How to install and configure CentOs 7, Hadoop with Spark and Yarn. The definitive guide.

This guide will walk you trough the following steps, the objective is to set up a fully functional CentOS 7 server for multiple users, with Hadoop HDFS file system, yarn resource manager and spark engine for data-processing on top; Some steps will be optional and some will be mandatory to assure system functionality. At the beginning of each chapter we will notify you if there are required steps before you proceed, to ensure that everything works fine.

Chapter List:

IMPORTANT DISCLAMER: Before we begin we feel it's important to remind you that Hadoop, yarn and spark are still in development and filled with nasty bugs, they receive updates frequently that could remove, add and/or make some settings obsolete, it's always important to rely on the latest info you find on the web and is not always true that an old workaround will solve the same problem for different version of the same product, this guide was made with the latest available version of all software (when possible) but could become obsolete very quickly, consider yourself warned. Last thing before we begin: this guide is made by two students, we don't consider ourselves expert in the field, although we noticed the lack of an all-around comprehensive guide on the matter, something we write here could be completely wrong, verify for yourself the correctness of everything you do with your system.

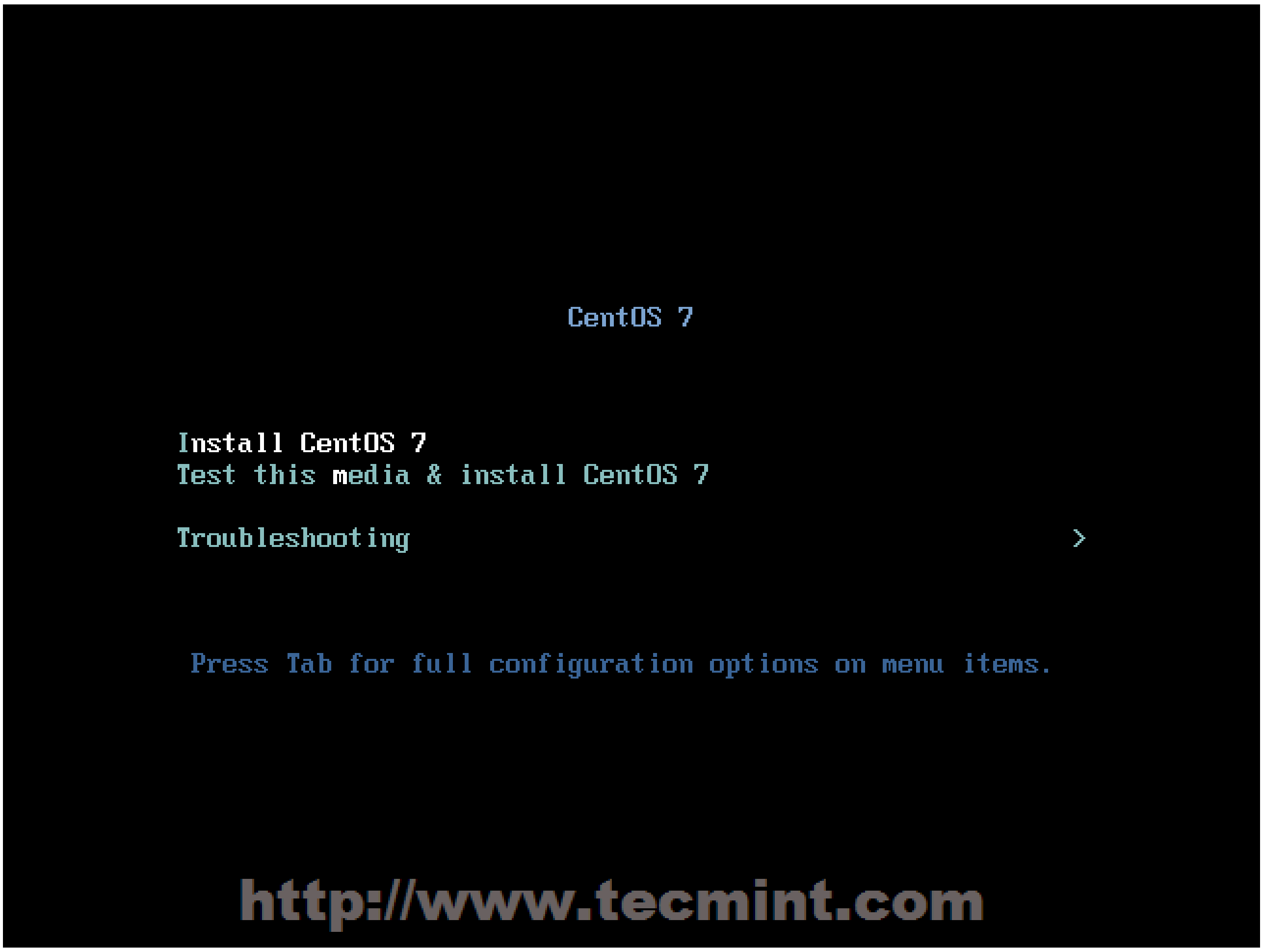
1. Installing CentOS 7 server

Required Steps: none.

This chapter will guide you on how to perform a minimal installation of latest version of CentOS 7.0, using the binary DVD ISO image, an installation that is best suitable for developing a future customizable server platform, with no Graphical User Interface, where you can install only the software that you need.

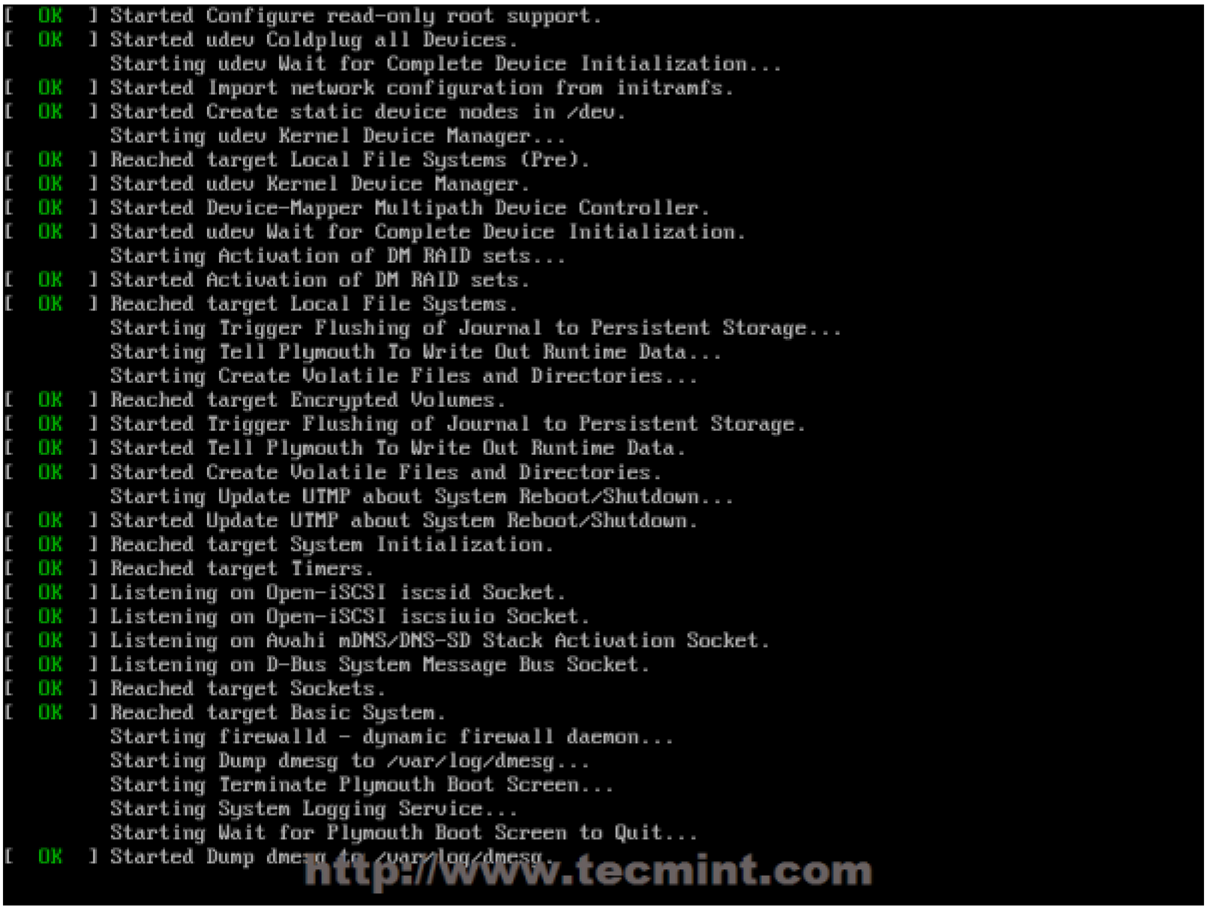
1. After downloading the last version of CentOS using above links or using official CentOS download page. Burn it to a DVD or create a bootable USB stick using LiveUSB Creator called Unetbootin.

2. After you have created the installer bootable media, place your DVD/USB into your system appropriate drive, start the computer, change your bios boot priority media if it is not already set, select your bootable unit and the first CentOS 7 prompt should appear. At the prompt choose Install CentOS 7 and press [Enter] key. Or if you're paranoid like us, choose the second option "Test this media & Install CentOS 7" to first test the drive you created.

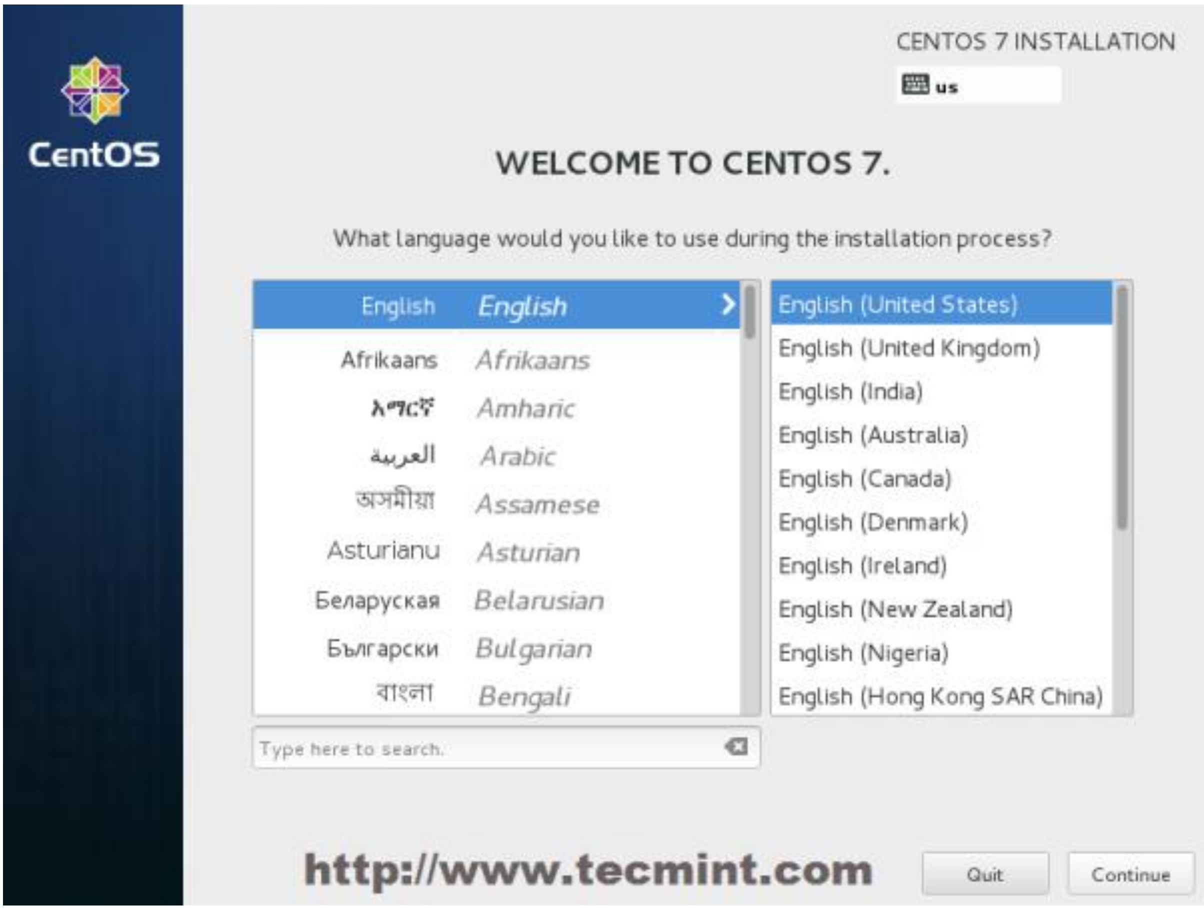


```
CentOS 7

Install CentOS 7
Test this media & install CentOS 7

Troubleshooting                                    >

Press Tab for full configuration options on menu items.
```

3. The system will start loading media installer and a Welcome screen should appear. Select your Installation Process Language, that will assist you through the entire installation procedure and click on Continue.



```
[  OK  ] Started Configure read-only root support.
[  OK  ] Started udev Coldplug all Devices.
         Starting udev Wait for Complete Device Initialization...
[  OK  ] Started Import network configuration from initramfs.
[  OK  ] Started Create static device nodes in /dev.
         Starting udev Kernel Device Manager...
[  OK  ] Reached target Local File Systems (Pre).
[  OK  ] Started udev Kernel Device Manager.
[  OK  ] Started Device-Mapper Multipath Device Controller.
[  OK  ] Started udev Wait for Complete Device Initialization.
         Starting Activation of DM RAID sets...
[  OK  ] Started Activation of DM RAID sets.
[  OK  ] Reached target Local File Systems.
         Starting Trigger Flushing of Journal to Persistent Storage...
         Starting Tell Plymouth To Write Out Runtime Data...
         Starting Create Volatile Files and Directories...
[  OK  ] Reached target Encrypted Volumes.
[  OK  ] Started Trigger Flushing of Journal to Persistent Storage.
[  OK  ] Started Tell Plymouth To Write Out Runtime Data.
[  OK  ] Started Create Volatile Files and Directories.
         Starting Update UTMP about System Reboot/Shutdown...
[  OK  ] Started Update UTMP about System Reboot/Shutdown.
[  OK  ] Reached target System Initialization.
[  OK  ] Reached target Timers.
[  OK  ] Listening on Open-iSCSI iscsid Socket.
[  OK  ] Listening on Open-iSCSI iscsiuio Socket.
[  OK  ] Listening on Avahi mDNS/DNS-SD Stack Activation Socket.
[  OK  ] Listening on D-Bus System Message Bus Socket.
[  OK  ] Reached target Sockets.
[  OK  ] Reached target Basic System.
         Starting firewalld - dynamic firewall daemon...
         Starting Dump dmesg to /var/log/dmesg...
         Starting Terminate Plymouth Boot Screen...
         Starting System Logging Service...
         Starting Wait for Plymouth Boot Screen to Quit...
[  OK  ] Started Dump dmesg to /var/log/dmesg.
```

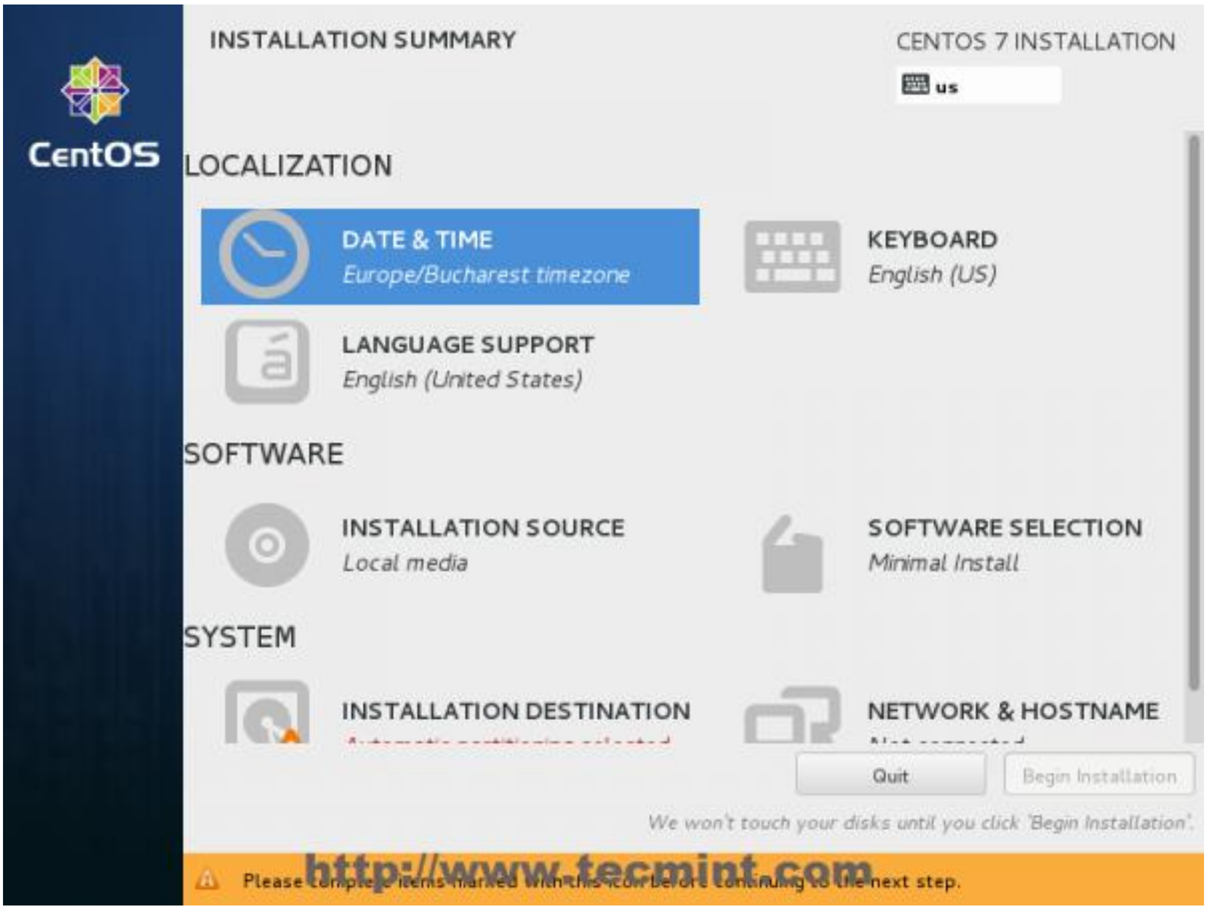


CENTOS 7 INSTALLATION

WELCOME TO CENTOS 7.

What language would you like to use during the installation process?

4. The next step, present screen prompt is Installation Summary. It contains a lot of options to fully customize your system. First thing you may want to setup is your time settings. Click on Date & Time and select your server physical location from the provided map and hit on upper Done button to apply configuration.



INSTALLATION SUMMARY

CENTOS 7 INSTALLATION

us

LOCALIZATION

DATE & TIME
Europe/Bucharest timezone

KEYBOARD
English (US)

LANGUAGE SUPPORT
English (United States)

SOFTWARE

INSTALLATION SOURCE
Local media

SOFTWARE SELECTION
Minimal Install

SYSTEM

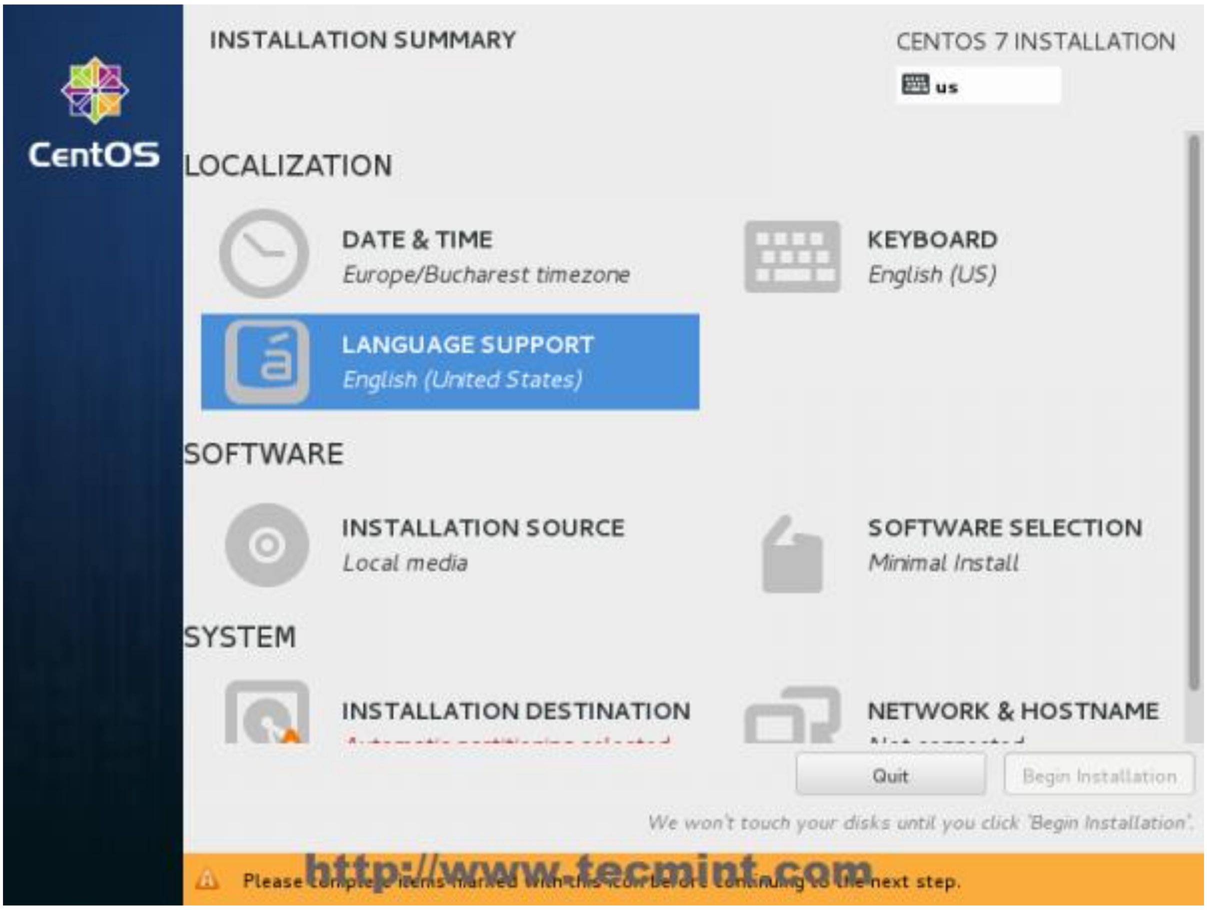
INSTALLATION DESTINATION

NETWORK & HOSTNAME

We won't touch your disks until you click 'Begin Installation'.

Please complete items marked with this icon before continuing to the next step.

5. The next step is to choose your Language Support and Keyboard settings. Choose your main and extra language for your system and when you're finished hit on Done button.



LOCALIZATION

DATE & TIME
Europe/Bucharest timezone

KEYBOARD
English (US)

LANGUAGE SUPPORT
English (United States)

SOFTWARE

INSTALLATION SOURCE
Local media

SOFTWARE SELECTION
Minimal Install

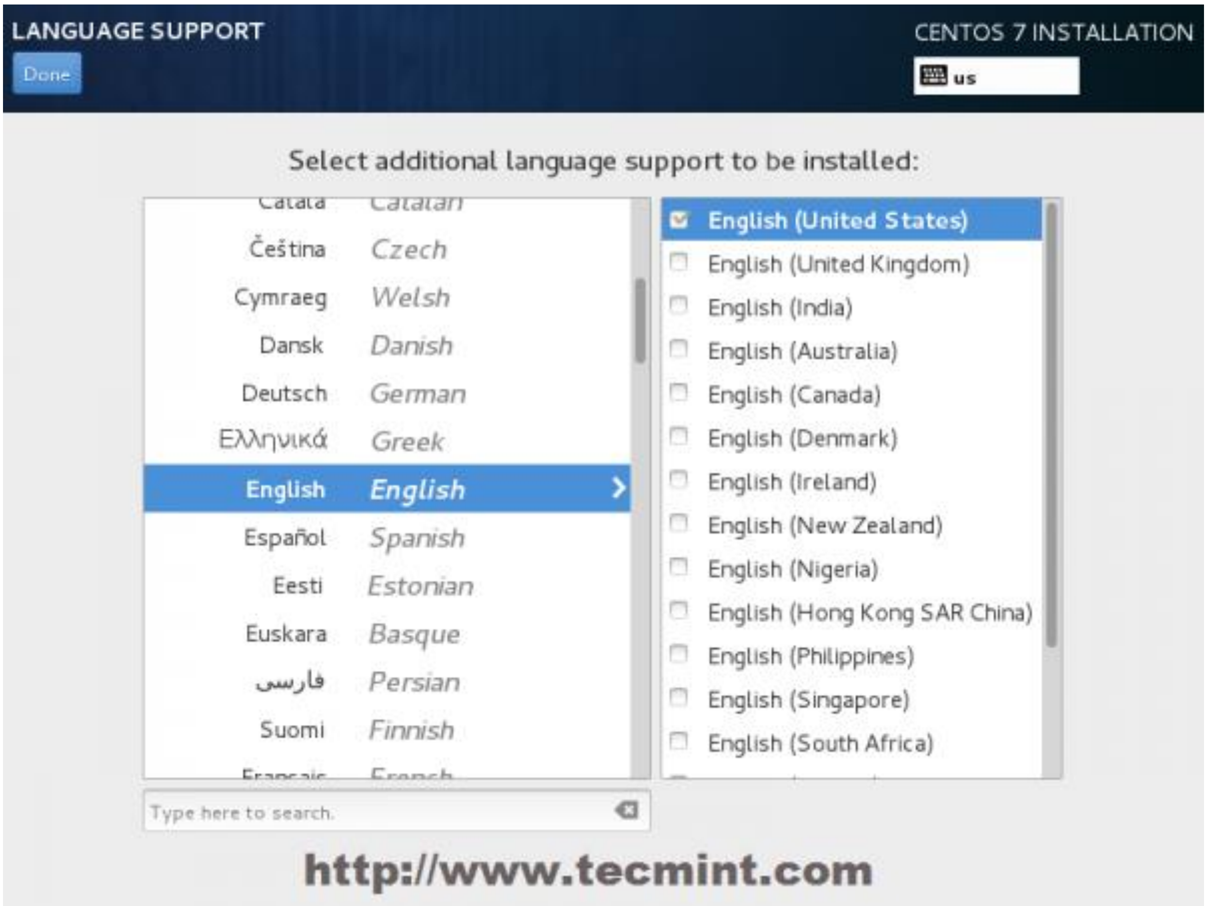
SYSTEM

INSTALLATION DESTINATION
Automatic partitioning selected

NETWORK & HOSTNAME
Not connected

We won't touch your disks until you click 'Begin Installation'.

⚠ Please complete items marked with this icon before continuing to the next step.

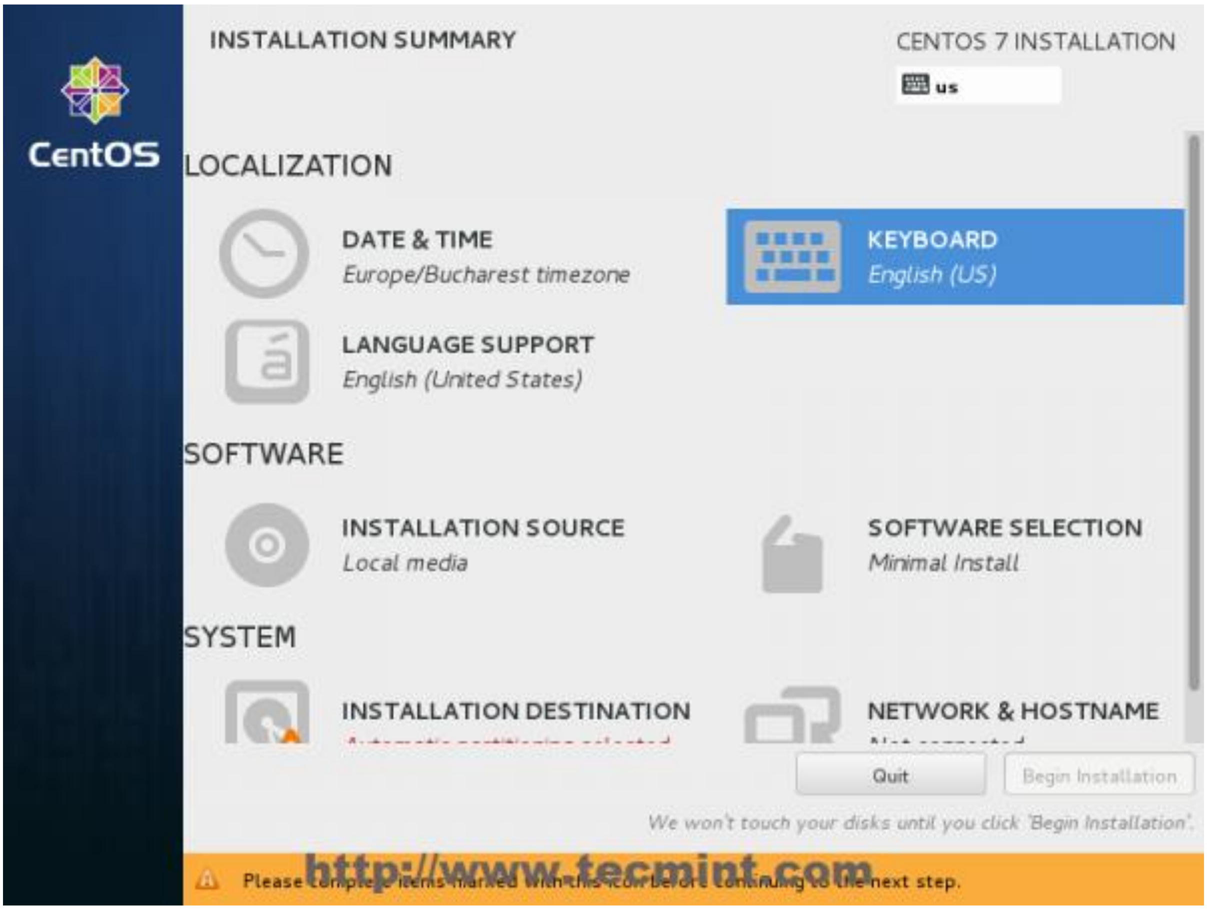


Select additional language support to be installed:

Català	*Catalan*
Čeština	*Czech*
Cymraeg	*Welsh*
Dansk	*Danish*
Deutsch	*German*
Ελληνικά	*Greek*
English	*English*
Español	*Spanish*
Eesti	*Estonian*
Euskara	*Basque*
فارسی	*Persian*
Suomi	*Finnish*
Français	*French*

- ☑ English (United States)
- □ English (United Kingdom)
- □ English (India)
- □ English (Australia)
- □ English (Canada)
- □ English (Denmark)
- □ English (Ireland)
- □ English (New Zealand)
- □ English (Nigeria)
- □ English (Hong Kong SAR China)
- □ English (Philippines)
- □ English (Singapore)
- □ English (South Africa)

6. In the same way choose your Keyboard Layout by hitting the plus button and test your keyboard configuration using the right input filed. After you finish setting up your keyboard, again hit on upper Done button to apply changes and go back to main screen on Installation Summary.



INSTALLATION SUMMARY

CENTOS 7 INSTALLATION

us

LOCALIZATION

DATE & TIME
Europe/Bucharest timezone

KEYBOARD
English (US)

LANGUAGE SUPPORT
English (United States)

SOFTWARE

INSTALLATION SOURCE
Local media

SOFTWARE SELECTION
Minimal Install

SYSTEM

INSTALLATION DESTINATION

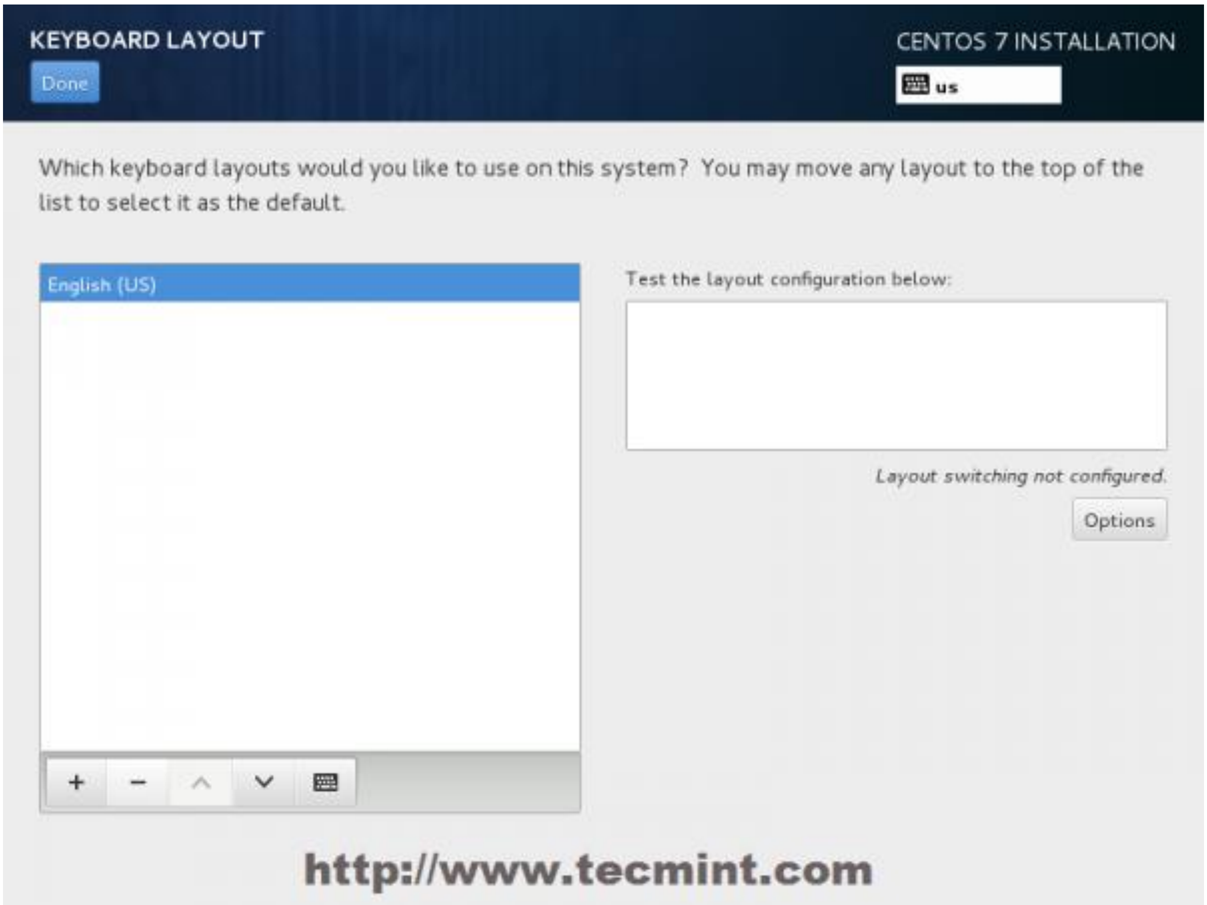
NETWORK & HOSTNAME

Quit | Begin Installation

We won't touch your disks until you click 'Begin Installation'.

⚠ Please complete items marked with this icon before continuing to the next step.

http://www.tecmint.com



KEYBOARD LAYOUT

Done

CENTOS 7 INSTALLATION

us

Which keyboard layouts would you like to use on this system? You may move any layout to the top of the list to select it as the default.

English (US)

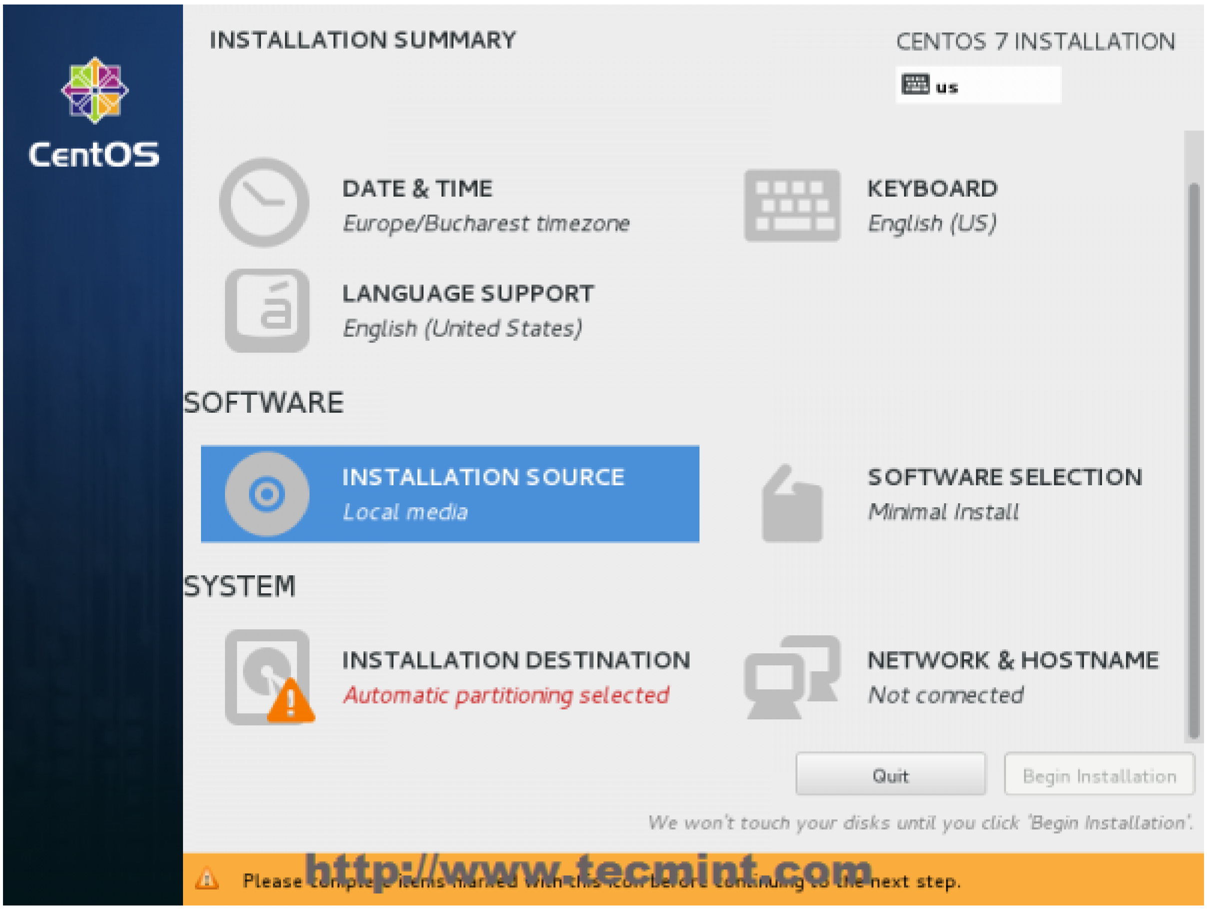
Test the layout configuration below:

Layout switching not configured.

Options

http://www.tecmint.com

7. On the next step you can customize your installation by using other Installation Sources than your local DVD/USB media, such as a network locations using HTTP, HTTPS, FTP or NFS protocols and even add some additional repositories, but use this methods only if you know what you're doing. So leave the default Auto-detected installation media and hit on Done to continue.



INSTALLATION SUMMARY

CENTOS 7 INSTALLATION

us

DATE & TIME
Europe/Bucharest timezone

KEYBOARD
English (US)

LANGUAGE SUPPORT
English (United States)

SOFTWARE

INSTALLATION SOURCE
Local media

SOFTWARE SELECTION
Minimal Install

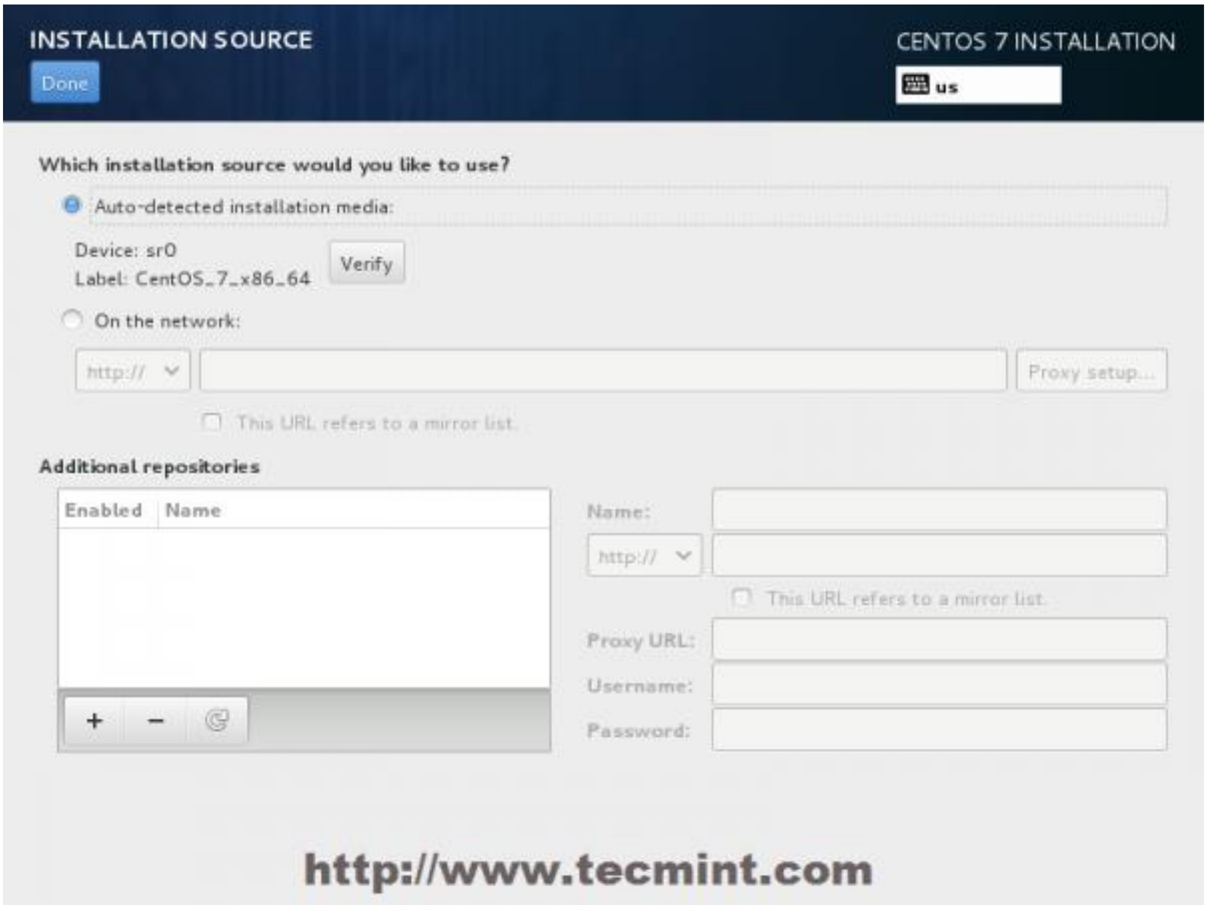
SYSTEM

INSTALLATION DESTINATION
Automatic partitioning selected

NETWORK & HOSTNAME
Not connected

We won't touch your disks until you click 'Begin Installation'.

Please complete items marked with this icon before continuing to the next step.



INSTALLATION SOURCE

CENTOS 7 INSTALLATION

us

Which installation source would you like to use?

● Auto-detected installation media:

Device: sr0
Label: CentOS_7_x86_64

○ On the network:

http://

□ This URL refers to a mirror list.

Additional repositories

Enabled	Name

Name:

http://

□ This URL refers to a mirror list.

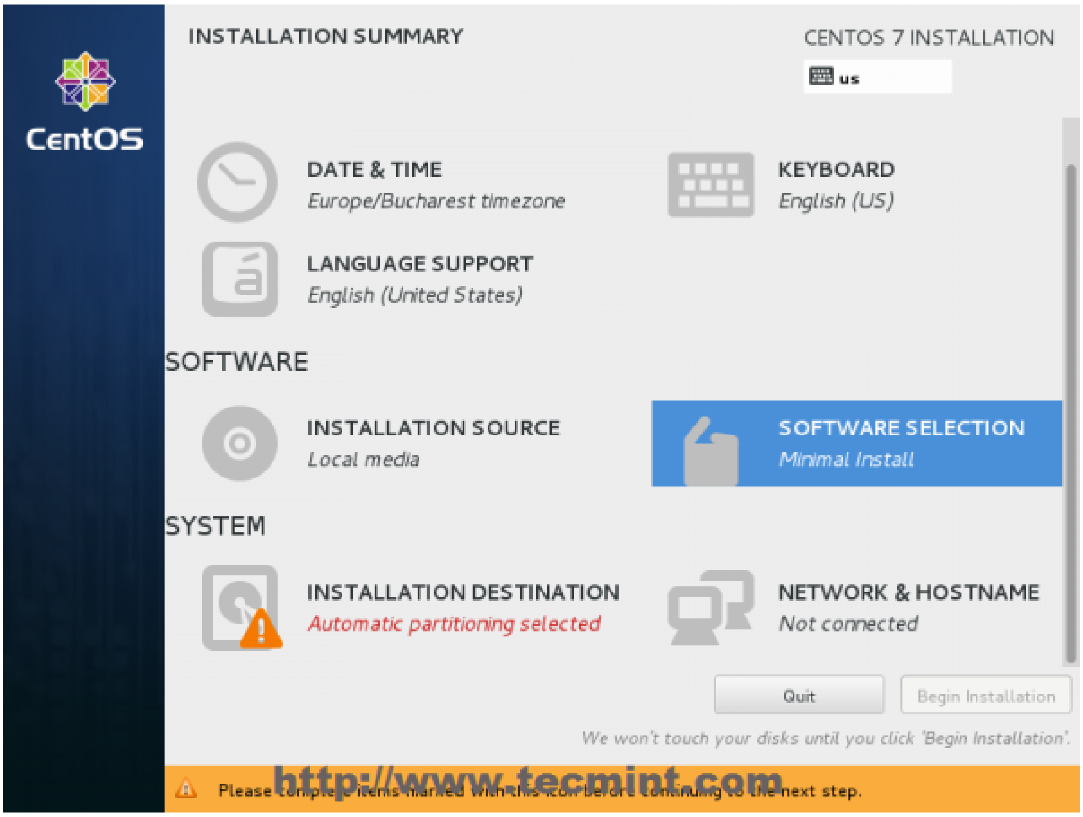
Proxy URL:

Username:

Password:

8. On the next step you can choose your system installation software. On this step CentOS offers a lot of Server and Desktop platform environments that you choose from, but, if you want a high degree of customization, especially if you are going to use CentOS 7 to run as a server platform, then I suggest you select Minimal Install with Compatibility Libraries as Add-ons, which will install a minimal basic system software and later you can add other packages as your needs require using yum groupinstall command.

NOTE: We choose to install a gui at first and then disable it late to free up resources, you will find how to re-enable it if necessary later in this guide.



INSTALLATION SUMMARY

CENTOS 7 INSTALLATION

us

DATE & TIME
Europe/Bucharest timezone

KEYBOARD
English (US)

LANGUAGE SUPPORT
English (United States)

SOFTWARE

INSTALLATION SOURCE
Local media

SOFTWARE SELECTION
Minimal Install

SYSTEM

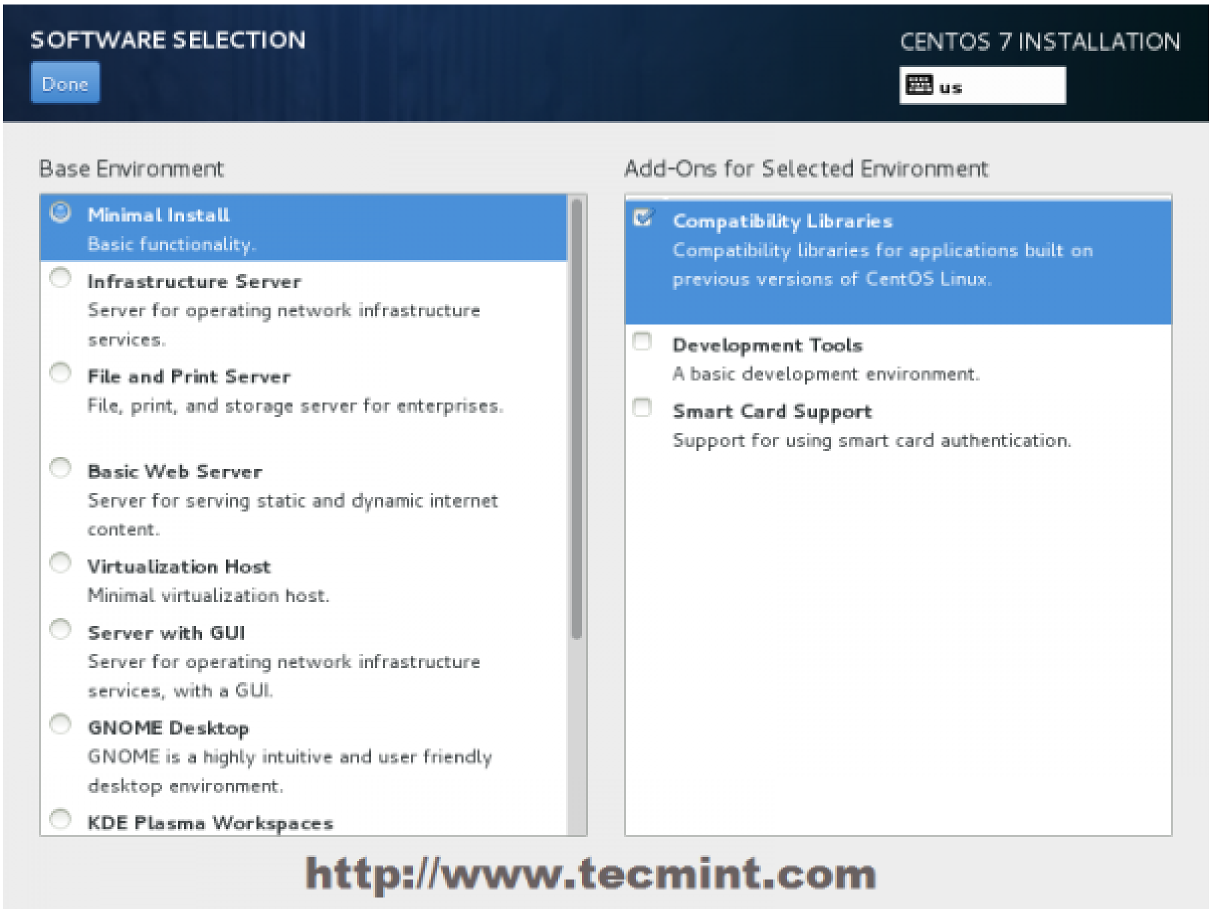
INSTALLATION DESTINATION
Automatic partitioning selected

NETWORK & HOSTNAME
Not connected

Quit
Begin Installation

We won't touch your disks until you click 'Begin Installation'.

Please complete items marked with this icon before continuing to the next step.



SOFTWARE SELECTION

Done

CENTOS 7 INSTALLATION

us

Base Environment

● Minimal Install
Basic functionality.

○ Infrastructure Server
Server for operating network infrastructure services.

○ File and Print Server
File, print, and storage server for enterprises.

○ Basic Web Server
Server for serving static and dynamic internet content.

○ Virtualization Host
Minimal virtualization host.

○ Server with GUI
Server for operating network infrastructure services, with a GUI.

○ GNOME Desktop
GNOME is a highly intuitive and user friendly desktop environment.

○ KDE Plasma Workspaces

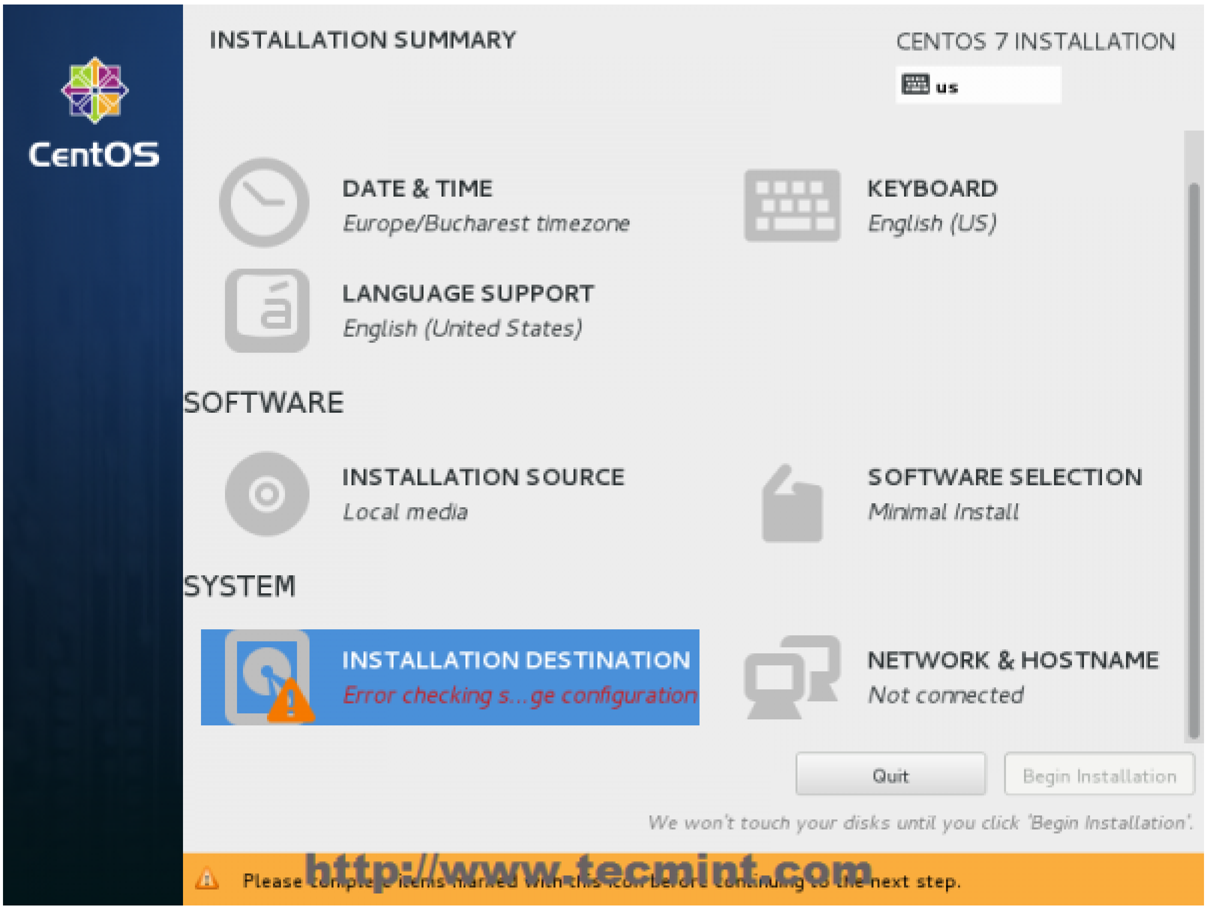
Add-Ons for Selected Environment

☑ Compatibility Libraries
Compatibility libraries for applications built on previous versions of CentOS Linux.

□ Development Tools
A basic development environment.

□ Smart Card Support
Support for using smart card authentication.

9. Now it's time to partition your hard-drive. Click on Installation Destination menu, select your disk and choose I will configure partitioning.



INSTALLATION SUMMARY

CENTOS 7 INSTALLATION

us

CentOS

DATE & TIME
Europe/Bucharest timezone

KEYBOARD
English (US)

LANGUAGE SUPPORT
English (United States)

SOFTWARE

INSTALLATION SOURCE
Local media

SOFTWARE SELECTION
Minimal Install

SYSTEM

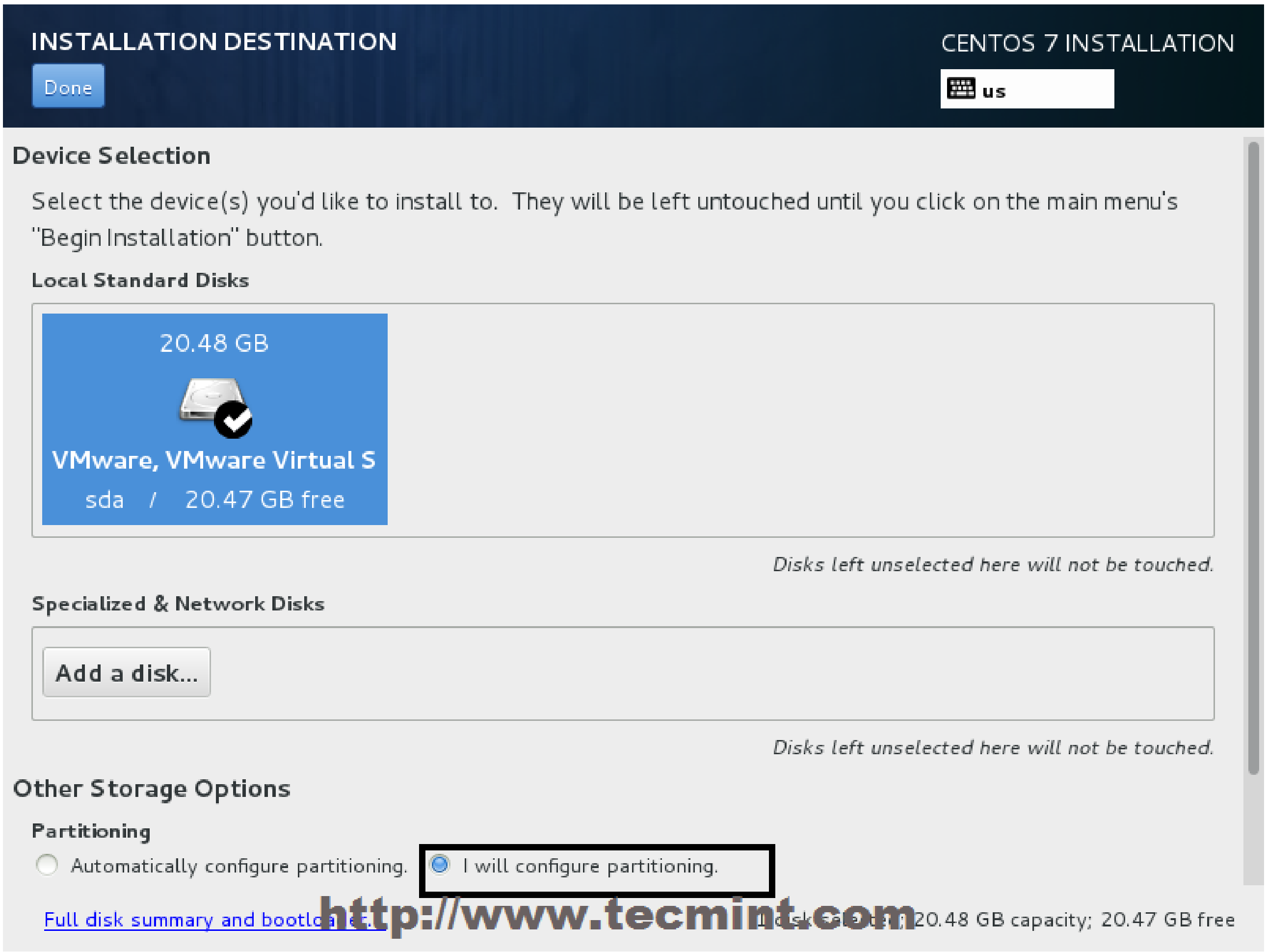
INSTALLATION DESTINATION
Error checking s...ge configuration

NETWORK & HOSTNAME
Not connected

Quit | Begin Installation

We won't touch your disks until you click 'Begin Installation'.

Please complete items marked with this icon before continuing to the next step.



INSTALLATION DESTINATION

Done

CENTOS 7 INSTALLATION

us

Device Selection

Select the device(s) you'd like to install to. They will be left untouched until you click on the main menu's "Begin Installation" button.

Local Standard Disks

20.48 GB
VMware, VMware Virtual S
sda / 20.47 GB free

Disks left unselected here will not be touched.

Specialized & Network Disks

Add a disk...

Disks left unselected here will not be touched.

Other Storage Options

Partitioning

○ Automatically configure partitioning. ● I will configure partitioning.

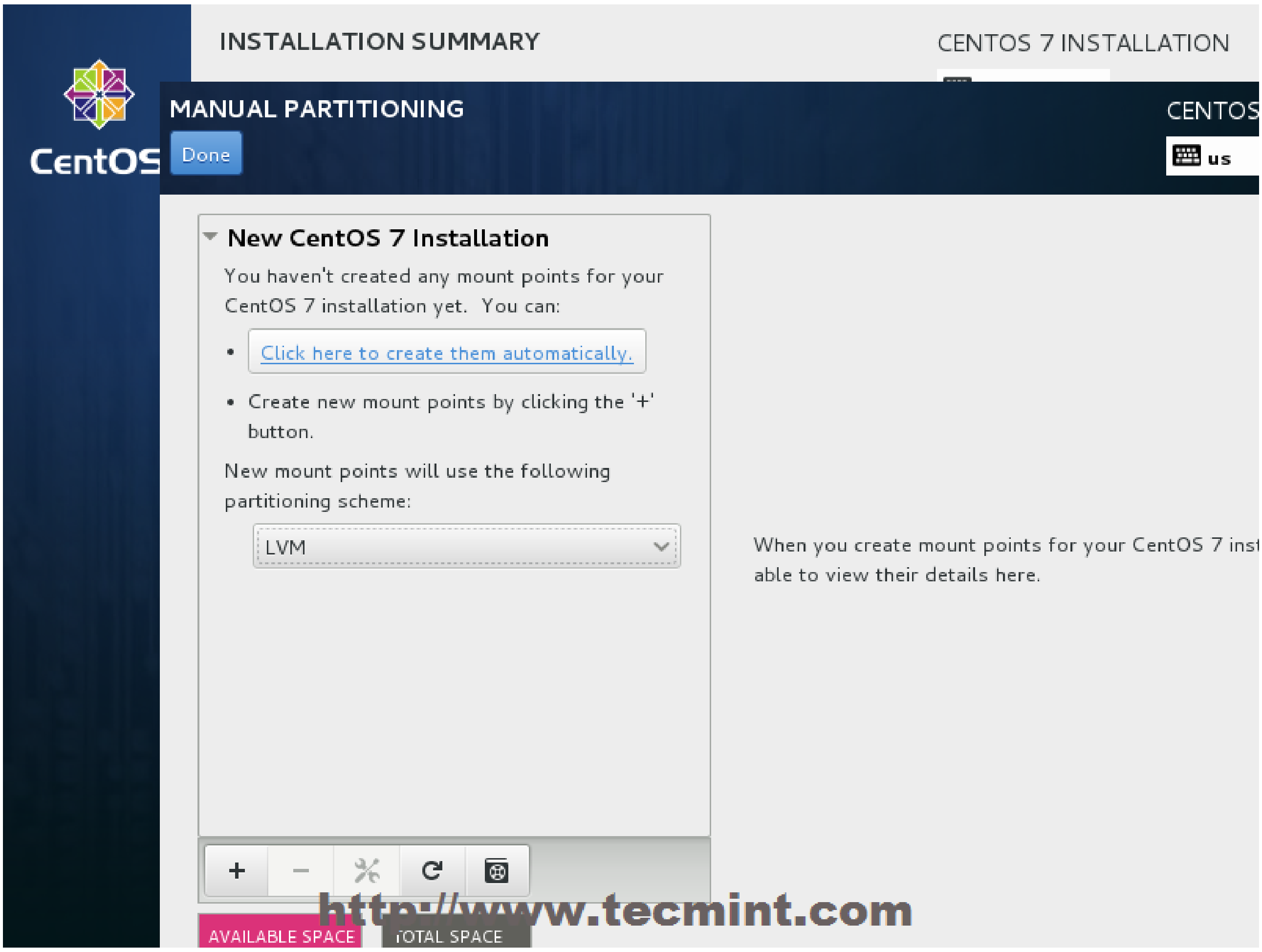
Full disk summary and bootloader... 1 disk selected; 20.48 GB capacity; 20.47 GB free

10. On the next screen, choose LVM (Logical Volume Manager) as partition layout and, then, click on Click here to create them automatically, option which will create three system partition using XFS filesystem, automatically redistributing your hard-disk space and gathering all LVS into one big Volume Group named centos. If you are not pleased with the default partition layout done automatically by the installer you can completely add, modify or resize your partition scheme. We are working with a 1 TB drive, so we set the dimension like this and left 300 gb for future upgrade in case we'll need one.

- /(home) – 300gb
- /boot – Non LVM 1gb
- /(root) – LVM 300gb
- Swap – LVM 10gb (8gb of ram + 2 extra following official Documentation)

NOTE: For those users, who have hard-disks more than 2TB in size, the installer automatically will convert partition table to GPT, but if you wish to use GPT table on smaller disks than 2TB, then you should use the argument inst.gpt to the installer boot command line in order to change the default behaviour.

Wait! You don't know what an LVM Partition is? Don't worry, you can find out more here (and a little bit later on this guide)!



INSTALLATION SUMMARY

CENTOS 7 INSTALLATION

MANUAL PARTITIONING

CENTOS

New CentOS 7 Installation

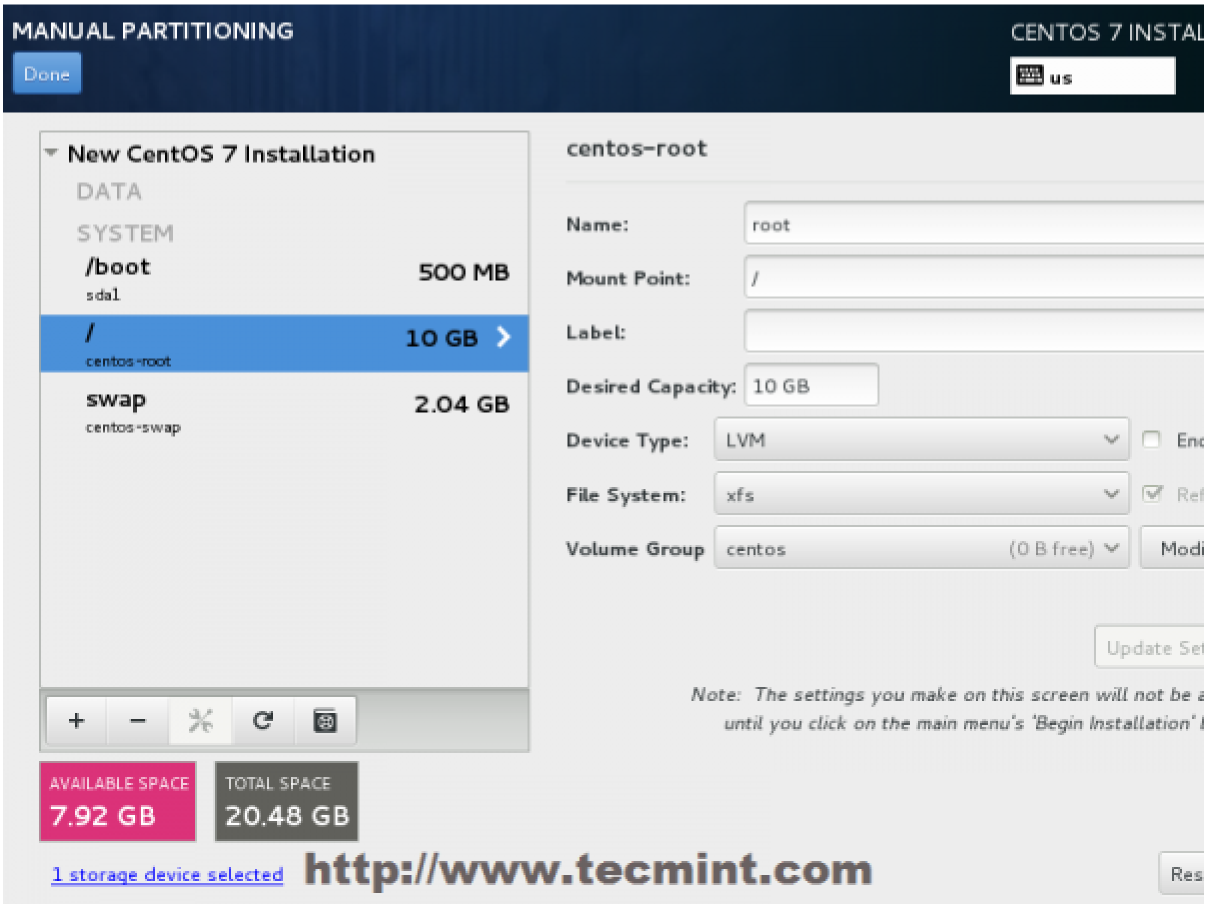
You haven't created any mount points for your CentOS 7 installation yet. You can:

- Click here to create them automatically.
- Create new mount points by clicking the '+' button.

New mount points will use the following partitioning scheme:

LVM

When you create mount points for your CentOS 7 inst able to view their details here.



Name: root

Mount Point: /

Label:

Desired Capacity: 10 GB

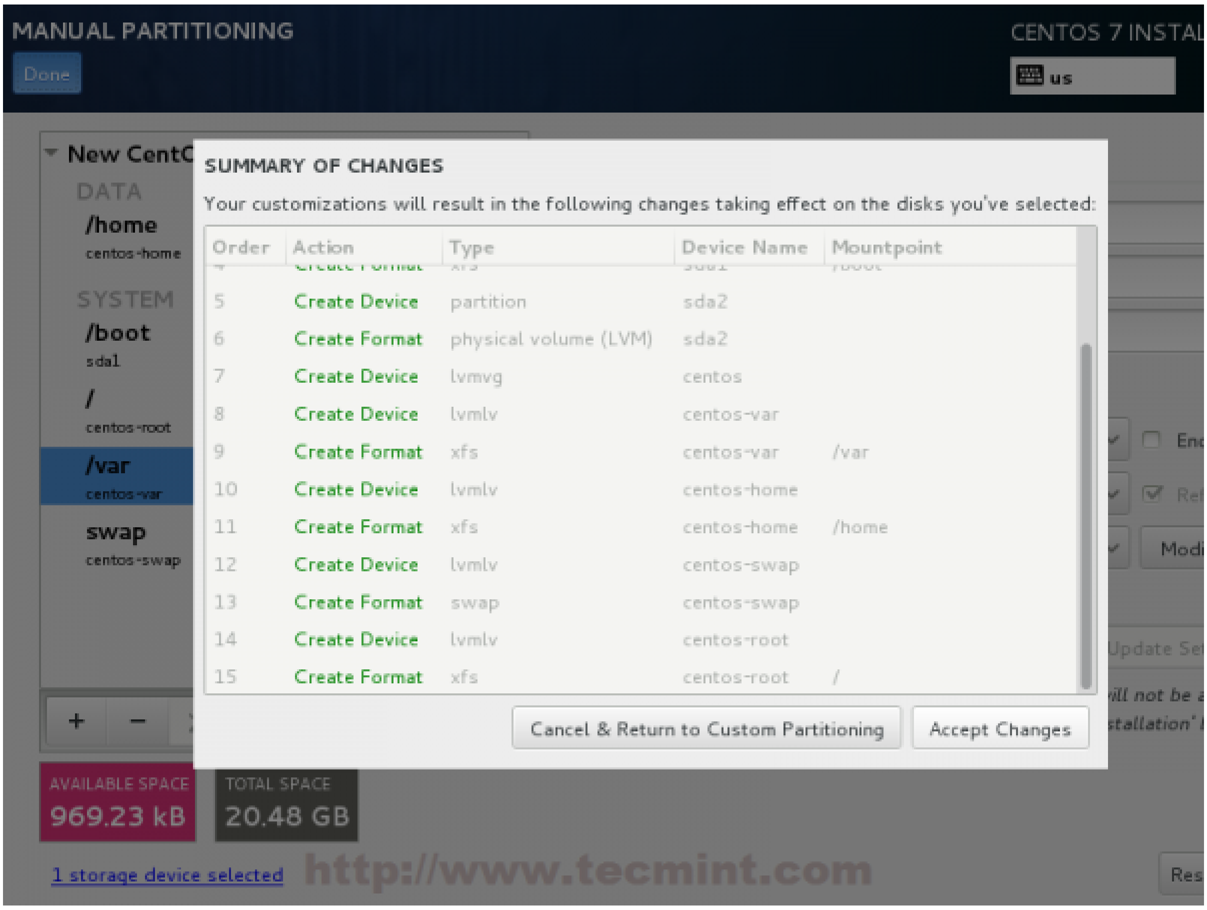
Device Type: LVM

File System: xfs

Volume Group centos (0 B free)

Note: The settings you make on this screen will not be a
until you click on the main menu's 'Begin Installation' I

When you finish hit on Done button and Accept Changes on the Summary of Changes prompt.



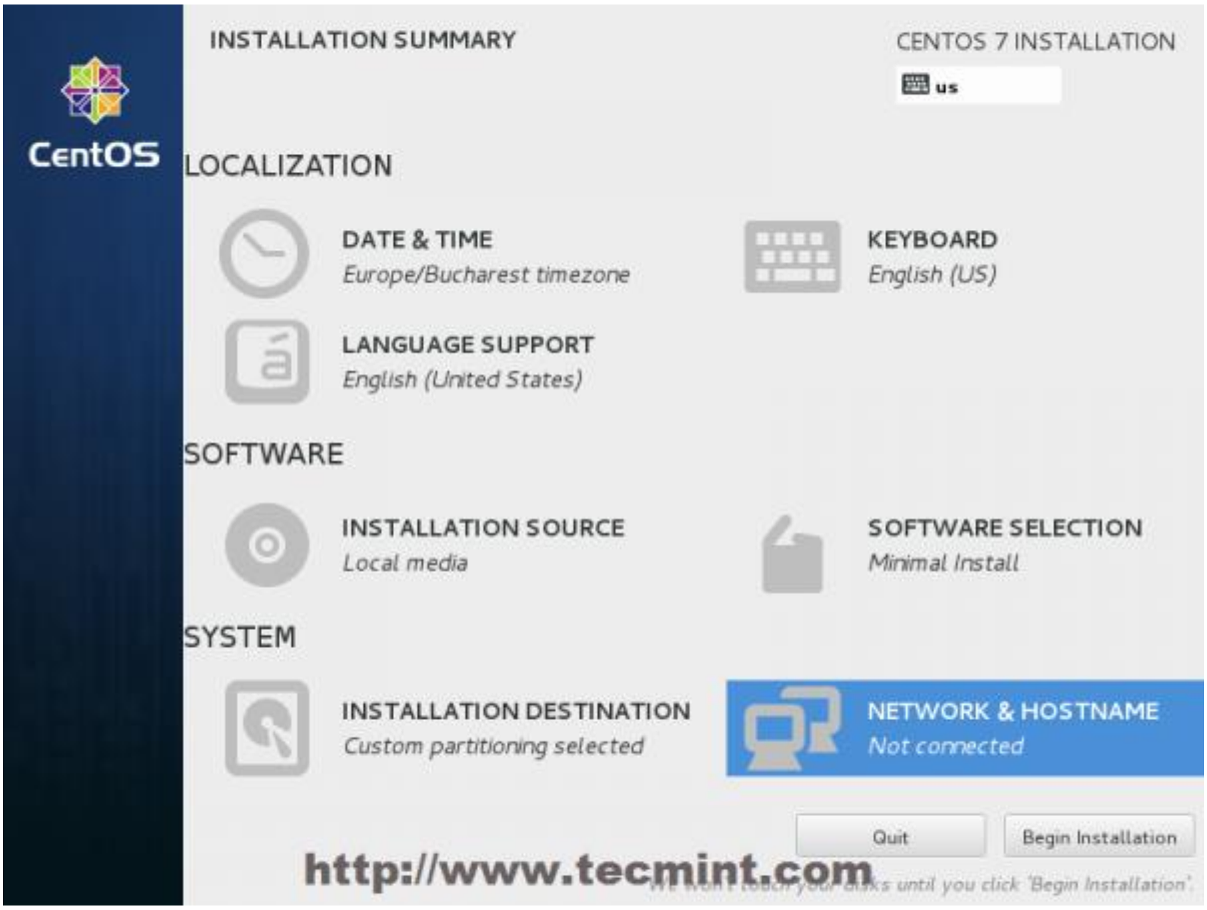
SUMMARY OF CHANGES

Your customizations will result in the following changes taking effect on the disks you've selected:

Order	Action	Type	Device Name	Mountpoint
4	Create Format	xfs	sda1	/boot
5	Create Device	partition	sda2	
6	Create Format	physical volume (LVM)	sda2	
7	Create Device	lvmvg	centos	
8	Create Device	lvmlv	centos-var	
9	Create Format	xfs	centos-var	/var
10	Create Device	lvmlv	centos-home	
11	Create Format	xfs	centos-home	/home
12	Create Device	lvmlv	centos-swap	
13	Create Format	swap	centos-swap	
14	Create Device	lvmlv	centos-root	
15	Create Format	xfs	centos-root	/

11. The next step is to set your system hostname and enable networking. Click on Network & Hostname label and type your system FQDN (Fully Qualified Domain Name) on Hostname filed, then enable your Network interface, switching the top Ethernet button to ON. If you have a functional DHCP server on you network then it will automatically configure all your network setting for enabled NIC, which should appear under your active interface.

NOTE: For our setup we choose master and slave respectively as hostnames.



INSTALLATION SUMMARY

CENTOS 7 INSTALLATION

LOCALIZATION

DATE & TIME
Europe/Bucharest timezone

KEYBOARD
English (US)

LANGUAGE SUPPORT
English (United States)

SOFTWARE

INSTALLATION SOURCE
Local media

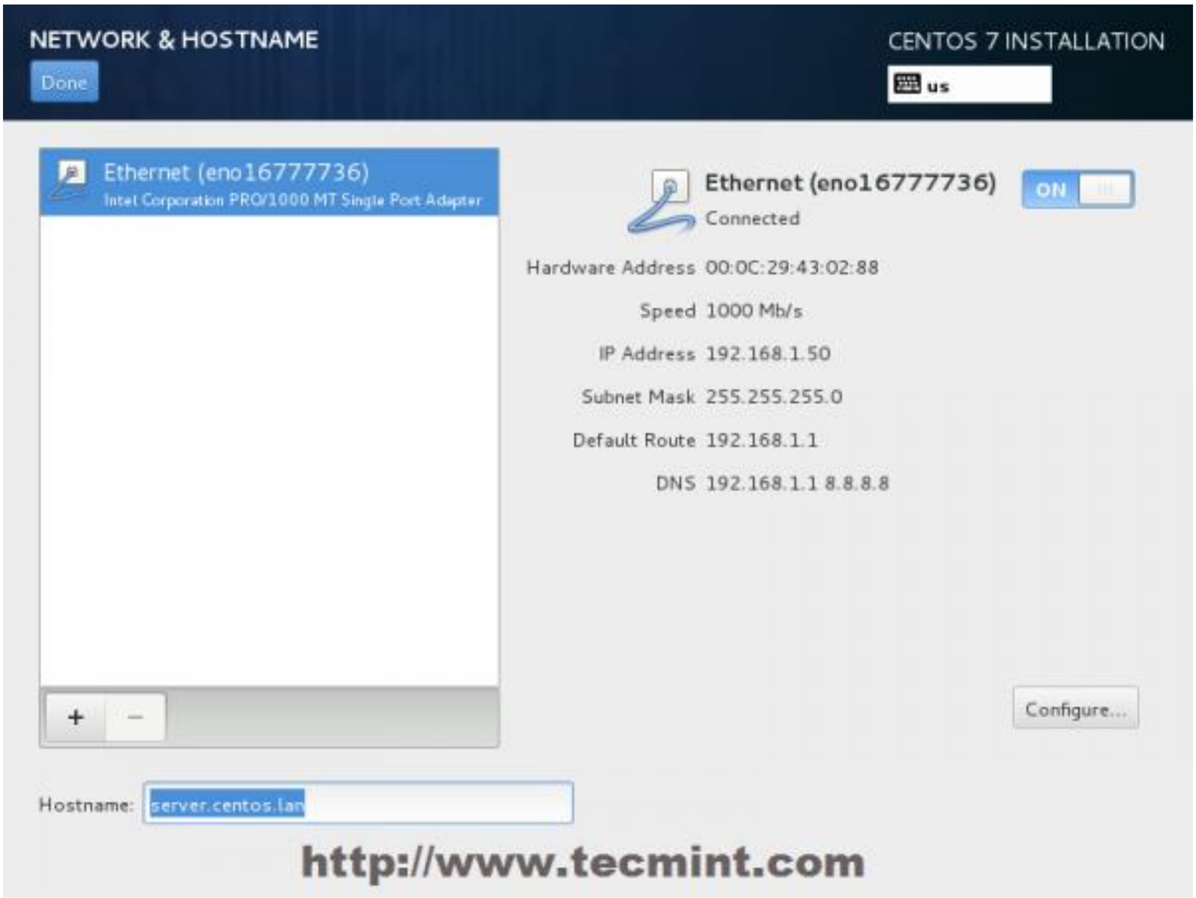
SOFTWARE SELECTION
Minimal Install

SYSTEM

INSTALLATION DESTINATION
Custom partitioning selected

NETWORK & HOSTNAME
Not connected

We won't touch your disks until you click 'Begin Installation'.



NETWORK & HOSTNAME

CENTOS 7 INSTALLATION

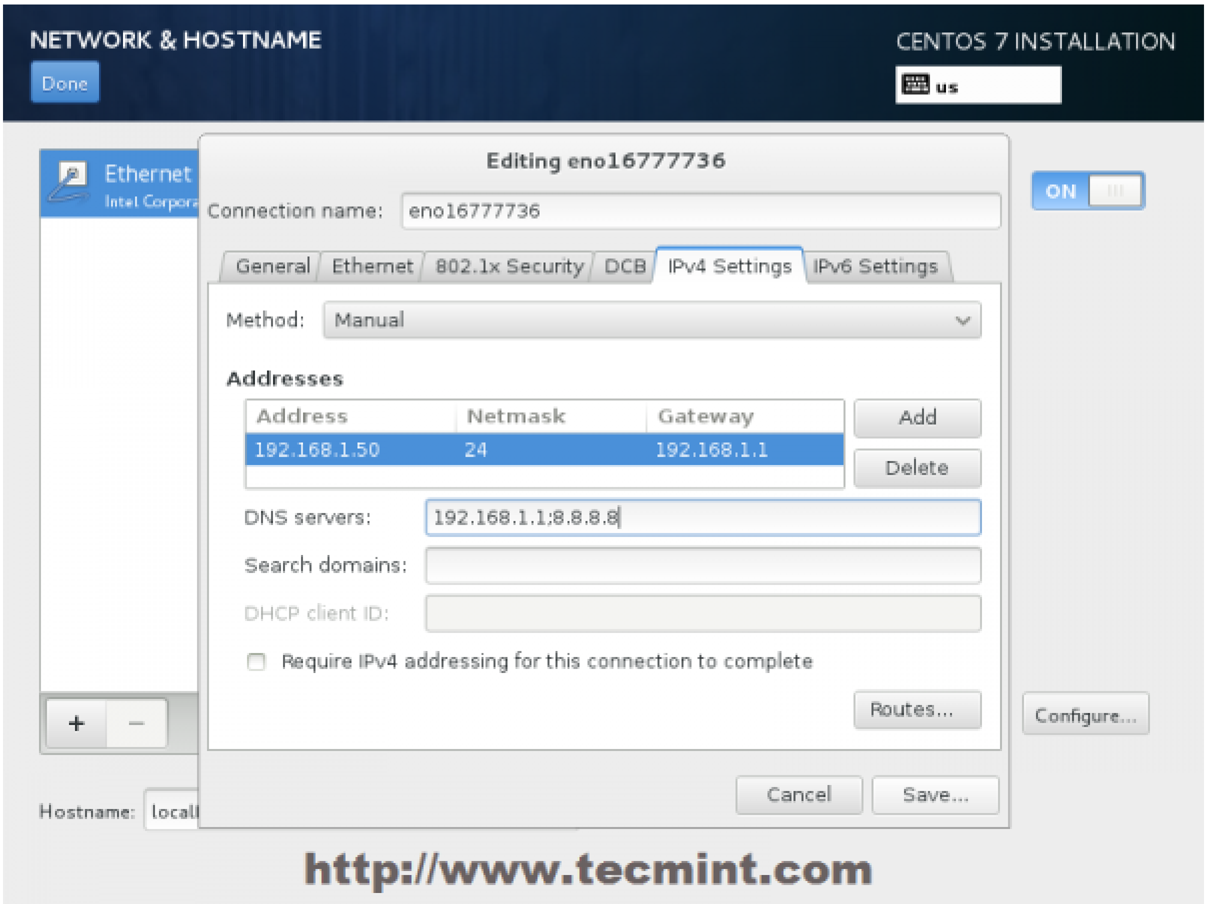
Ethernet (eno16777736)
Intel Corporation PRO/1000 MT Single Port Adapter

Ethernet (eno16777736)
Connected

Hardware Address 00:0C:29:43:02:88
Speed 1000 Mb/s
IP Address 192.168.1.50
Subnet Mask 255.255.255.0
Default Route 192.168.1.1
DNS 192.168.1.1 8.8.8.8

Hostname: server.centos.lan

12. If your system will be destined as a server it's better to set static network configuration on Ethernet NIC by clicking on Configure button and add all your static interface settings like in the screenshot below, and when you're finished hit on Save button, disable and enable Ethernet card by switching the button to OFF and ON, and, then hit on Done to apply setting and go back to main menu.


NETWORK & HOSTNAME

Editing eno16777736

Connection name: eno16777736

Method: Manual

Addresses

Address	Netmask	Gateway
192.168.1.50	24	192.168.1.1

DNS servers: 192.168.1.1;8.8.8.8

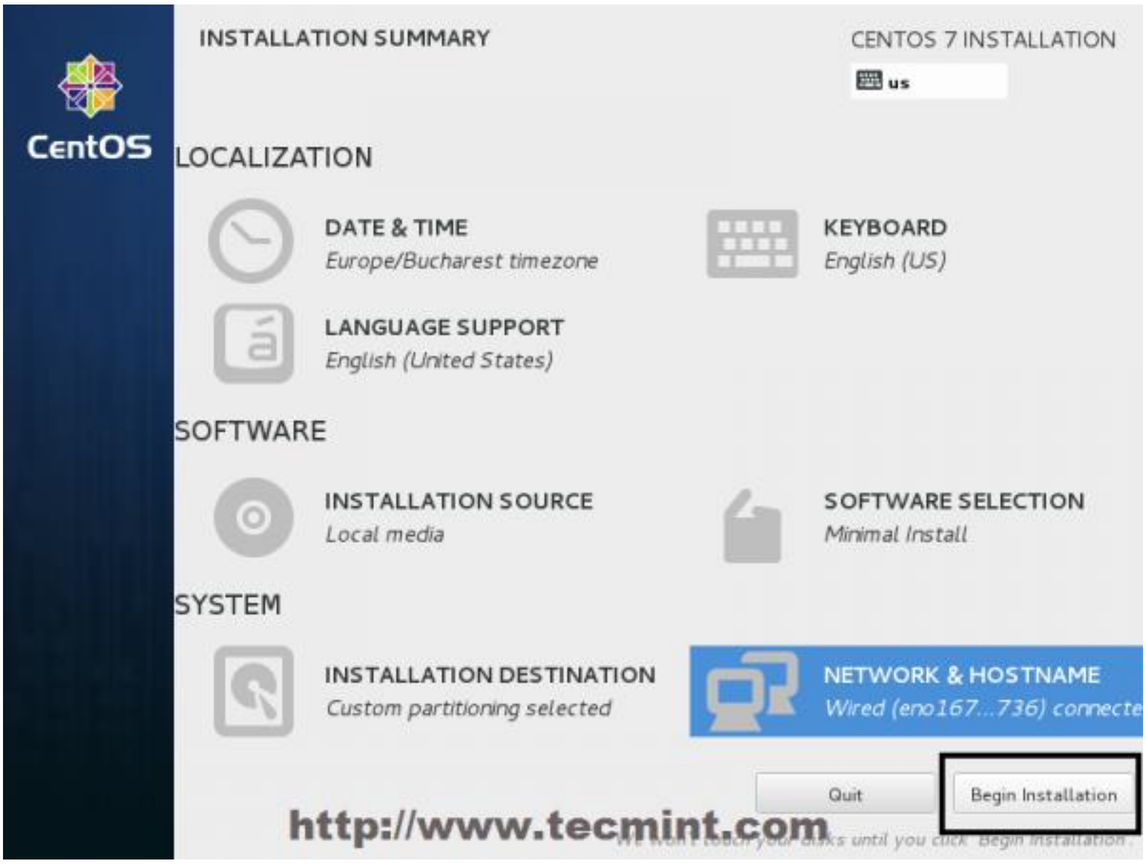
Search domains:

DHCP client ID:

□ Require IPv4 addressing for this connection to complete

Hostname: locall

13. Now it's time to start installation process by pressing on Begin Installation button and set up a strong password for root account.


INSTALLATION SUMMARY

LOCALIZATION

DATE & TIME
Europe/Bucharest timezone

KEYBOARD
English (US)

LANGUAGE SUPPORT
English (United States)

SOFTWARE

INSTALLATION SOURCE
Local media

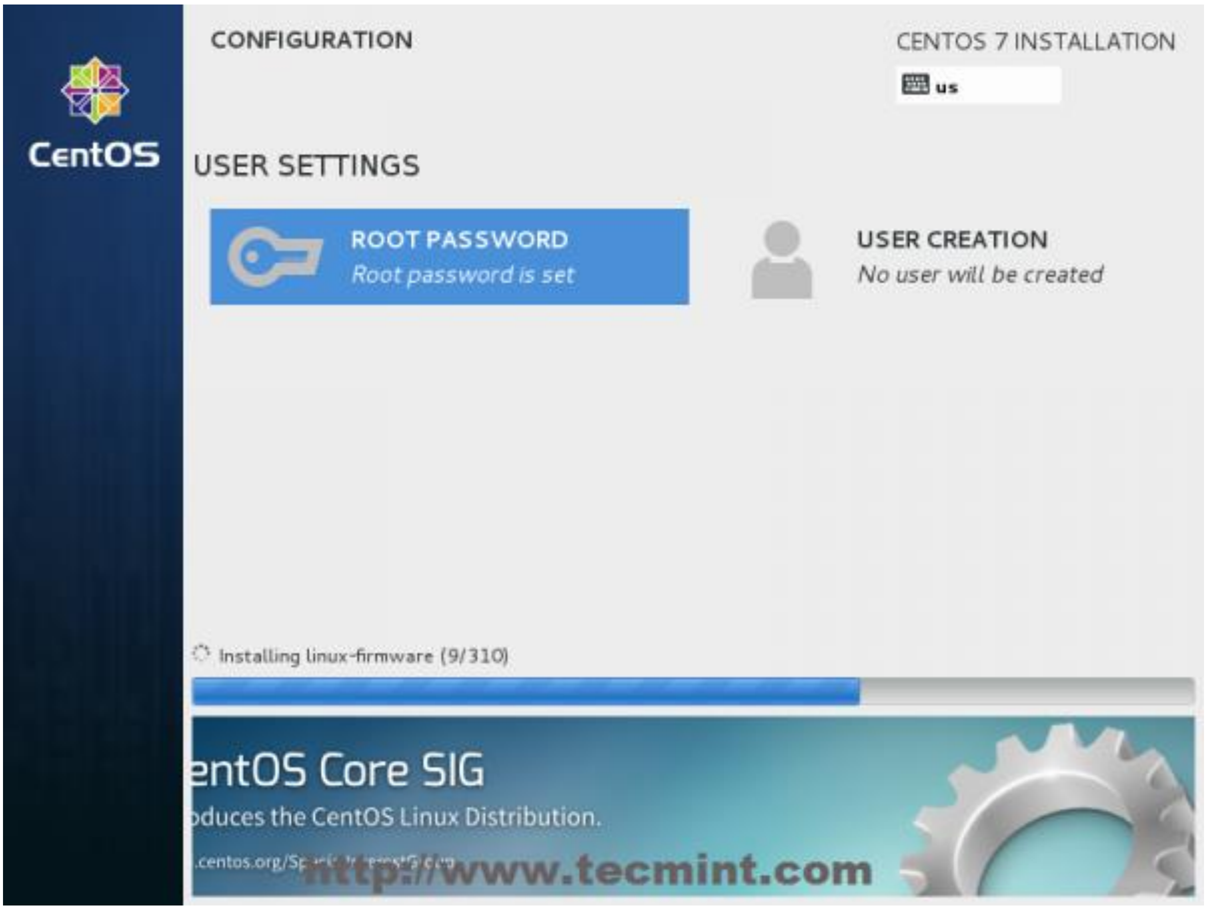
SOFTWARE SELECTION
Minimal Install

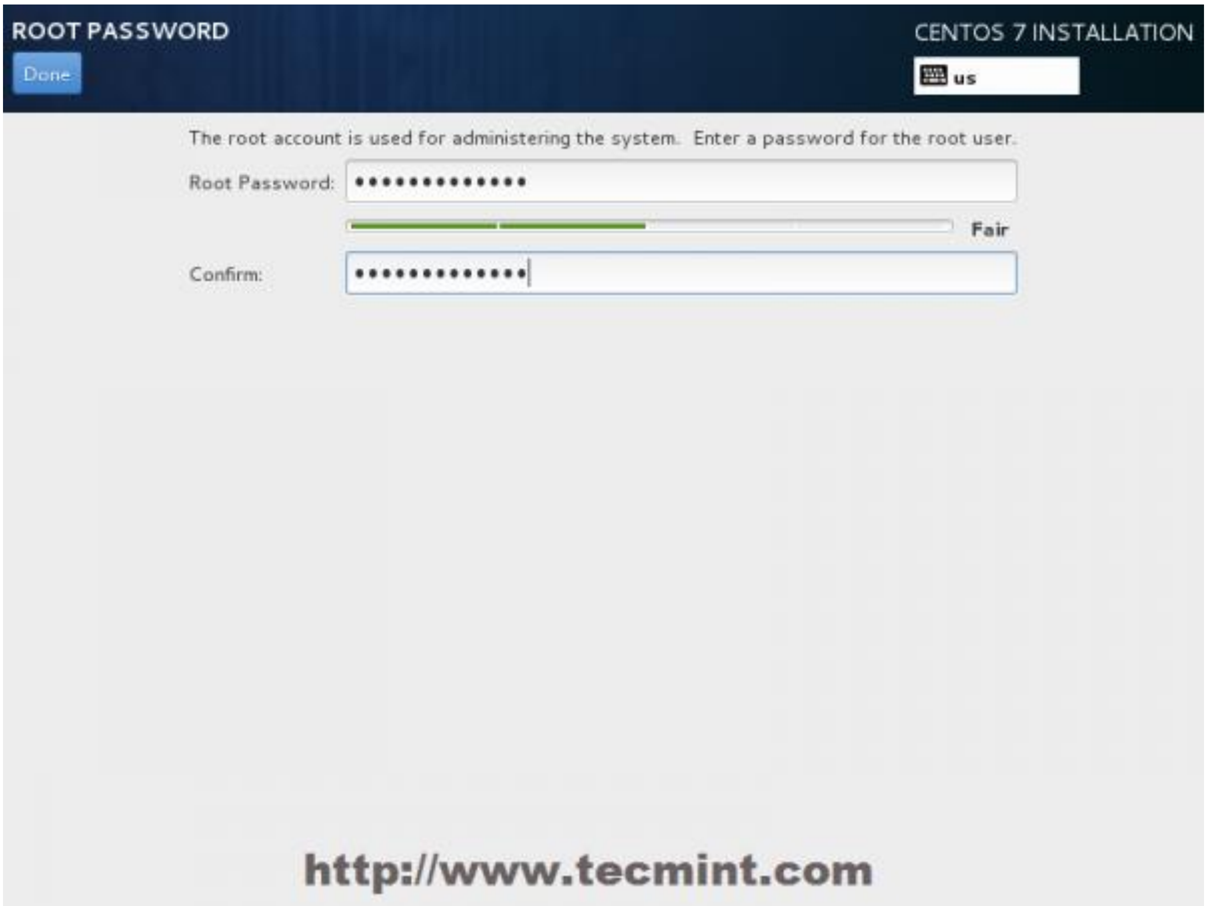
SYSTEM

INSTALLATION DESTINATION
Custom partitioning selected

NETWORK & HOSTNAME
Wired (eno167...736) connecte

We won't touch your disks until you click 'Begin Installation'





The root account is used for administering the system. Enter a password for the root user.

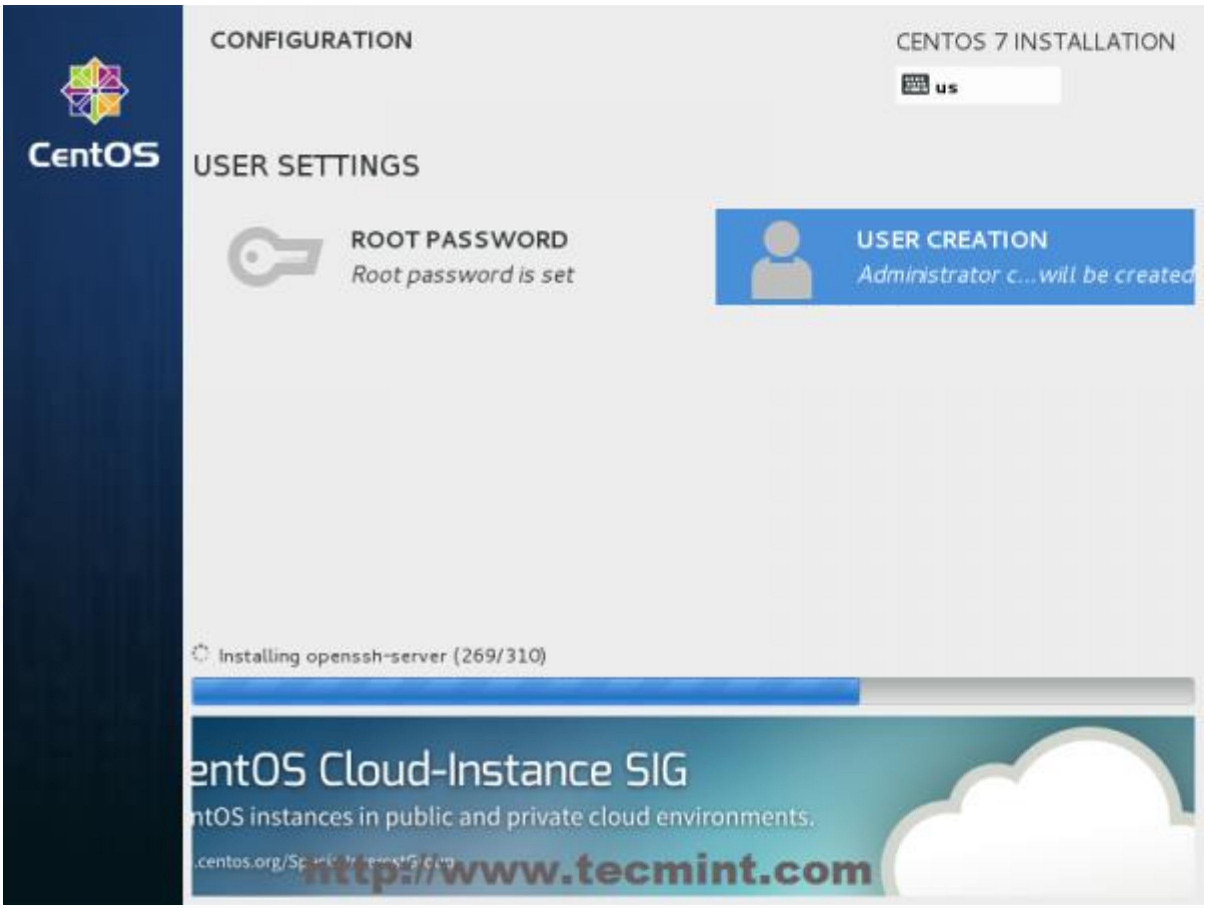
Root Password: ••••••••••••••

Fair

Confirm: ••••••••••••••

14. After you finish setting up a strong password for root account move to User Creation and create your first system user. You can designate this user to become a System Admin with root privileges using sudo command by checking the box Make this user administrator, then click on Done to go back on main menu and wait for the installation process to finish. Although it can be done later we suggest to create a user called “spark” or “hadoop” from the beginning, giving him root access, this user is required in the following steps to correctly install and run Hadoop cluster.

!!!IMPORTANT!!!: “spark” user needs to exist and have root privilege on **all** node of the cluster.



CONFIGURATION

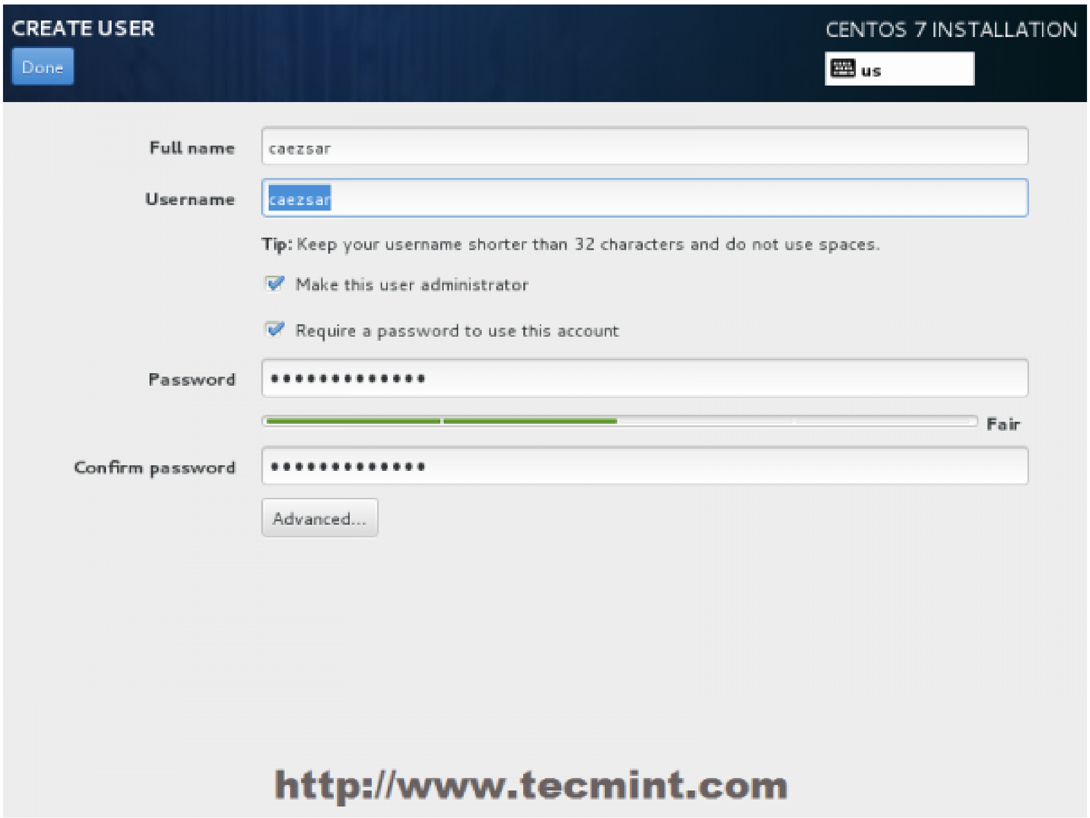
CENTOS 7 INSTALLATION

USER SETTINGS

ROOT PASSWORD
Root password is set

USER CREATION
Administrator c...will be created

Installing openssh-server (269/310)



CREATE USER

CENTOS 7 INSTALLATION

Full name: caezsar

Username: caezsar

Tip: Keep your username shorter than 32 characters and do not use spaces.

☑ Make this user administrator

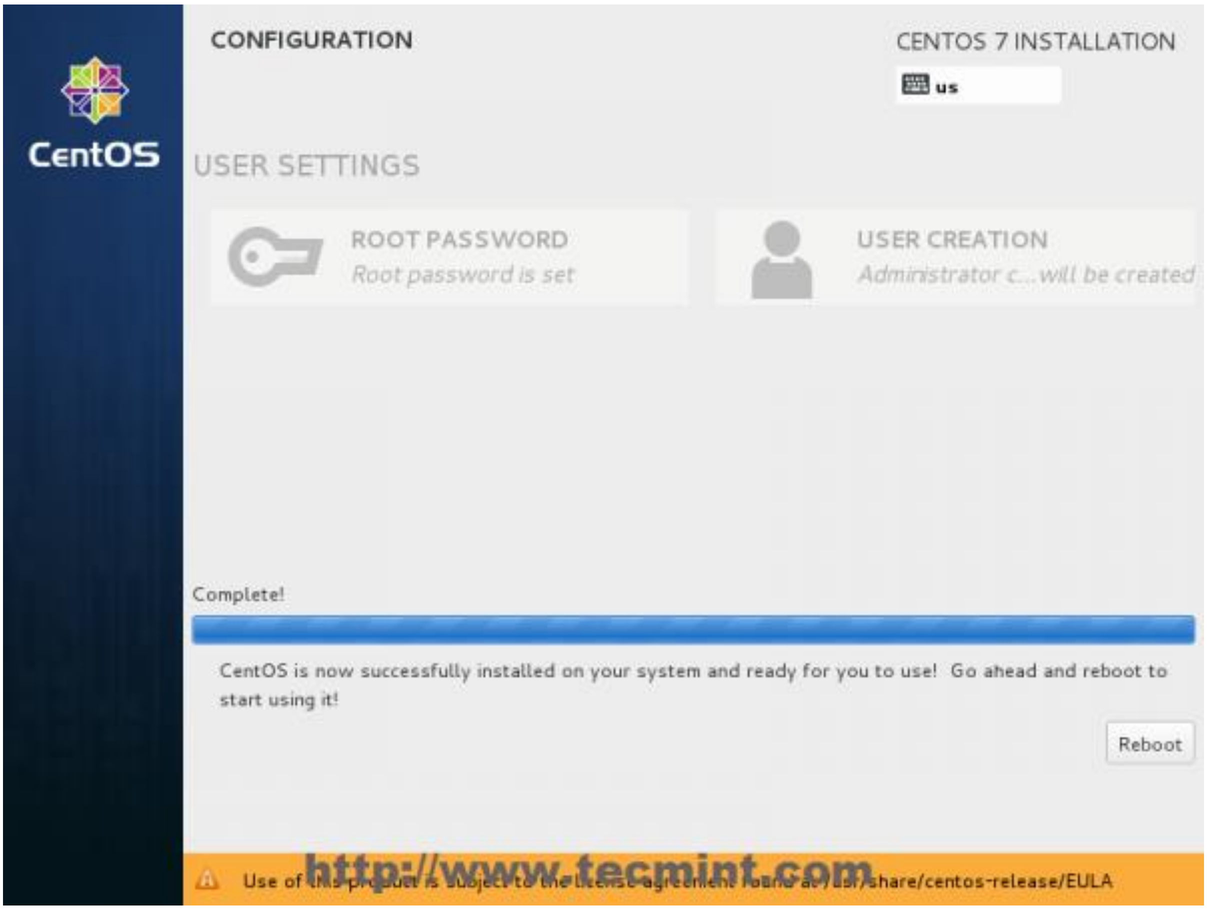
☑ Require a password to use this account

Password: •••••••••••••

Fair

Confirm password: •••••••••••••

15. After the installation process finishes, the installer will show a successfully message on screen, demanding to reboot your system in order to use it.



Root password is set

Administrator c...will be created

Complete!

CentOS is now successfully installed on your system and ready for you to use! Go ahead and reboot to start using it!

Use of this product is subject to the license agreement found at /usr/share/centos-release/EULA

16. Congratulation! You have now installed last version of CentOS on your bare new machine. Remove any installation media and reboot your computer so you can login to your new minimal CentOS 7 environment and ~~bash your head trying to make Hadoop work~~ perform other system tasks, such as update your system and install other useful software needed to run day to day tasks.

BONUS: Enabling/Disabling GUI

In some cases it can be useful to have a GUI, we choosed to install the minimal one, that at the moment is disabled. To activate/deactivate the system GUI is very simple.
In CentOS 7, the systemd process replaces the init process for starting services at boot time and also for changing the runlevels. It uses “targets” instead of run-levels and relies on systemctl command to change runlevel or to change the target.
With the command:

```
$ systemctl get-default
```

we can determine the state that the system currently configured to boot to. Because the system is running on Non-GUI Mode, “systemctl get-default” command will return “`multi-user.target`”. If the system is running on GUI Mode, “systemctl get-default” command will return “`graphical.target`”
So, to change default runlevel from Non-GUI (text-based) mode to GUI in CentOS 7 we simply have to write:

```
$ systemctl set-default graphical.target
```

And, to do the opposite:

```
$ systemctl set-default multi-user.target
```

Then reboot the system and changes will be applied.

2. Managing LVM

WARNING: Working on partitions is a very delicate and dangerous operation, you should **NEVER** do it without having a proper backup of the partitions you want to work with or, even better, an 1:1 image of the entire disk.

Have you messed up setting partitions on your new CentOS system? Do you have a system already in place but want to change your partitions dimensions without formatting the whole disk? Don't worry with LVM you can.

What is LVM?

Logical Volume Management (LVM) is a powerful, robust mechanism for managing storage space. LVM can be considered as a thin software layer on top of the hard disks and partitions, which creates an abstraction of continuity and ease-of-use for managing hard drive replacement, repartitioning and backup. Due to his high flexibility, it is used by CentOS operating system. The main advantage of this mechanism includes the possibility of creating single logical volumes of multiple physical volumes or entire hard disk, allowing for dynamic volume resizing. This permits to manage a large number of hard disk allowing to add or replace an hard disk without downtime or service disruption. So LVM allows filesystem to be easily resized as needed.

In LVM, there are several layers, each build on top of the other:

PV[s] (Physical Volumes) -> VG[s] (Volume Groups) -> LV[s] (Logical Volumes) -> Filesystems.

Logical Volumes are allocated/extended within the boundaries of their underlying storage pool which is called a Volume Group in LVM terminology. It is important to notice that we can only extend a Logical Volume within the free space of the underlying Volume Group.

Actual configuration on our server.

Here is the actual partition map of the master server (the slave has the same configuration):

```
[spark@master ~]$ lsblk
NAME                MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda                   8:0    0   931G  0 disk
├─sda1                8:1    0  39,2M  0 part
├─sda2                8:2    0     2G  0 part
├─sda3                8:3    0     1G  0 part /boot
├─sda4                8:4    0     1K  0 part
└─sda5                8:5    0 609,8G  0 part
  ├─centos-root 253:0    0   300G  0 lvm  /
  ├─centos-swap 253:1    0   9,8G  0 lvm  [SWAP]
  └─centos-home 253:2    0   300G  0 lvm  /home
sr0                  11:0    1  1024M  0 rom
```

```
[spark@master ~]$ sudo fdisk -l
Disk /dev/sda: 999.7 GB, 999653638144 bytes, 1952448512 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Identificativo disco: 0x573a96bf

Dispositivo Boot      Start         End      Blocks   Id  System
/dev/sda1                63       80324       40131   de  Dell Utility
/dev/sda2   *         81920     4276223     2097152    c  W95 FAT32 (LBA)
/dev/sda3           4276224     6373375     1048576   83  Linux
/dev/sda4           6373376  1952448511   973037568    5  Extended
/dev/sda5           6375424  1285152767   639388672   8e  Linux LVM
```

```
Disk /dev/mapper/centos-root: 322.1 GB, 322122547200 bytes, 629145600 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/mapper/centos-swap: 10.5 GB, 10481565696 bytes, 20471808 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/mapper/centos-home: 322.1 GB, 322122547200 bytes, 629145600 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

Instead of the command `lsblk` or `fdisk -l` you can also use `df -h` to see partition map.

The filesystem table is:

```
[spark@master ~]$ cat /etc/fstab
#
# /etc/fstab
# Created by anaconda on Thu Jun 14 17:41:53 2077
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root /                       xfs     defaults        0 0
UUID=b80a8b2f-5db9-40de-987c-2d087e114d28 /boot xfs     defaults        0 0
/dev/mapper/centos-home /home                   xfs     defaults        0 0
/dev/mapper/centos-swap swap                    swap    defaults        0 0
```

As can be seen, we choose not to allocate all the space of the disk, but to leave around 250GB of unallocated space. There is a reason for that.

Old setup problem.

In the old server setup we encountered the problem of managing a resize of the partitions, because on the first installation the partitions were badly set. To be more precise, we had the `root` partition with just 50gb of space allocated, the remaining space (around 950gb) was allocated entirely to the `home` partition. So, when the space in the root partition ended (and linux really does not like that, when the space in root partition finish you are not allowed to do a lot of operation) we had to increase the dimension of the partition. Despite the flexibility of the LVM, that was still a problem. Because our server had just one disk, we couldn't grow the root partition into another hard disk, the only options remaining was to reduce the home partition and increase the root one. That is not a trivial operation. Linux offer several commands to manage logical partition, in particular `lvresize` to change the dimension of a partition and `lvextend` to increase the dimension. But, meanwhile the operation of extending an existing partition is considered almost safe, the opposite one is **NOT**. Reducing the size of a logical volume is one of the riskier operation for the data involved. In fact, we lost all the data that we had in the home partition (both in master and slave), so a backup is **MANDATORY**.
For this reason, is wise to never allocate all the space to the `home` and `root` partition when you are creating it, is better to leave some unallocated space so that you can grow the desired partition as needed.

How to resize LVM partitions.

In order to resize partitions, after a lot of research, we decided proceeded in this way:

We made a backup of the home partition doing a simple tar of it:

```
$ tar -czvf  /place/in/which/save/the/backup.tgz /home/*
```

We unmounted the /home partition:

```
$ umount /partition/to/unmount/
```

Then resized it, the option L means that the dimension will be set at the desired amount of space, in this case 300GB:

```
$ lvresize -L 300GB /partition/to/resize/
```

We remade the filesystem (IMPORTANT: CentOS standard partition are xfs)

```
$ mkfs -t xfs -f /partition/resized/before
```

Then re-mounted home partition:

```
$ mount -t type device destination_dir
```

And restore the home backup:

```
$ tar -xzvf /mnt/backupdisk/home_bkp.tgz -C /home/
```

It can also be necessary to modify the `fstab` file, but usually if the partition is labelled correctly (es. Home) a reboot can be enough.

For the root partition, because LVM are intended for make this kind of operation, is sufficient to do this:

```
$ lvextend -t -r -L 150GB /dev/centos/root
```

where the `-r` option resize underlying filesystem together with the LV using fsadm, and the `-t` is test.

NOTE: It's better to **extend partition before they are completely full**, as said before Linux doesn't like when there is no space in root partition. To be more precise, in the case of `lvresize` and `lvextend`, you will have to skip some control that is better to not touch and renounce to the automatic backup.

We still advice to make a backup also of the root partition, just to be sure.

There is also a graphical way to do the same operation, using the tool "`system-config-lvm`" made by RedHat. Because we didn't have a GUI (to enable GUI in our server, see the apposite section), we decided in the end to don't use that, but we still tried it on another system with a live CD.

You can install the tool with:

```
$ yum install system-config-lvm
```

Then launch it with:

```
$ sudo system-config-lvm
```

3. Securing your server

Required Steps: Having a functional installation of CentOS 7 System with LVM Partitions, for a brand-new installation refer to Chapter 1.

Securing your server is an important part in managing a server, in this chapter you'll be guided through basic need-to-know knowledge on security and how to set up a firewall and a passwordless SSH, the latter being the most important part to ensure Hadoop functionality.

Update Your System–Frequently

Keeping your software up to date is the single biggest security precaution you can take for any operating system. Software updates range from critical vulnerability patches to minor bug fixes, and many software vulnerabilities are actually patched by the time they become public.

Automatic Security Updates

There are arguments for and against automatic updates on servers. Fedora's Wiki has a good breakdown of the pros and cons, but the risk of automatic updates will be minimal if you limit them to security updates. Not all package managers make that easy or possible, though. The practicality of automatic updates is something you must judge for yourself because it comes down to what you do with your server. Bear in mind that automatic updates apply only to packages sourced from repositories, not self-compiled applications. You may find it worthwhile to have a test environment that replicates your production server. Updates can be applied there and reviewed for issues before being applied to the live environment. On CentOS you can use "yum-cron" for automatic updates.

Disallow root logins over SSH.

This requires all SSH connections be by non-root users. Once a limited user account is connected, administrative privileges are accessible either by using sudo or changing to a root shell using:

```
$  sudo -i
```

Modify: `/etc/ssh/sshd_config`

```
# Authentication:

...

PermitRootLogin no
```

3.1 Configuring Firewall

Required Steps: Chapter 1.

Using a firewall to block unwanted inbound traffic to your server provides a highly effective security layer. By being very specific about the traffic you allow in, you can prevent intrusions and network mapping. A best practice is to allow only the traffic you need, and deny everything else. See our documentation on some of the most common firewall applications: FirewallD is the iptables controller available for the CentOS / Fedora family of distributions.

Out of the box, enterprise Linux distributions such as CentOS or RHEL come with a powerful firewall built-in, and their default firewall rules are pretty restrictive. Thus, if you install any custom services (e.g., web server, NFS, Samba, Hadoop, Yarn, Spark), chances are their traffic will be blocked by the firewall rules. You need to open up necessary ports on the firewall to allow their traffic.

On CentOS/RHEL 6 or earlier, the iptables service allows users to interact with netfilter kernel modules to configure firewall rules in the user space. Starting with CentOS/RHEL 7, however, a new userland interface called firewalld has been introduced to replace iptables service.

To check the current firewall rules, use this command:

```
$ sudo iptables -L
```

Open a Port on CentOS/RHEL 7

Starting with CentOS and RHEL 7, firewall rule settings are managed by firewalld service daemon. A command-line client called firewall-cmd can talk to this daemon to update firewall rules permanently.

To open up a new port (e.g., TCP/80) permanently, use these commands.

```
$ sudo firewall-cmd --zone=public --add-port=80/tcp --permanent

$ sudo firewall-cmd --reload
```

Without "--permanent" flag, the firewall rule would not persist across reboots.

After each new rule remember to reload the firewall.

Check the updated rules with:

```
$ firewall-cmd --list-all
```

If you need to test if a port is open you can just use one of these commands:

```
$ telnet ip port

$ nc -vz ip port
```

The most important thing in our case is to ensure that every communication between the nodes in our cluster goes smooth, so we need to do two things:

First, we assign a custom name to each of our nodes so we don't have to type their ip address ever again, to do so just go to /etc/hosts, open it with a text editor like nano, vi or `gedit` if you have installed a gui and add the following lines:

```
IP_ADDRESS1 master

IP_ADDRESS2 slave
```

Save and exit. Repeat for every node in the cluster.

Next we can add a special rule to our firewall to ensure no communication will get blocked between our nodes:

```
$ sudo firewall-cmd --permanent --zone=public --add-source=IP_ADDRESS
```

CentOS 7 firewall defaults to zone=public, if you are not sure what you zone is just type:

```
$ sudo firewall-cmd --get-default-zone
```

Add this rule between every node in the cluster.

3.2 Configuring SSH and passwordless SSH

Required Steps: Chapter 1./3.1

SSH, also known as Secure Socket Shell, is a network protocol that provides administrators with a secure way to access a remote computer. SSH also refers to the suite of utilities that implement the protocol. Secure Shell provides strong authentication and secure encrypted data communications between two computers connecting over an insecure network such as the Internet. SSH is widely used by network administrators for managing systems and applications remotely, allowing them to log in to another computer over a network, execute commands and move files from one computer to another.

By default, password authentication is used to connect to your server via SSH. A cryptographic key-pair is more secure because a private key takes the place of a password, which is generally much more difficult to brute-force. In this section we'll create a key-pair and configure the server to not accept passwords for SSH logins between the 2 node of our cluster, this operation is fundamental to let Hadoop communicate between the nodes.

1. Create an Authentication Key-pair

This first step is done on your master first and then on your slave, you will create a 4096-bit RSA key-pair. Start logging in with your "spark" user created in Chapter 1. During the creation of the keys, you will be given the option to encrypt the private key with a passphrase. This means that it cannot be used without entering the passphrase, unless you save it to your local desktop's keychain manager. We suggest you use the key-pair without a passphrase, but you are free to use one. If you've already created an RSA key-pair, this command will overwrite it, potentially locking you out of other systems. If you've already created a key-pair, skip this step. To check for existing keys, run:

```
$ ls ~/.ssh/id rsa*.
```

To create the key type:

```
$ ssh-keygen -b 4096
```

Press Enter to use the default names id_rsa and id_rsa.pub in /home/spark/.ssh before it asks you to enter your passphrase, at this point you can choose what to do, if you don't want to use any passphrase just press enter.

2. Upload the public key to your nodes

Replace spark with the name of the user you plan to administer Hadoop as.

```
$ ssh-copy-id spark@master
$ ssh-copy-id spark@slave
```

Or:

```
$ ssh-copy-id -i $HOME/.ssh/id_rsa.pub spark@master
$ ssh-copy-id -i $HOME/.ssh/id_rsa.pub spark@slave
```

3. Repeat for all nodes.

You should repeat key generation and key sharing on all the nodes of the cluster even into the same node, so in our case we did: master → slave, master → master, slave → master, slave → slave. Remember, you need one key for master and one for slave, but you need to copy it to both yourself and the other node. This is because Hadoop will ssh to the same node, and if a password is required you need to enter it manually every time.

4. Now exit and log back into your node, the try from one of the nodes to ssh to the others.

```
$ ssh spark@slave
```

If you specified a passphrase for your private key, you'll need to enter it.

4. Installing Java, Python and Scala

For correct operation Hadoop/Spark needs the latest version of Java, Python and Scala. Let's see how to install it in CentOS:

Installing Java

The actual version of Java in our system is:

```
openjdk version "1.8.0_171"
OpenJDK Runtime Environment (build 1.8.0_171-b10)
OpenJDK 64-Bit Server VM (build 25.171-b10, mixed mode)
```

And can be checked with command:

```
$ java -version
```

We choose to install openjdk instead of oraclejdk because of compatibility reason.
NOTE: at the moment in which we are writing this guide, **Java 1.9 is not compatible** with Hadoop/Spark.

You can install it with the command:

```
$ sudo yum install java-1.8.0-openjdk-devel
```

Installing Scala

The actual version of Scala in our system is:

```
$ Scala code runner version 2.12.6 -- Copyright 2002-2018, LAMP/EPFL and Lightbend, Inc.
```

And can be checked with command:

```
$ scala -version
```

In theory, there is a very simple way to install Scala on Centos, and that is:

```
$ wget link.to.last.scala.version.rpm
$ sudo yum localinstall -y last.scala.version.downloaded.before.rpm
```

But, for a mysterious reason, in our case that didn't work. So we solved it with this script:

`install_scala_centos.sh`

```
export SCALA_VERSION=scala-2.12.6
sudo wget http://www.scala-lang.org/files/archive/${SCALA_VERSION}.tgz
sudo echo "SCALA_HOME=/usr/local/scala/scala-2.12.6" > /etc/profile.d/scala.sh
sudo echo 'export SCALA_HOME' >> /etc/profile.d/scala.sh
sudo mkdir -p /usr/local/scala
sudo -s cp $SCALA_VERSION.tgz /usr/local/scala/
cd /usr/local/scala/
sudo -s tar xvf $SCALA_VERSION.tgz
sudo rm -f $SCALA_VERSION.tgz
sudo chown -R root:root /usr/local/scala

sudo update-alternatives --install "/usr/bin/scala" "scala" "/usr/local/scala/scala-2.12.6/bin/scala" 1
sudo update-alternatives --install "/usr/bin/scalac" "scalac" "/usr/local/scala/scala-2.12.6/bin/scalac" 1
sudo update-alternatives --install "/usr/bin/scalap" "scalap" "/usr/local/scala/scala-2.12.6/bin/scalap" 1
sudo update-alternatives --install "/usr/bin/scaladoc" "scaladoc" "/usr/local/scala/scala-2.12.6/bin/scaladoc" 1
sudo update-alternatives --install "/usr/bin/fsc" "fsc" "/usr/local/scala/scala-2.12.6/bin/fsc" 1
```

Installing Python

The actual version of Python in our system is (also in that case, we choosed 2.7 instead of 3.7 due to compatibility advantages):

```
Python 2.7.5
```

And can be checked with command:

```
python –V
```

You can install it with the command:

```
sudo yum install -y python27
```

5. Installing Hadoop cluster

Required Steps: Chapter 1/3.1/3.2/4

Hadoop is an open-source Apache project that allows creation of parallel processing applications on large data sets, distributed across networked nodes. It's composed of the Hadoop Distributed File System (HDFS™) that handles scalability and redundancy of data across nodes, and Hadoop YARN: a framework for job scheduling that executes data processing tasks on all nodes.

Before you begin be sure to have completed all the necessary steps needed to set your firewall, ssh connection between nodes and installation of java, python and scala. The steps below use example ip for each node. Adjust each example according to your configuration.

Architecture of a Hadoop Cluster

Before configuring the master and slave node, it's important to understand the different components of a Hadoop cluster. A master node keeps knowledge about the distributed file system, like the inode table on an ext3 filesystem, and schedules resources allocation. Master node will handle this role in this guide, and host two daemons:

The **NameNode**: manages the distributed file system and knows where stored data blocks inside the cluster are.

The **ResourceManager**: manages the YARN jobs and takes care of scheduling and executing processes on slave nodes.

Slave nodes store the actual data and provide processing power to run the jobs. In our setup we will use only one, called slave, and will host two daemons:

The **DataNode** manages the actual data physically stored on the node; it's named, **NameNode**.

The **NodeManager** manages execution of tasks on the node.

Did you know YARN isn't the only resource manager you can use on your server? Do you want to know more? Check this out.

5.1 Installation

We will configure Hadoop first on our master and then copy the configured copy on Hadoop to all our nodes.

Download and Unpack Hadoop Binaries

Login to node-master as the hadoop user, download the Hadoop tarball from Hadoop project page, and unzip it:

```
$ wget http://apache.panu.it/hadoop/common/hadoop-3.0.3/hadoop-3.0.3.tar.gz
$ tar -xzf Hadoop-3.0.3.tar.gz
$ mv Hadoop-3.0.3.tar.gz /Hadoop
```

As you can see, we installed Hadoop in the root, that way any user in the system will be able to use it.

Set Environment Variables

Add Hadoop binaries to your PATH. To set Environment Variables for every user on the system you need to edit `/etc/profile` and add the following line:

```
export PATH=/hadoop/bin:/hadoop/sbin:$PATH
```

You will also need to add the following lines, don't worry about their usage, they will become helpful later:

```
export PATH=/hadoop/bin:/hadoop/sbin:$PATH
export HADOOP_PREFIX="/hadoop"
export PATH=$PATH:$HADOOP_PREFIX/bin
export PATH=$PATH:$HADOOP_PREFIX/sbin
export HADOOP_MAPRED_HOME=${HADOOP_PREFIX}
export HADOOP_COMMON_HOME=${HADOOP_PREFIX}
export HADOOP_HDFS_HOME=${HADOOP_PREFIX}
export YARN_HOME=${HADOOP_PREFIX}
```

After saving and exiting the text editor, type in your shell this command to update the system Environment Variables:

```
$ source /etc/profile
```

In the end our `/etc/profile` file looks like this:

```
unset i
unset -f pathmunge
export JAVA_HOME=/usr/lib/jvm/jre-1.8.0-openjdk
export PATH=$JAVA_HOME/bin:$PATH
export JRE_HOME=/usr/lib/jvm/jre
export SCALA_HOME=/usr/local/scala/scala-2.12.6
export PATH=$PATH:$SCALA_HOME/bin
export PATH=/hadoop/bin:/hadoop/sbin:$PATH
export HADOOP_PREFIX="/hadoop"
export PATH=$PATH:$HADOOP_PREFIX/bin
export PATH=$PATH:$HADOOP_PREFIX/sbin
export HADOOP_MAPRED_HOME=${HADOOP_PREFIX}
export HADOOP_COMMON_HOME=${HADOOP_PREFIX}
export HADOOP_HDFS_HOME=${HADOOP_PREFIX}
export YARN_HOME=${HADOOP_PREFIX}
```

Set JAVA_HOME

Get your Java installation path. If you installed open-jdk from your package manager, you can get the path with the command:

```
$ update-alternatives --display java
```

Take the value of the current link and remove the trailing `/bin/java`. For example on Debian, the link is `/usr/lib/jvm/java-8-openjdk-amd64/jre/bin/java`, so JAVA_HOME should be `/usr/lib/jvm/java-8-openjdk-amd64/jre`.

If you installed java from Oracle, JAVA_HOME is the path where you unzipped the java archive.

Edit `/hadoop/etc/hadoop/hadoop-env.sh` and replace this line:

```
export JAVA_HOME=${JAVA_HOME}
```

with your actual java installation path. For example on a Debian with open-jdk-8:

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/jre
```

Set core-site.xml

Open `/Hadoop/etc/Hadoop/core-site.xml` with:

```
$ sudo nano /hadoop/etc/Hadoop/core-site.xml
```

And add the following lines:

```
<configuration>

<property>
<name>fs.defaultFS</name>
<value>hdfs://192.167.155.71:9000</value>
</property>

<property>
<name>fs.default.name</name>
<value>hdfs://192.167.155.71:9000</value>
</property>

<property>
<name>hadoop.tmp.dir</name>
<value>/tmp</value>
</property>

</configuration>
```

Set hdfs-site.xml

Open /Hadoop/etc/Hadoop/hdfs-site.xml with:

```
$ sudo nano /hadoop/etc/Hadoop/hdfs-site.xml
```

And add the following lines:

```
<configuration>

<property>
<name>dfs.namenode.name.dir</name>
<value>/hadoop/data/nameNode</value>
</property>

<property>
<name>dfs.datanode.data.dir</name>
<value>/hadoop/data/dataNode</value>
</property>

<property>
<name>dfs.webhdfs.enabled</name>
<value>true</value>
</property>
```

```
<property>
<name>dfs.http.address</name>
<value>master:9870</value>
</property>

<property>
<name>dfs.datanode.data.dir</name>
<value>/hadoop/data/dataNode</value>
</property>

<property>
<name>dfs.webhdfs.enabled</name>
<value>true</value>
</property>

<property>
<name>dfs.http.address</name>
<value>master:9870</value>
</property>

<property>
<name>mapred.job.tracker.http.address</name>
<value>master:9871</value>
</property>

<property>
<name>dfs.permissions.enabled</name>
<value>true</value>
</property>

<property>
<name>dfs.replication</name>
<value>2</value>
</property>

</configuration>
```

The last property, dfs.replication, indicates how many times data is replicated in the cluster. You can set 2 to have all the data duplicated on the two nodes. Don’t enter a value higher than the actual number of slave nodes.

Set YARN as Job Scheduler

In /hadoop/etc/hadoop/, rename mapred-site.xml.template to mapred-site.xml:

```
$ cd /hadoop/etc/hadoop

$ mv mapred-site.xml.template mapred-site.xml
```

Edit the file, setting yarn as the default framework for MapReduce operations:

```
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
```

Configure YARN

Edit `yarn-site.xml`:

```
<configuration>

<property>
<name>yarn.acl.enable</name>
<value>0</value>
</property>

<property>
<name>yarn.resourcemanager.hostname</name>
<value>master</value>
</property>

<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>

</configuration>
```

Configure workers

The file workers is used by startup scripts to start required daemons on all nodes. Edit `/hadoop/etc/hadoop/workers` to be:

```
master

slave
```

Ok, we are almost done! Now you can proceed to duplicate Hadoop between all nodes in your cluster or you can configure additional settings on YARN; we strongly suggest the second options, so jump to Chapter 6 and then come back to this part, or you could just go ahead and start duplicating Hadoop configuration across the cluster without configuring YARN.

Duplicate Config Files on Each Node

Copy the hadoop binaries to slave node:

```
$ cd /

$ scp hadoop slave:/
```

Connect to slave via ssh. A password isn't required, thanks to the ssh keys copied in Chapter 3.2:

```
$ ssh slave
```

And check that everything went fine. Repeat these steps for each node on your cluster.

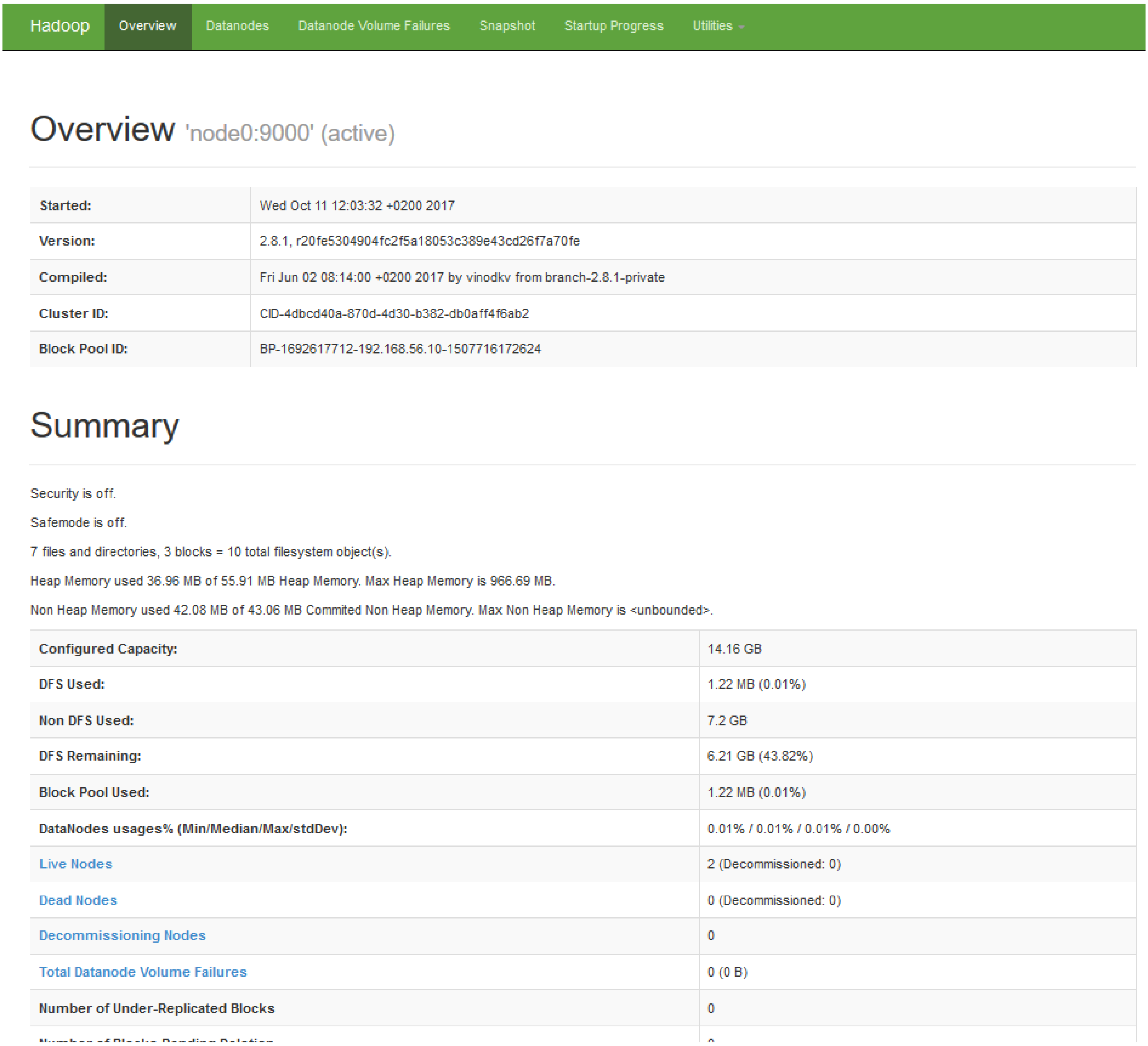
Next step is to set up web interfaces and making sure that you can manage e monitor your cluster from remote, but if you don't care about this part you can skip ahed to Chapter 7 where we'll be discussing how to install Spark on your cluster. Don't worry! It's going to be much easier than what you just did!

5.2 Web Interface

You can also automatically use the friendlier web user interface. Point your browser to `http://master:9870` and you'll get a user-friendly monitoring console.

If you can't reach the page there could be two possible problems:

1) You haven't started hdfs or there was an error on start-up, try and follow the steps in Chapter 8 after you complete installation and configuration process.
2) Your port was blocked from the server firewall, don't worry go back at Chapter 3.1 to figure out how to open ports and follow those steps for port 9870.



Overview 'node0:9000' (active)

Started:	Wed Oct 11 12:03:32 +0200 2017
Version:	2.8.1, r20fe5304904fc2f5a18053c389e43cd26f7a70fe
Compiled:	Fri Jun 02 08:14:00 +0200 2017 by vinodkv from branch-2.8.1-private
Cluster ID:	CID-4dbcd40a-870d-4d30-b382-db0aff4f6ab2
Block Pool ID:	BP-1692617712-192.168.56.10-1507716172624

Summary

Security is off.

Safemode is off.

7 files and directories, 3 blocks = 10 total filesystem object(s).

Heap Memory used 36.96 MB of 55.91 MB Heap Memory. Max Heap Memory is 966.69 MB.

Non Heap Memory used 42.08 MB of 43.06 MB Commited Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity:	14.16 GB
DFS Used:	1.22 MB (0.01%)
Non DFS Used:	7.2 GB
DFS Remaining:	6.21 GB (43.82%)
Block Pool Used:	1.22 MB (0.01%)
DataNodes usages% (Min/Median/Max/stdDev):	0.01% / 0.01% / 0.01% / 0.00%
Live Nodes	2 (Decommissioned: 0)
Dead Nodes	0 (Decommissioned: 0)
Decommissioning Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	0
Number of Blocks Pending Deletion	0

6. Setting YARN

Required Steps: Chapter 1/3.1/3.2/4/5

Configure Memory Allocation

Memory allocation can be tricky on low RAM nodes because default values are not suitable for nodes with less than 8GB of RAM. This section will highlight how memory allocation works for MapReduce jobs, and provide a sample configuration for 8GB RAM nodes.

The Memory Allocation Properties

A YARN job is executed with two kind of resources:

An Application Master (AM) is responsible for monitoring the application and coordinating distributed executors in the cluster.

Some executors that are created by the AM actually run the job. For a MapReduce jobs, they'll perform map or reduce operation, in parallel.

Both are run in containers on slave nodes. Each slave node runs a NodeManager daemon that's responsible for container creation on the node. The whole cluster is managed by a ResourceManager that schedules container allocation on all the slave-nodes, depending on capacity requirements and current charge.

Four types of resource allocations need to be configured properly for the cluster to work. These are:

How much memory can be allocated for YARN containers on a single node. This limit should be higher than all the others; otherwise, container allocation will be rejected and applications will fail. However, it should not be the entire amount of RAM on the node.

This value is configured in yarn-site.xml with *yarn.nodemanager.resource.memory-mb*.

How much memory a single container can consume and the minimum memory allocation allowed. A container will never be bigger than the maximum, or else allocation will fail and will always be allocated as a multiple of the minimum amount of RAM.

Those values are configured in yarn-site.xml with *yarn.scheduler.maximum-allocation-mb* and *yarn.scheduler.minimum-allocation-mb*.

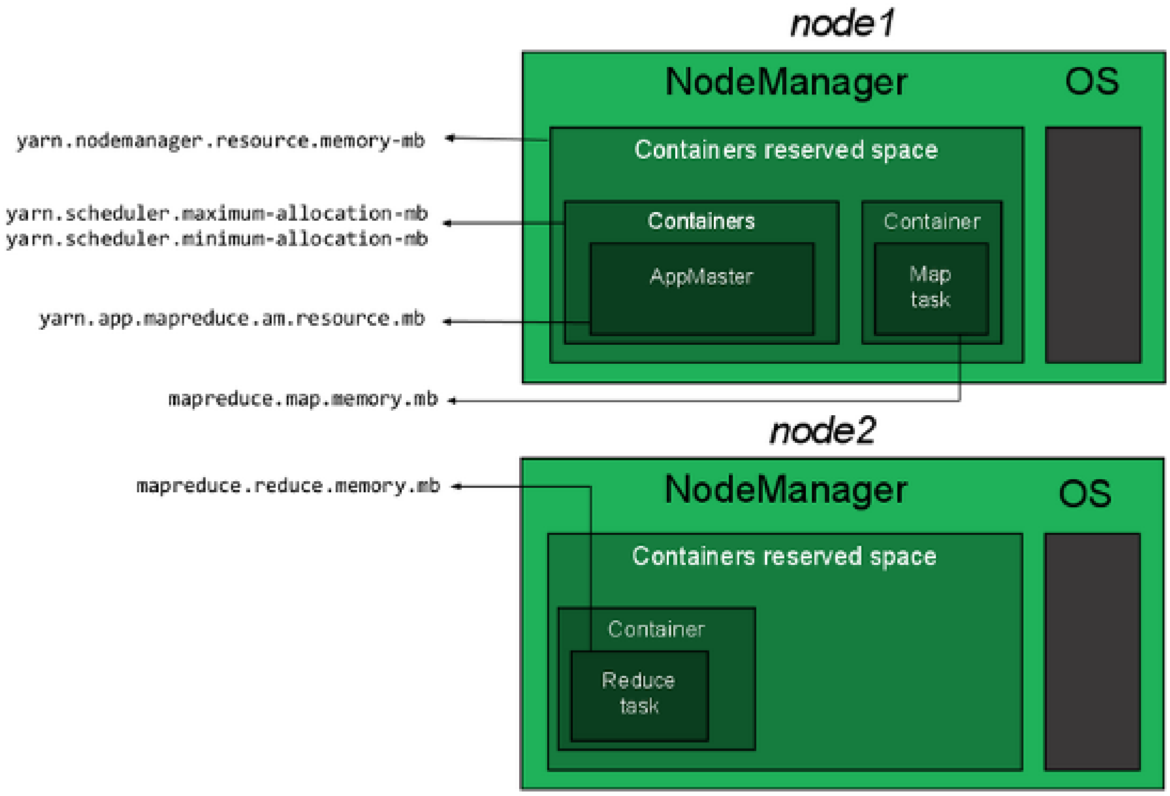
How much memory will be allocated to the ApplicationMaster. This is a constant value that should fit in the container maximum size.

This is configured in mapred-site.xml with *yarn.app.mapreduce.am.resource.mb*.

How much memory will be allocated to each map or reduce operation. This should be less than the maximum size.

This is configured in mapred-site.xml with properties *mapreduce.map.memory.mb* and *mapreduce.reduce.memory.mb*.

The relationship between all those properties can be seen in the following figure:


node1
NodeManager
OS
Containers reserved space
Containers
AppMaster
Container
Map task
yarn.nodemanager.resource.memory-mb
yarn.scheduler.maximum-allocation-mb
yarn.scheduler.minimum-allocation-mb
yarn.app.mapreduce.am.resource.mb
mapreduce.map.memory.mb
node2
NodeManager
OS
Containers reserved space
Container
Reduce task
mapreduce.reduce.memory.mb

6.1 Settings

Sample Configuration for 8GB Nodes

For 8GB nodes, a working configuration may be:

Edit `/hadoop/etc/hadoop/yarn-site.xml` and add the following lines:

```
<configuration>

<!-- Site specific YARN configuration properties -->

<property>
<name>yarn.acl.enable</name>
<value>0</value>
</property>

<property>
<name>yarn.resourcemanager.hostname</name>
<value>master</value>
</property>

<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>
```

```
<property>

<name>yarn.nodemanager.resource.memory-mb</name>
<value>6144</value>
</property>

<property>
<name>yarn.scheduler.maximum-allocation-mb</name>
<value>6144</value>
</property>

<property>
<name>yarn.resourcemanager.hostname</name>
<value>master</value>
</property>

<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>

<property>
<name>yarn.nodemanager.resource.memory-mb</name>
<value>6144</value>
</property>

<property>
<name>yarn.scheduler.maximum-allocation-mb</name>
<value>6144</value>
</property>

<property>
<name>yarn.scheduler.minimum-allocation-mb</name>
<value>128</value>
</property>

<property>
<name>yarn.nodemanager.vmem-check-enabled</name>
<value>false</value>
</property>

<property>
<name>yarn.webapp.ui2.enable</name>
<value>true</value>
</property>

<property>
<name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
<value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>

<property>
<name>yarn.scheduler.minimum-allocation-mb</name>
<value>128</value>
</property>
```

```
<property>
<name>yarn.webapp.ui2.enable</name>
<value>true</value>
</property>

<property>
<name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
<value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>

<property>
<name>yarn.application.classpath</name>
<value>

      /hadoop/etc/hadoop,
      /hadoop/share/hadoop/common/*,
      /hadoop/share/hadoop/common/lib/*,
      /hadoop/share/hadoop/hdfs/*,
      /hadoop/share/hadoop/hdfs/lib/*,
      /hadoop/share/hadoop/mapreduce/*,
      /hadoop/share/hadoop/mapreduce/lib/*,
      /hadoop/share/hadoop/yarn/*,
      /hadoop/share/hadoop/yarn/lib/*
</value>
</property>

<property>
<name>yarn.nodemanager.vmem-check-enabled</name>
<value>false</value>
</property>

</configuration>
```

The last property disables virtual-memory checking and can prevent containers from being allocated properly on JDK8.

Edit `/hadoop/etc/hadoop/mapred-site.xml` and add the following lines:

```
<configuration>

<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>

<property>
<name>yarn.app.mapreduce.am.resource.mb</name>
<value>512</value>
</property>

<property>
<name>mapreduce.map.memory.mb</name>
<value>256</value>
</property>
```

```
<property>
<name>mapreduce.reduce.memory.mb</name>
<value>256</value>
</property>

<property>
<name>mapreduce.application.classpath</name>
<value>yarn</value>
</property>

<property>
<name>yarn.app.mapreduce.am.resource.mb</name>
<value>512</value>
</property>

<property>
<name>mapreduce.map.memory.mb</name>
<value>256</value>
</property>

<property>
<name>mapreduce.reduce.memory.mb</name>
<value>256</value>
</property>

<property>
<name>mapreduce.application.classpath</name>
<value>
      /hadoop/etc/hadoop,
      /hadoop/share/hadoop/common/*,
      /hadoop/share/hadoop/common/lib/*,
      /hadoop/share/hadoop/hdfs/*,
      /hadoop/share/hadoop/hdfs/lib/*,
      /hadoop/share/hadoop/mapreduce/*,
      /hadoop/share/hadoop/mapreduce/lib/*,
      /hadoop/share/hadoop/yarn/*,
      /hadoop/share/hadoop/yarn/lib/*
</value>
</property>

</configuration>
```

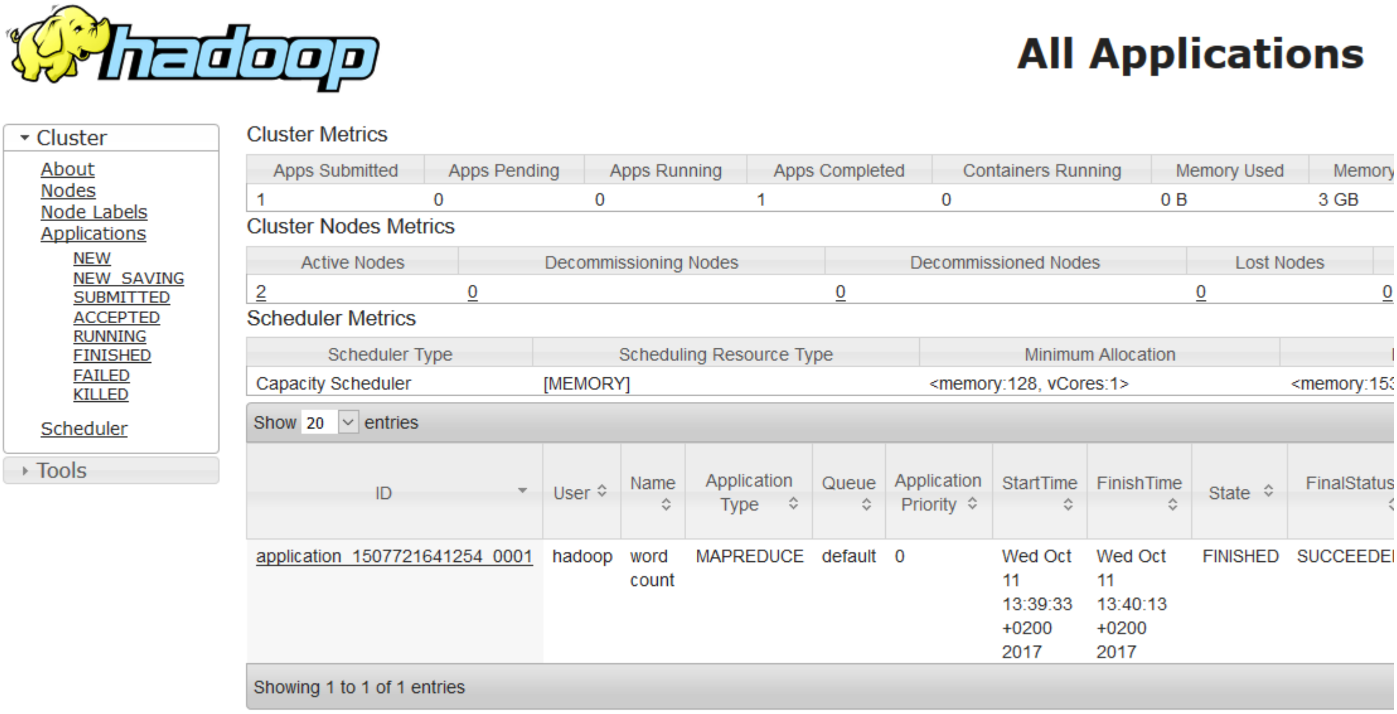
Now you’re all set and almost ready to start! If you have already done it, and you modified some settings in YARN you need to re-duplicate your Hadoop configuration to all nodes in the cluster, if you haven’t, just go back to Chapter 5.1 and follow the steps required to duplicate Hadoop configuration to the cluster.

6.2 Web Interface

As with HDFS, YARN provides a friendlier web UI, started by default on port 8088 of the Resource Manager. Point your browser to `http://master:8088` and browse the UI:

If you can't reach the page there could be two possible problems:

1) You haven't started YARN or there was an error on start-up, try and follow the steps in Chapter 8 after you complete installation and configuration process.
2) Your port was blocked from the server firewall, don't worry go back at Chapter 3.1 to figure out how to open ports and follow those steps for port 8088.



All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory
1	0	0	1	0	0 B	3 GB

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	
2	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	
Capacity Scheduler	[MEMORY]	<memory:128, vCores:1>	<memory:153

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus
application_1507721641254_0001	hadoop	word count	MAPREDUCE	default	0	Wed Oct 11 13:39:33 +0200 2017	Wed Oct 11 13:40:13 +0200 2017	FINISHED	SUCCEEDEI

Showing 1 to 1 of 1 entries

7. Installing Spark

Required Steps: Chapter 1/3.1/3.2/4/5/6

What is Spark?

Spark is a general purpose cluster computing system. It can deploy and run parallel applications on clusters ranging from a single node to thousands of distributed nodes. Spark was originally designed to run Scala applications, but also supports Java, Python and R.

Just like we did with Hadoop run the commands in this guide from master unless otherwise specified. Be sure you have a user (in our case “spark”) that can access all cluster nodes with SSH keys without a password. Note the path of your Hadoop installation. This guide assumes it is installed in /hadoop. If it is not, adjust the path in the examples accordingly.

Download and Install Spark Binaries

Spark binaries are available from the Apache Spark download page. Adjust each command below to match the correct version number.

Get the download URL from the Spark download page, download it, and uncompress it.

For Spark 2.3.1 with Hadoop 2.7 or later, log on master as the spark user, and run:

```
$ cd /

$ wget https://www.apache.org/dyn/closer.lua/spark/spark-2.3.1/spark-2.3.1-bin-hadoop2.7.tgz

$ tar -xvf spark-2.3.1-bin-hadoop2.7.tgz

$ mv spark-2.3.1-bin-hadoop2.7 spark
```

Add the Spark binaries directory to your PATH if you haven’t already done it. Edit `/etc/profile` and add the following line:

```
export PATH=/spark/bin:$PATH
```

And then type this shell command:

```
$ source /etc/profile
```

Integrate Spark with YARN

To communicate with the YARN Resource Manager, Spark needs to be aware of your Hadoop configuration. This is done via the HADOOP_CONF_DIR environment variable. The SPARK_HOME variable is not mandatory, but is useful when submitting Spark jobs from the command line.

Edit the system profile `/etc/profile` and add the following lines:

```
export HADOOP_CONF_DIR=/hadoop/etc/hadoop

export SPARK_HOME=/spark

export LD_LIBRARY_PATH=/hadoop/lib/native:$LD_LIBRARY_PATH
```

Then use this command to update:

```
$ source /etc/profile
```

Restart your session by logging out and logging in again.

```
$ mv $SPARK_HOME/conf/spark-defaults.conf.template $SPARK_HOME/conf/spark-defaults.conf
```

Rename the spark default template config file:

Edit `$SPARK_HOME/conf/spark-defaults.conf` and add this line to set spark.master to yarn:

```
spark.master    yarn
```

Spark is now ready to interact with your YARN cluster. Easy right?

Understand Client and Cluster Mode

Spark jobs can run on YARN in two modes: cluster mode and client mode. Understanding the difference between the two modes is important for choosing an appropriate memory allocation configuration, and to submit jobs as expected.

A Spark job consists of two parts: Spark **Executors** that run the actual tasks, and a Spark **Driver** that schedules the Executors.

Cluster mode: everything runs inside the cluster. You can start a job from your laptop and the job will continue running even if you close your computer. In this mode, the Spark Driver is encapsulated inside the YARN Application Master.

Client mode the Spark driver runs on a client, such as your laptop. If the client is shut down, the job fails. Spark Executors still run on the cluster, and to schedule everything, a small YARN Application Master is created.

Client mode is well suited for interactive jobs, but applications will fail if the client stops. For long running jobs, cluster mode is more appropriate.

For our cluster we choose Client Mode to avoid hogging resources.

Configure Memory Allocation

Allocation of Spark containers to run in YARN containers may **fail** if memory allocation is not configured properly. For nodes with less than 4G RAM, the default configuration is not adequate and may trigger swapping and poor performance, or even the failure of application initialization due to lack of memory.

Be sure to understand how Hadoop YARN manages memory allocation before editing Spark memory settings so that your changes are compatible with your YARN cluster's limits.

NOTE: See the memory allocation section of Chapter 6 and 6.1 for more details on managing your YARN cluster's memory.

Give Your YARN Containers Maximum Allowed Memory

If the memory requested is above the maximum allowed, YARN will reject creation of the container, and your Spark application won't start.

Get the value of *yarn.scheduler.maximum*-allocation-mb in `$HADOOP_CONF_DIR/yarn-site.xml`. This is the maximum allowed value, in MB, for a single container.

Make sure that values for Spark memory allocation, configured in the following section, are below the maximum.

This guide will use a sample value of 6144 for *yarn.scheduler.maximum-allocation-mb*. If your settings are lower, adjust the samples with your configuration.

Configure the Spark Driver Memory Allocation in Cluster Mode

Set the default amount of memory allocated to Spark Driver in cluster mode via *spark.driver.memory* (this value defaults to 1G). To set it to 5g, add this line to the file:

```
spark.driver.memory    5g
```

From the Command Line: Use the --driver-memory parameter to specify the amount of memory requested by spark-submit. See the following section about application submission for examples.

NOTE: Values given from the command line will override whatever has been set in `spark-defaults.conf`.

Configure the Spark Application Master Memory Allocation in Client Mode

In client mode, the Spark driver will not run on the cluster, so the above configuration will have no effect. A YARN Application Master still needs to be created to schedule the Spark executor, and you can set its memory requirements.

Set the amount of memory allocated to Application Master in client mode with spark.yarn.am.memory for 8GB ram system (default to 512M) in `$SPARK_HOME/conf/spark-defaults.conf:`

```
spark.yarn.am.memory    512m
```

This value can not be set from the command line.

Configure Spark Executors' Memory Allocation

The Spark Executors' memory allocation is calculated based on two parameters inside `$SPARK_HOME/conf/spark-defaults.conf`: *spark.executor.memory*: sets the base memory used in calculation

spark.yarn.executor.memoryOverhead: is added to the base memory. It defaults to 7% of base memory, with a minimum of 384MB

NOTE: Make sure that Executor requested memory, including overhead memory, is below the YARN container maximum size, otherwise the Spark application won't initialize.

Example: for *spark.executor.memory* of 1Gb , the required memory is 1024+384=1408MB. For 512MB, the required memory will be 512+384=896MB

To set executor memory to 512MB, edit `$SPARK_HOME/conf/spark-defaults.conf` and add the following line:

```
spark.executor.memory          512m
```

Now you're all set and almost ready to start! If you haven't done it already, just go back to Chapter 5.2, 6.2 and forward to Chapter 7.1 and follow the steps required to set up web inferfaces. Or just go ahead to Chapter 8 and run your first test application!

In the end this is how our `spark-default.conf` looks like:

```
spark.master yarn
spark.yarn.am.memory 512m
spark.driver.memory 5g
spark.yarn.executor.memoryOverhead 384m
spark.submit.deployMode client
spark.eventLog.enabled  true
spark.eventLog.dir hdfs://master:9000/spark-logs
spark.history.provider org.apache.spark.deploy.history.FsHistoryProvider
spark.history.fs.logDirectory hdfs://master:9000/spark-logs
spark.history.fs.update.interval  10s
spark.history.ui.port 18080
spark.eventLog.permissions=777
spark.eventLog.compress  true
```

!!!IMPORTANT!!!: You don’t have to duplicate spark directory or configuration to all node in the cluster, yarn will take care of that once you’ll run a spark application.

7.1 History Server

When you submit a job, Spark Driver automatically starts a web UI on port 4040 that displays information about the application. However, when execution is finished, the Web UI is dismissed with the application driver and can no longer be accessed.

Spark provides a History Server that collects application logs from HDFS and displays them in a persistent web UI. The following steps will enable log persistence in HDFS:

Edit `$SPARK_HOME/conf/spark-defaults.conf` and add the following lines to enable Spark jobs to log in HDFS:

```
spark.eventLog.enabled  true
spark.eventLog.dir hdfs://master:9000/spark-logs
```

Create the log directory in HDFS with this command (more on Chapter 8):

```
$ hdfs dfs -mkdir /spark-logs
```

Configure History Server related properties in `$SPARK_HOME/conf/spark-defaults.conf:`

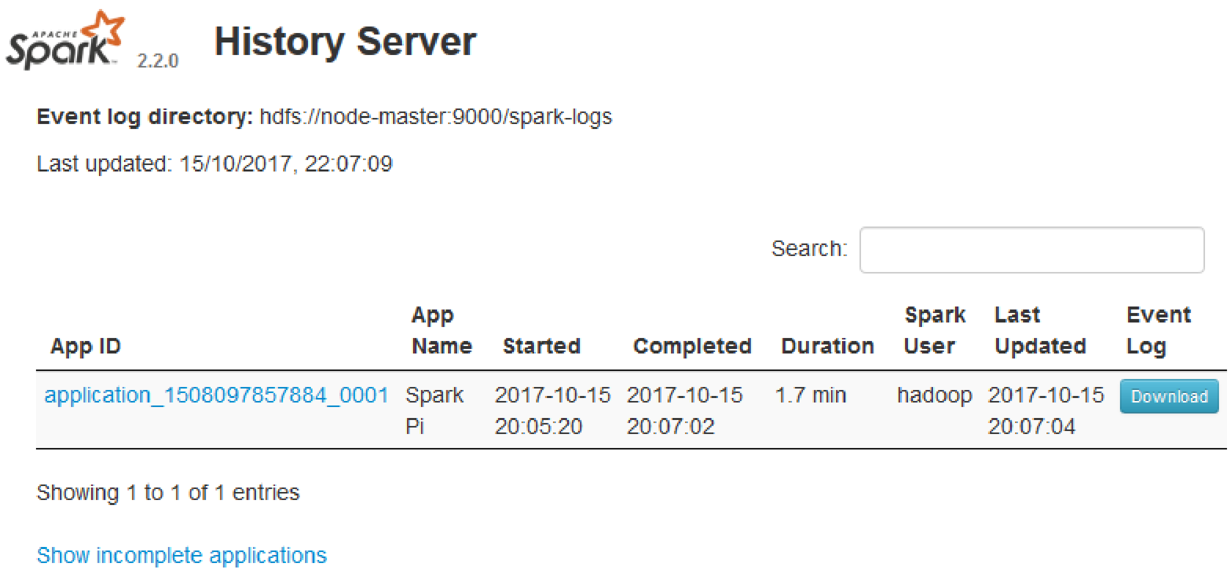
```
spark.history.provider            org.apache.spark.deploy.history.FsHistoryProvider
spark.history.fs.logDirectory     hdfs://master:9000/spark-logs
spark.history.fs.update.interval  10s
spark.history.ui.port             18080
```

You may want to use a different update interval than the default 10s. If you specify a bigger interval, you will have some delay between what you see in the History Server and the real time status of your application. If you use a shorter interval, you will increase I/O on the HDFS.
Run the History Server:

```
$ $SPARK_HOME/sbin/start-history-server.sh
```

Repeat steps from Chapter 8 to start a job with spark-submit that will generate some logs in the HDFS:
Access the History Server by navigating to `http://master:18080` in a web browser:
If you can't reach the page there could be two possible problems:

1) You haven't started a Spark job or there was an error on start-up, try and follow the steps in Chapter 8 after you complete installation and configuration process.
2) Your port was blocked from the server firewall, don't worry go back at Chapter 3.1 to figure out how to open ports and follow those steps for port 18080.



History Server

Event log directory: hdfs://node-master:9000/spark-logs

Last updated: 15/10/2017, 22:07:09

App ID	App Name	Started	Completed	Duration	Spark User	Last Updated	Event Log
application_1508097857884_0001	Spark Pi	2017-10-15 20:05:20	2017-10-15 20:07:02	1.7 min	hadoop	2017-10-15 20:07:04	Download

Showing 1 to 1 of 1 entries

Show incomplete applications

8. Running your first test application

Just like we did with Hadoop run the commands in this guide from master unless otherwise specified. Be sure you have a user (in our case “spark”) that can access all cluster nodes with SSH keys without a password. Note the path of your Hadoop installation. This guide assumes it is installed in /hadoop. If it is not, adjust the path in the examples accordingly.

Format HDFS

HDFS needs to be formatted like any classical file system. On master, run the following command as spark user:

```
$ hdfs namenode -format
```

Your Hadoop installation and hdfs is now configured and ready to run.

Run and monitor HDFS

This section will walk through starting HDFS on NameNode and DataNodes, and monitoring that everything is properly working and interacting with HDFS data.

Start and Stop HDFS

Start the HDFS by running the following script from master:

```
$ start-dfs.sh
```

It’ll start NameNode and SecondaryNameNode on master, and DataNode on slave, according to the configuration in the workers config file.

Check that every process is running with the jps command on each node. You should get on node-master (PID will be different):

```
21922 Jps

21603 NameNode

21787 SecondaryNameNode
```

and on slave:

```
19728 DataNode

19819 Jps
```

To stop HDFS on master and slave nodes, run the following command from master:

```
$ stop-dfs.sh
```

Monitor your HDFS Cluster

You can get useful information about running your HDFS cluster with the hdfs dfsadmin command. Try for example:

```
$ hdfs dfsadmin -report
```

This will print information (e.g., capacity and usage) for all running DataNodes. To get the description of all available commands, type:

```
$ hdfs dfsadmin -help
```

Put and Get Data to HDFS

Writing and reading to HDFS is done with command hdfs dfs. First, manually create your home directory. All other commands will use a path relative to this default home directory:

```
$ hdfs dfs -mkdir -p /user/spark
```

The option -p is used to create all the directories expressed in the given path.

Let’s use some textbooks from our first application.

Create a books directory in HDFS. The following command will create it in the home directory, /user/spark/books:

```
$ hdfs dfs -mkdir books
```

Grab a few books from the Gutenberg project:

```
$ cd /home/spark

$ wget -O alice.txt https://www.gutenberg.org/files/11/11-0.txt

$ wget -O holmes.txt https://www.gutenberg.org/ebooks/1661.txt.utf-8

$ wget -O frankenstein.txt https://www.gutenberg.org/ebooks/84.txt.utf-8
```

Put the three books through HDFS, in the books directory:

```
$ hdfs dfs -put alice.txt holmes.txt frankenstein.txt books
```

NOTE: if for some unknown reason Hadoop doesn’t recognize books directory try to write the full path and it will work.

List the contents of the book directory:

```
$ hdfs dfs -ls books
```

Move one of the books to the local filesystem:

```
$ hdfs dfs -cat books/alice.txt
```

There are many commands to manage your HDFS. For a complete list, you can look at the Apache HDFS shell documentation, or print help with:

```
$ hdfs dfs -help
```

Run YARN

HDFS is a distributed storage system, it **doesn’t** provide any services for running and scheduling tasks in the cluster. This is the role of the YARN framework. The following section is about starting, monitoring, and submitting jobs to YARN.

Start YARN with the script:

```
$ start-yarn.sh
```

Check that everything is running with the jps command. In addition to the previous HDFS daemon, you should see a ResourceManager on master, and a NodeManager on slave.

To stop YARN, run the following command on master:

```
$ stop-yarn.sh
```

Monitor YARN

The yarn command provides utilities to manage your YARN cluster. You can also print a report of running nodes with the command:

```
$ yarn node -list
```

Similarly, you can get a list of running applications with command:

```
$ yarn application -list
```

To get all available parameters of the yarn command, see Apache YARN documentation.

Submit MapReduce Jobs to YARN

Yarn jobs are packaged into jar files and submitted to YARN for execution with the command yarn jar. The Hadoop installation package provides sample applications that can be run to test your cluster. You’ll use them to run a word count on the three books previously uploaded to HDFS.

Submit a job with the sample jar to YARN. On master, run:

```
Found 2 items
-rw-r--r--   1 hadoop supergroup          0 2017-10-11 14:09 output/_SUCCESS
-rw-r--r--   1 hadoop supergroup     269158 2017-10-11 14:09 output/part-r-00000
```

Print the result with:

```
$ hdfs dfs -cat output/part-r-00000
```

In addition you can also try to run this example:

```
$ yarn jar /hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.0.3.jar
wordmean "holmes.txt" output
```

How to Submit a Spark Application to the YARN Cluster

Applications are submitted with the spark-submit command. The Spark installation package contains sample applications, like the parallel calculation of Pi, that you can run to practice starting Spark jobs.

To run the sample Pi calculation, use the following command:

```
$ spark-submit --class org.apache.spark.examples.SparkPi $SPARK_HOME/examples/jars/spark-
examples_ 2.11-2.3.1.jar 10
```

You can add --deploy-mode as a parameter to specifies which mode to use, client or cluster. To run the same application in cluster mode, replace `--deploy-mode client` with `--deploy-mode cluster`.

In addition you can also try a simple wordcount with this command:

```
$ spark-submit --class org.apache.spark.examples.JavaWordCount
$SPARK_HOME/examples/jars/spark-examples_2.11-2.3.1.jar books/holmes.txt
```

NOTE: Since the output of this application it's pretty long, you can add "> output.txt" at the end of the command above to redirect all output to a file in your /home/spark directory.

Run the Spark Shell

The Spark shell provides an interactive way to examine and work with your data.

Put some data into HDFS for analysis. This example uses the text of Alice In Wonderland from the Gutenberg project:

```
$ cd /home/spark
$ wget -O alice.txt https://www.gutenberg.org/files/11/11-0.txt
$ hdfs dfs -mkdir inputs
$ hdfs dfs -put alice.txt inputs
```

Start the Spark shell:

9. Setting Multi-User Access with POSIX

Required Steps: Chapter 1/3.1/3.2/4/5/6/7/8/

This Chapter was taken from the book: Expert Hadoop Administration: Managing, Tuning, and Securing Spark, YARN, and HDFS

Managing HDFS Permissions and Users

HDFS as a file system is somewhat similar to the POSIX file system in terms of the file permissions it requires. However, HDFS doesn't have the concept of users and groups as in the other file systems. It's important to understand the nature of the HDFS super user and how to manage the granting of permissions to users. You also need to learn how to set up users so they're ready to read data and write to the HDFS file system.

HDFS File Permissions

In a Linux system, you create OS users and make them members of an existing operating system group. In Hadoop, you associate a directory with an owner and a group. You need not actually "create" either the users or the groups. Rather, you use the concept of users and groups to set file and directory permissions. The following sections show how file and directory permissions work in HDFS.

HDFS Permission Checking

The HDFS configuration parameter *dfs.permissions.enabled* in the hdfs-site.xml file determines whether permission checking is enabled in HDFS:

```
<property>

<name>dfs.permissions.enabled</name>

<value>true</value>

</property>
```

The default value of the parameter is true, meaning permission checking is enabled. If you set this parameter to false, you turn HDFS permission checking off. Obviously, you can do this in a development environment to overcome frequent permission-related error messages, but in a production cluster, you need to keep it at its default setting.

HDFS File and Directory Permissions

HDFS uses a symbolic notation (r, w) to denote the read and write permissions, just as a Linux operating system does.

When a client accesses a directory, if the client is the same as the directory's owner, Hadoop tests the owner's permissions.

If the group matches the directory's group, then Hadoop tests the user's group permissions. If neither the owner nor the group names match, Hadoop tests the "other" permission of the directory. If none of the permissions checks succeed, the client's request is denied.

Although there's an execute (x) permission for a file, it's ignored for files, and as far as directories go, the execute permission implies that you can access the subdirectories of that directory. Unlike in the underlying Linux operating system, Hadoop has nothing like the UIDs (User IDs) or GIDs (Group IDs) to identify users and groups. HDFS simply stores users and groups of a directory or file as strings.

A user can write to an HDFS directory only if that user has the correct permissions. In this example, the Linux root user tries to copy a file to a user's HDFS directory and fails due to lack of permissions.

```
[root@master]# hdfs dfs -put test.txt /user/antani/test2222/

put: Permission denied: user=root, access=WRITE, inode="/user/antani/

test2222":hdfs:supergroup:drwxr-xr-x

[root@hadoop01]#

Permission Denied Errors in HDFS
```

You may receive the permission denied error when you're issuing an HDFS command from the command line, as in the previous example, or even when you're trying to browse the HDFS file system through the NameNode web page. For example, you may receive the following error when you try to browse files through the web UI.

```
Permission denied:user=antani,access=READ_EXECUTE,inode="/user":hadoop:hdfs:drwx.---
---
```

In this case, you need to change the access privileges on the HDFS directory /user, after logging in as the user hdfs, from the command line:

```
$ hdfs dfs -chmod -R 755 /user
```

Running administrative commands as the root user or any other non-privileged (from the perspective of Hadoop) user will result in errors. If you run the Hadoop file system checking command fsck as the root user, you'll get the following error:

```
$ su root

$ hdfs fsck /

...

FSCK ended at Sun May 29 14:46:27 CDT 2016 in 39473 milliseconds

Permission
denied:user=root,access=READ_EXECUTE,inode="/lost+found/user":hdfs:supergroup:drwxr-
-r--

Fsck on path '/' FAILED

#
```

The FAILED result you get from running the fsck command here doesn't mean the file system is corrupt! It simply means that you failed to execute the fsck command. A similar thing happens when you run the dfsadmin –report command as any user other than the HDFS super user, hdfs:

```
$ hdfs dfsadmin –report

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

report: Access denied for user root. Superuser privilege is required

#
```

In both the cases described here, the right thing to do is to either log in as the user hdfs and execute the commands, or if you have the sudo privileges to the hdfs user account, run the commands as follows:

```
$ sudo -u spark hdfs fsck /
$ sudo -u hdfs hdfs dfsadmin -report
```

NOTE: Unlike the regular Linux or UNIX permissions mode, Access Control Lists (ACLs) let you define permissions for some of a group’s members. For example, you can grant or deny write permissions on a file only to specific users or groups. ACLs are disabled by default, but you can enable them by configuring the NameNode appropriately with the dfs.namenode.acls.enabled configuration parameter.

HDFS Users and Super Users

Typically, database administrators create users in their databases, with each user having specific privileges and/or roles that enable them to perform various actions in the database. In the context of Hadoop, creating a user is kind of a misnomer, as HDFS really doesn’t have anything that lets you create user identities as you would on Linux systems. It also doesn’t enable you to create any groups.

In the default mode of authentication, called simple authentication, Hadoop relies on the underlying operating system to determine client identities. If you set up a Kerberized system (a system that has been set up to authenticate connections through Kerberos), then Kerberos will determine the client identities.

Note that you don’t need to create an operating system account on the underlying Linux system for your HDFS users to be able to access and use HDFS. It’s a good practice to create OS accounts for all Hadoop users who’ll be using the local file system on the gateway servers for their Hadoop-related work.

Creating HDFS (and Hadoop) Users

In order to enable new users to use your Hadoop cluster, follow these general steps.

Create an OS account on the Linux system from which you want to let a user execute Hadoop jobs. Before creating the user, you may have to create the group as well:

```
$ group add studenti

$ useradd –g studenti antani

$ passwd antani
```

Here, studenti is an OS group we’ve created for a set of users. The passwd command lets me set a password for the user.

Make sure that you’ve set the permissions on the Hadoop temp directory you’ve specified in the core-site.xml file, so all Hadoop users can access it:

```
<property>

  <name>hadoop.tmp.dir</name>

  <value>/tmp/hadoop-$(user.name)</value>

</property>
```

If the file permissions on the HDFS temp directory aren’t 777, make them so:

```
$ hdfs -dfs –chmod –R 777 /tmp/

$ hdfs -dfs –chmod –R 777 /spark-logs/
```

In order to “create” a new HDFS user, you need to create a directory under the /user directory. This directory will serve as the HDFS “home” directory for the user.

```
$ hdfs dfs -mkdir /user/antani
```

By default, when you create a directory or a file, the owner is the user that creates the directory (or file) and the group is the group of that user, as shown here.

```
# sudo -u spark

# hdfs dfs -ls /user

Found 135 items

drwxr-xr-x   - spark      supergroup      0 2016-05-28 08:18 /user/antani

....
```

In this case, I used the spark account to create the directory, so the owner is spark and the group is supergroup. Change the ownership of the directory, since you don’t want to use the default owner/group (spark/supergroup) for this directory.

```
$ su spark

$ hdfs dfs –chown –R antani:studenti

$ hdfs dfs –ls /user/

$ drwxr-xr-x   - antani   studenti      0 2016-04-27 12:40 /user/antani
```

You can check the new directory structure for the user with the following command:

```
$ hdfs dfs –ls /user/alapati
```

User antani can now store the output of his MapReduce and other jobs under that user’s home directory in HDFS.

Refresh the user and group mappings to let the NameNode know about the new user:

```
$ hdfs dfsadmin - refreshUserToGroupsMappings
```

Optional: You can set a space quota for the new directory you’ve created:

```
$ hdfs dfsadmin -setSpaceQuota 30g /user/antani
```

The new user can now log into the gateway servers and execute his or her Hadoop jobs and store data in HDFS.

User Identities

Hadoop supports two modes of operation—simple and Kerberos—to determine user identities. The simple mode of operation is the default. You specify the mode of operation with the hadoop.security.authentication property in the hdfs-site.xml file.

The HDFS Super User

Since Hadoop doesn’t have the concept of a user identity, there’s no fixed super user for Hadoop. The system super user for Hadoop is simply the operating system user that starts the NameNode. The HDFS super user doesn’t have to be the root user of the NameNode host. If you wish, you can allocate a set of users to a separate super user group.

You can make a set of users members of a super user group by setting the dfs.permissions.supergroup configuration parameter in the hdfs-site.xml file, as shown here.

```
<property>
  <name>dfs.permissions.superusergroup</name>
  <value>supergroup</value>
</property>
```

In this example, supergroup is the name of the group of super users in the cluster. The following example shows that the user hdfs belongs to the group supergroup:

```
# hdfs dfs -ls /
Found 7 items
drwxr-xr-x   - spark   spark   0 2014-06-25 16:39 /data
drwxr-xr-x   - spark   supergroup         0 2015-05-05 15:46 /system
drwxrwxrwt   - spark   spark   0 2015-05-09 09:33 /tmp
drwxr-xr-x   - spark   supergroup         0 2015-05-05 13:20 /user
...
#
```

A lot of the administrative HDFS commands need to be run as the “spark” OS user, which is the default HDFS super user. If you run these commands as any other user, including the root user in a Linux system, you’ll get the following error:

```
Access denied for user root. Superuser privilege is required.
```

The root user in Linux is indeed a super user but only for the local file system. It’s user spark who’s king when it comes to the HDFS file system. You can perform administration-related HDFS commands only as the spark user or by sudoing to that user. You can use the Linux sudo command to use the privileged administrative commands, as shown in the following example.

```
$ sudo –u spark hdfs dfs –rm /user/test/test.txt
```

The End