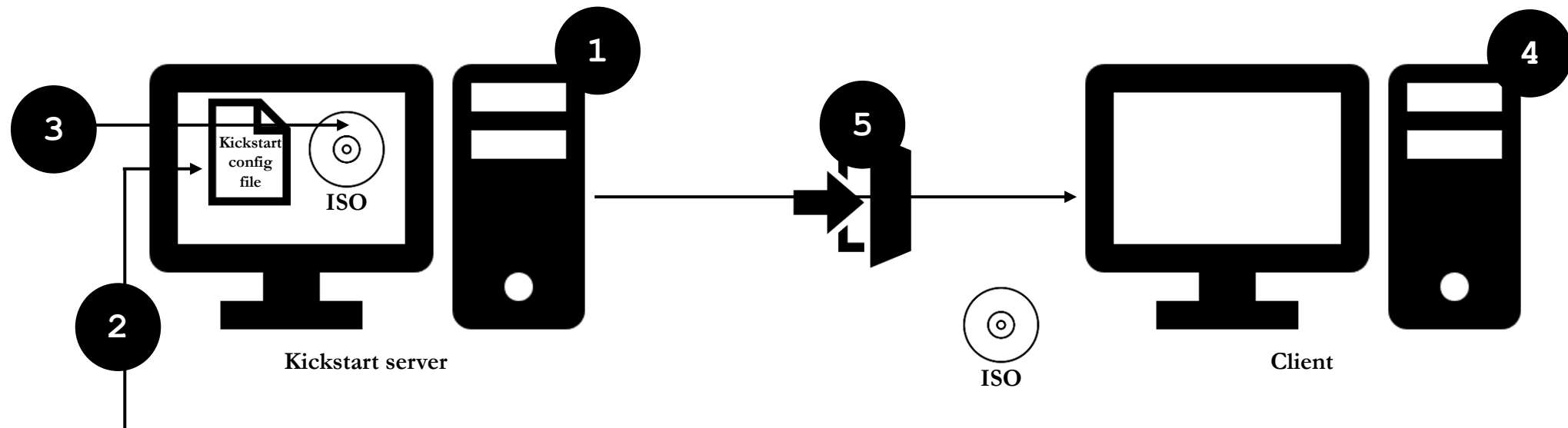
- Purpose?



# Kickstart (Automate Linux Installation)



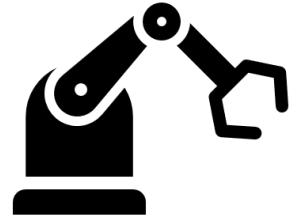
- To use Kickstart, you must:
  1. Choose a Kickstart server and create/edit a Kickstart file
  2. Make the Kickstart file available on a network location
  3. Make the installation source available
  4. Make boot media available for client which will be used to begin the installation
  5. Start the Kickstart installation



Network = NFS, FTP, **HTTP**, or HTTPS

# Kickstart (Automate Linux Installation)

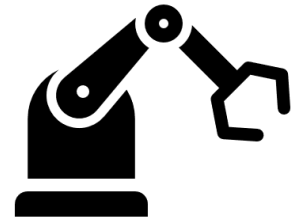
- CentOS/Redhat 7
  - Kickstart program can be downloaded which allows you to define parameters through the GUI
    - **yum install system-config-kickstart**
  - Or you can use the installation kickstart file which was created during the first installation (**anaconda-ks.cfg**)
- CentOS/Redhat 8
  - There is no GUI available to edit the file
- Why changed?
  - Most systems are virtual and templates can be used
  - Automation software are in used such as Anisble.



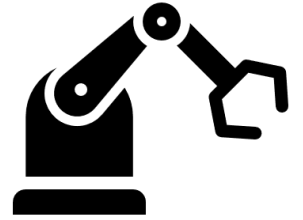
# Kickstart (Automate Linux Installation)

- Step by step procedure for Kickstart

1. Identify the server
2. Take a snapshot of the server
3. Install kickstart configurator (for version 7)
  - `yum install system-config-kickstart`
4. Start the kickstart file configurator and define parameters OR use the `/root/anaconda-ks.cfg`
  - `system-config-kickstart` (To start the configurator)
  - We will use anaconda installation kickstart file and change the hostname only
5. Make sure httpd package is installed, if not then install the package and start the httpd service
  - `rpm -qa | grep http`
  - `yum/dnf install httpd`
  - `systemctl start httpd`
  - `systemctl enable httpd.`



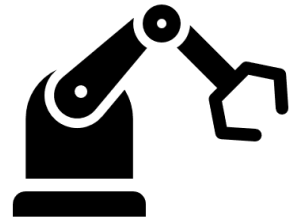
# Kickstart (Automate Linux Installation)



6. Copy kickstart file to httpd directory and change the permissions
  - `cp /root/anaconda-ks.cfg /var/www/html`
  - `chmod a+r /var/www/html/anaconda-ks.cfg`
  - `systemctl stop|disable firewalld`
  - Check file through browser on another PC `http://192.168.1.x/anaconda-ks.cfg`
7. Create a new VM and attach the CentOS iso image
8. Change the network adapter to Bridged adapter
9. Hit Esc
10. `boot: linux ks=http://192.168.1.x/anaconda-ks.cfg`  
For NFS → `boot: linux inst.ks=nfs:192.168.1.x:/rhel8`
11. Wait and enjoy the automated installation

# Kickstart (Automate Linux Installation)

## Kickstart for clients with static IP



```
boot: linux ks=http://server.example.com/ks.cfg ksdevice=eth0 IP:192.168.1.50  
netmask=255.255.255.0 gateway=192.168.1.1
```

Where:

**ksdevice** = is the network adapter of the client

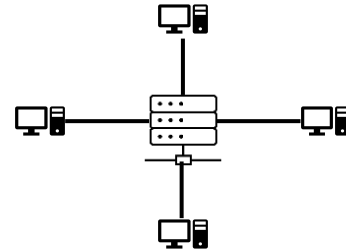
**IP** = IP you are assigning to the client

**netmask** = Subnet mask for the client

**gateway** = Gateway IP address for the client

# DHCP

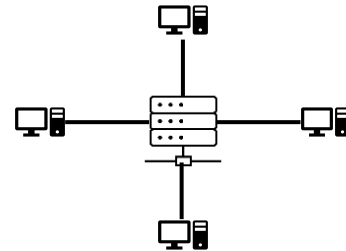
**In this video I will show you how to setup DHCP server conceptually because if you want to setup DHCP server on your Linux machine then you will have to re-configure your router/modem in your home which can route DHCP traffic to your new DHCP server.  
Reconfiguring router/modem will make all your devices at home lose the network connectivity**



- DHCP stands for Dynamic Host Configuration Protocol
- In order to communicate over the network, a computer needs to have an IP address
- DHCP server is responsible to automatically assign IP addresses to servers, laptops, desktops, and other devices on the network
- Wait a second...
  - Right now in our home how IPs are assigned to our devices?
    - Answer → The router or gateway given to you by your ISP provider
  - How IPs are assigned in corporate world?
    - Answer → Dedicated routers run DHCP service to assign IPs on the network

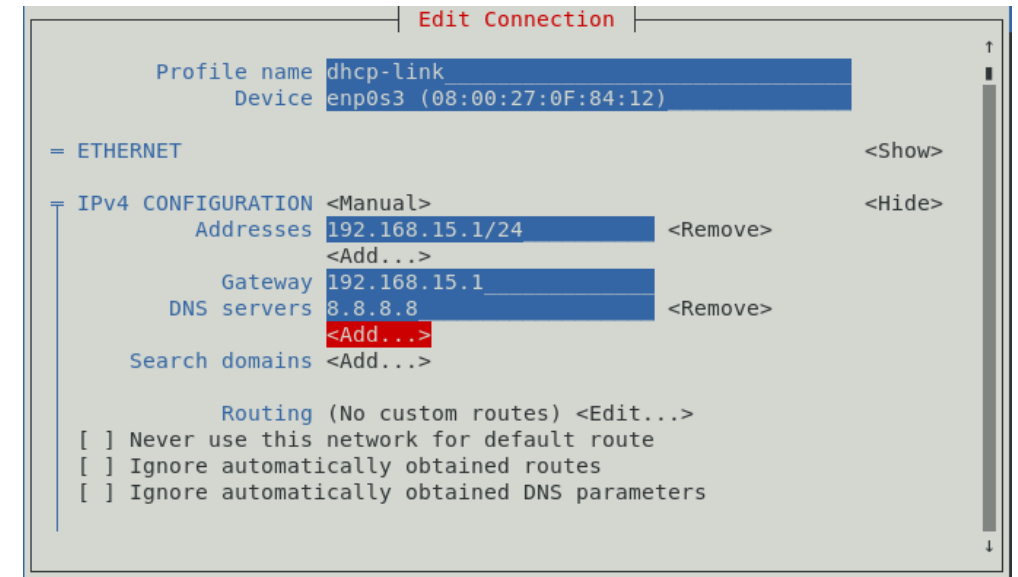
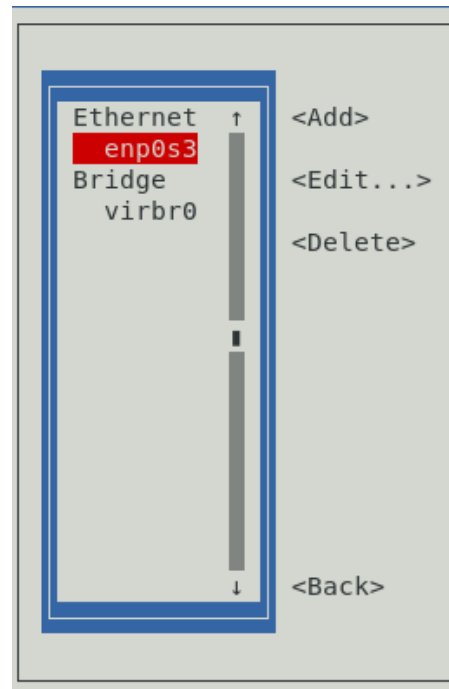
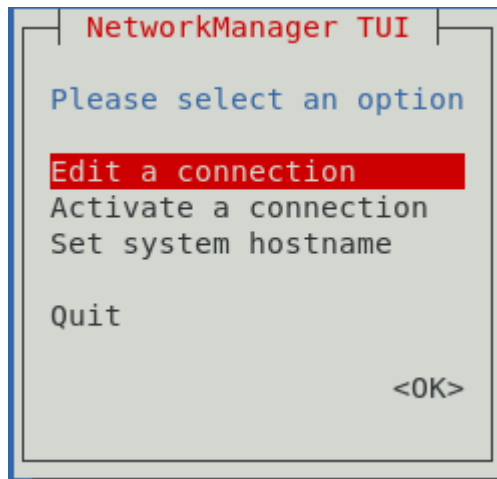


# DHCP

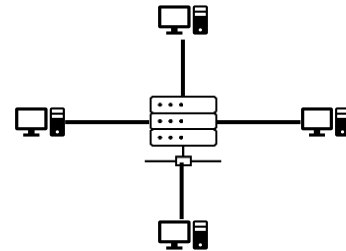


## Step by steps instructions

- Pick a server to be your DHCP and take a snapshot
- Assign a static IP to the DHCP server
  - `vi /etc/sysconfig/network/enp0s3`
  - Or simply run `nmtui` command to use GUI based network tool



# DHCP



- Install dhcp server package
  - `yum install dhcp` (version 7)
  - `dnf install dhcp-server` (version 8)
- Edit the configuration file with desired parameters
  - `vi /etc/dhcp/dhcp.conf`
  - `cp /usr/share/doc/dhcp-x.x.x/dhcpd.conf.example /etc/dhcp/dhcpd.conf`

```
#
# DHCP Server Configuration file.
# see /usr/share/doc/dhcp-server/dhcpd.conf.example
# see dhcpd.conf(5) man page
#
default-lease-time 600;
max-lease-time 7200;

ddns-update-style none;
authoritative;

subnet 192.168.15.0 netmask 255.255.255.0 {
    range 192.168.15.50 192.168.15.200;
    option routers 192.168.15.1;
    option subnet-mask 255.255.255.0;
    option domain-name-servers 8.8.8.8, 8.8.4.4;
}
~
~
~
```

- The DHCP server will reserve the IP address for at least 10 minutes
- The DHCP server will reserve the IP address for a max of 2 hours
- Defines the subnet range of 256 addresses
- Defines the DHCP range assignment of 150 addresses
- Router defines the default gateway
- Defines the default subnet mask that will be assigned to each host
- Defines the DNS nameservers which will be assigned to each host.

# DHCP

- Start dhcpd service
  - `systemctl start dhcpd`
  - `systemctl enable dhcp`
- Disable `firewalld` or allow dhcp port over firewall
  - `systemctl stop firewalld`
  - OR
  - `firewall-cmd --add-service=dhcp --permanent`
  - `firewall-cmd --reload`
- Switch DHCP service from your router/modem to your new DHCP server
  - Login to your ISP provided router
  - Disable dhcp and enable forwarding to the new dhcp server.

