CentOS 7 Installation Steps with Screenshots

CentOS community has released its Latest Operating System named as **CentOS 7.** Some of the **new features** in CentOS 7 as compared with CentOS 6.X are listed below :

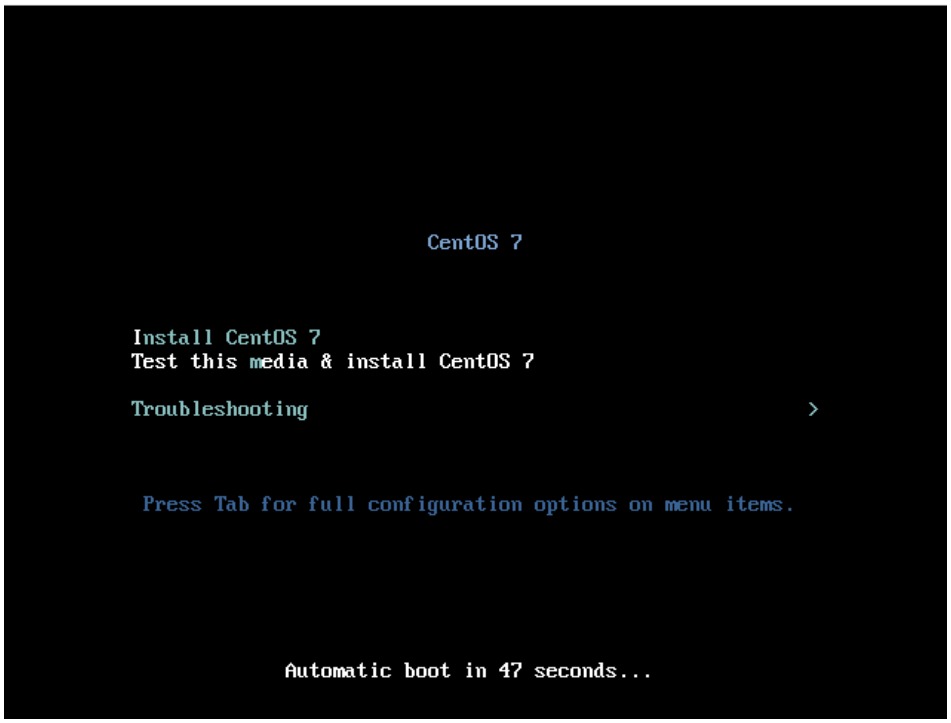
- CentOS 7 uses **XFS** as its default file system.
- **OpenJDK-7** is the default JDK.
- initd has been replaced by **systemd**.
- New Linux **Kernel 3.10.0**, support for **Linux Containers**, and the inclusion of the Open VMware Tools and 3D graphics drivers out of the box.
- Change in the numbering Scheme, Official release is **Centos 7.0-1406** , where as 7 Comes from RHEL 7 and 1406 shows release date(June 2014).

In this article we will go through the CentOS 7 Installations steps with screenshots.

Step:1 First Download the **.iso** file from the **CentOS website**, burn it onto the disc. Boot your PC from DVD.

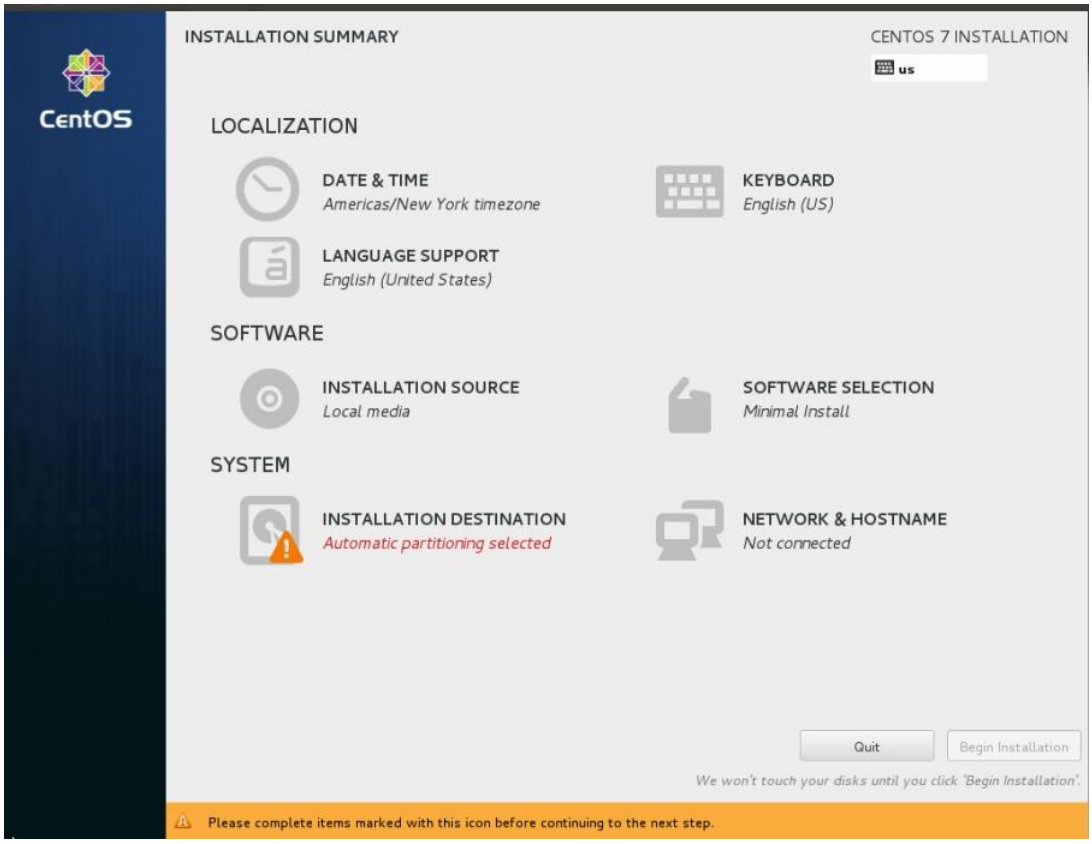
Use this link to Download CentOS 7 (64 bit)

Step:2 Choose **'Install CentOS 7'** option and press enter

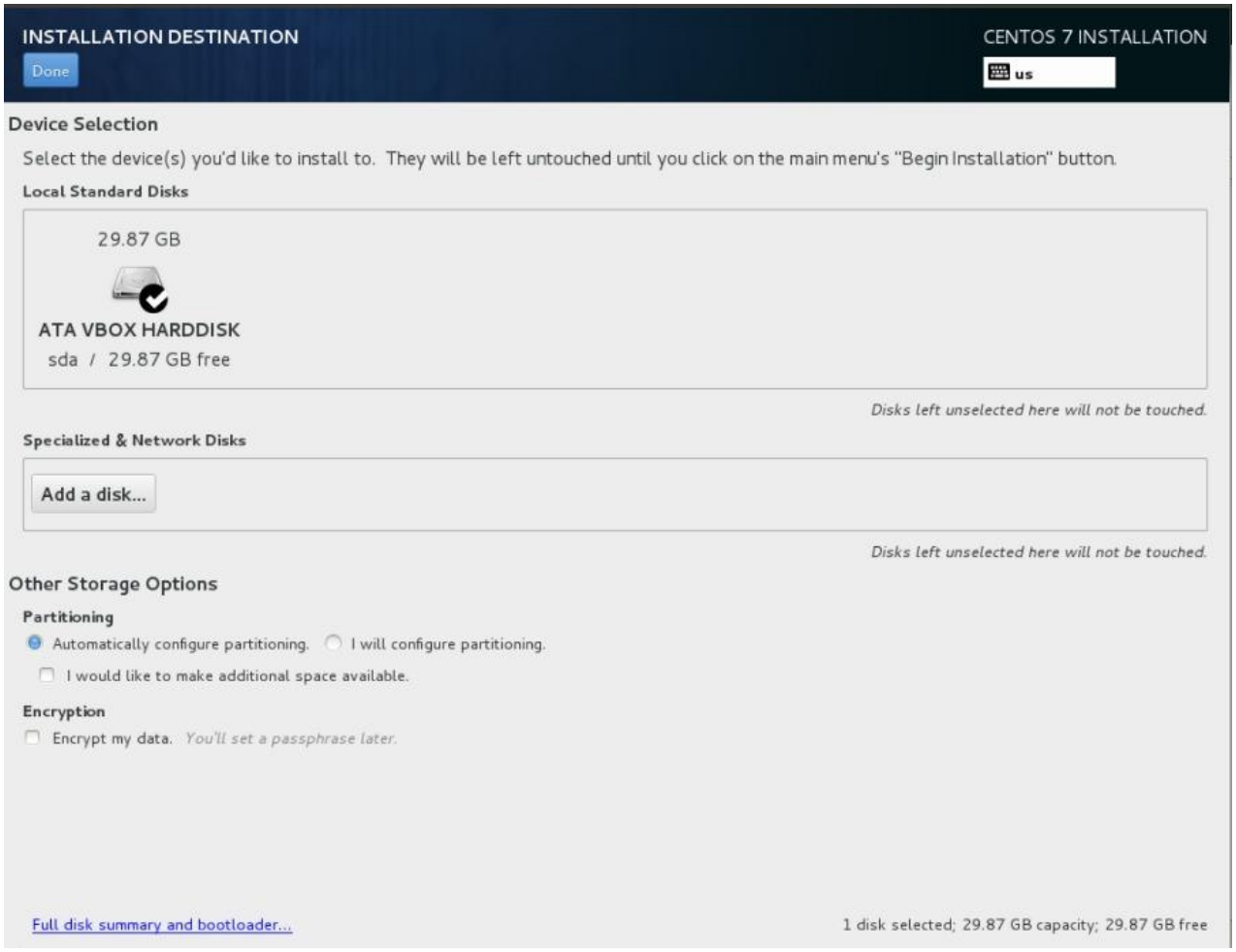


Step:3 Choose your respective Language and click on **Continue**, in my case i have choose English , as shown below

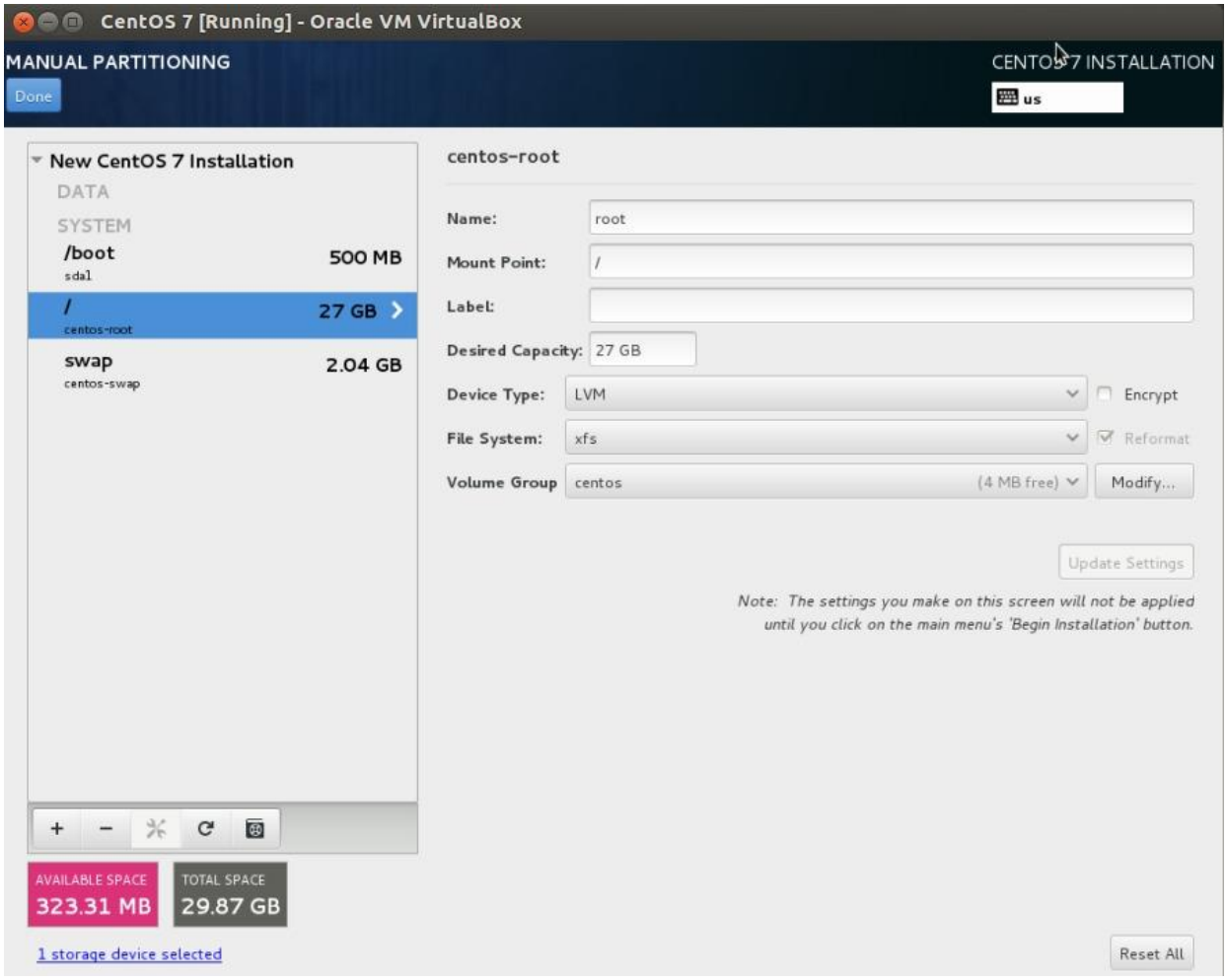
Step: 4 Change the **Installation Destination** , by default installer will do automatic partitioning for your hard disk. To create your own customize partition table click on '**Installation Destination**'.



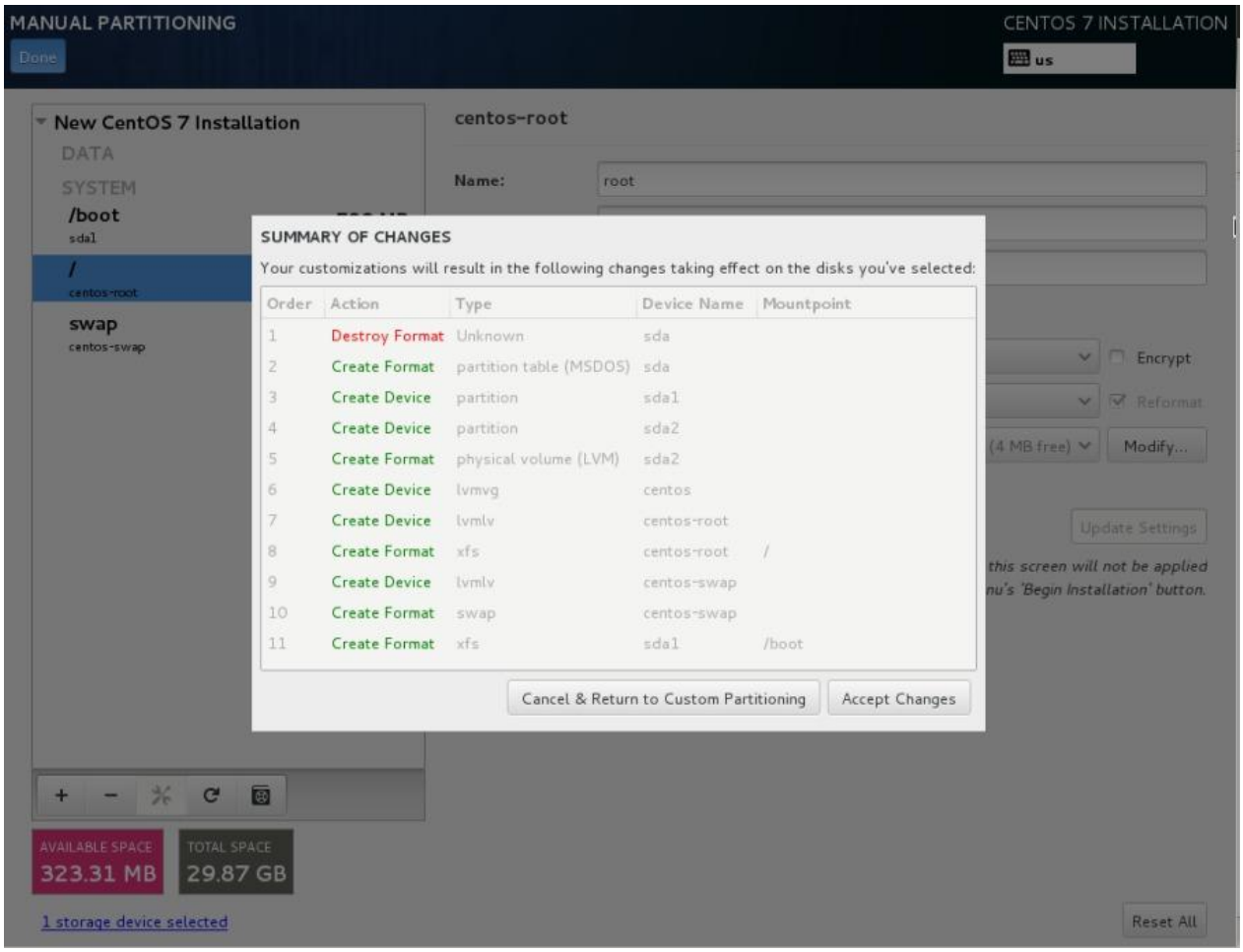
As you can see below i have around 30 GB hard drive for OS installation. choose '**I will configure partitioning**' then click **Done**



Step:5 Create the partition table , In my case i am putting everything under LVM and created **/boot , /** and **swap** parition as shown below :

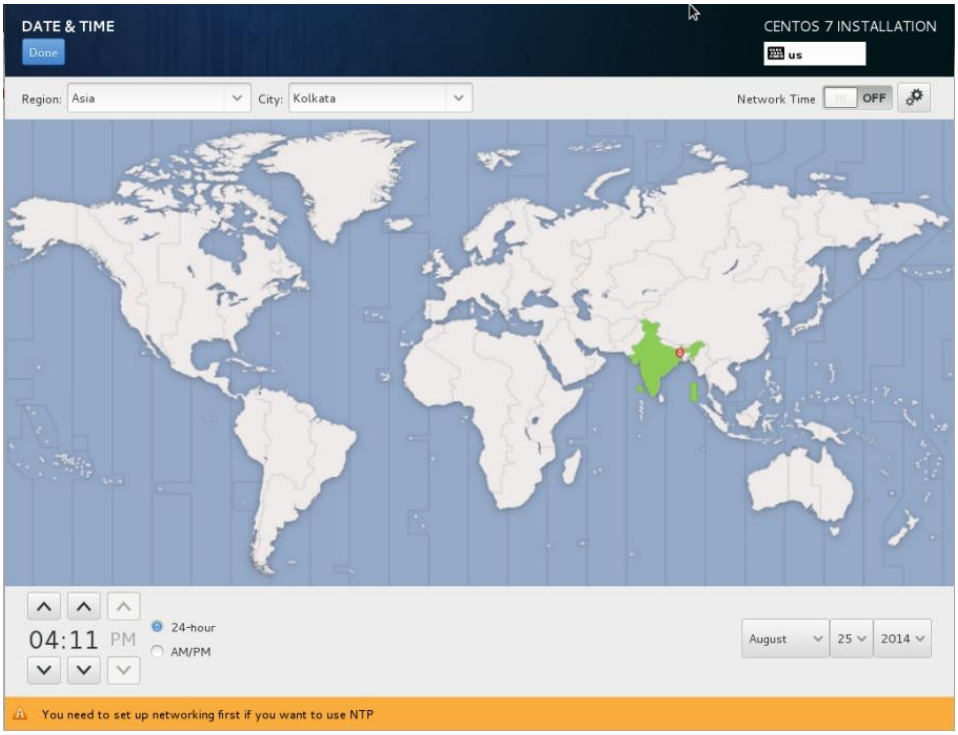


Click on Done

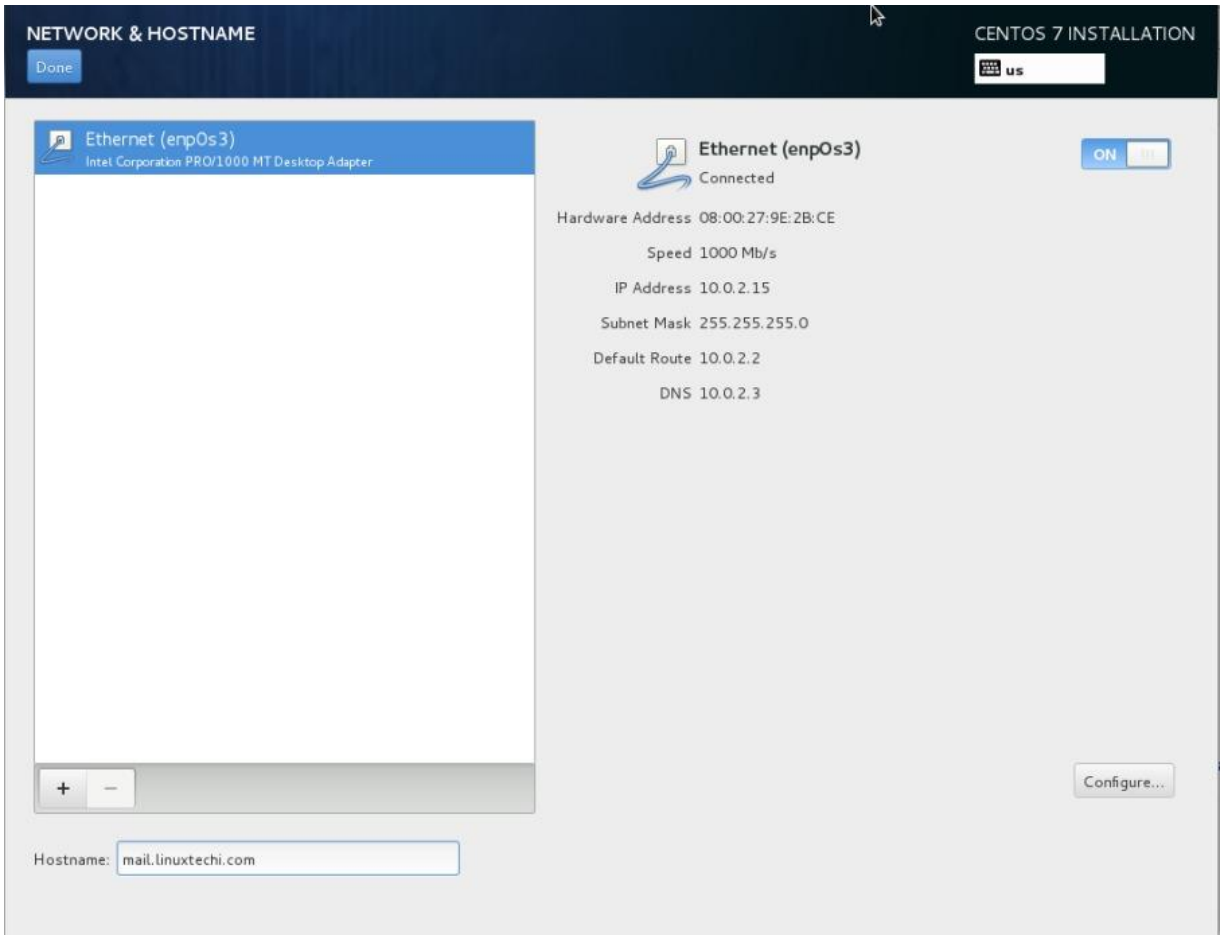


Click on Accept Changes

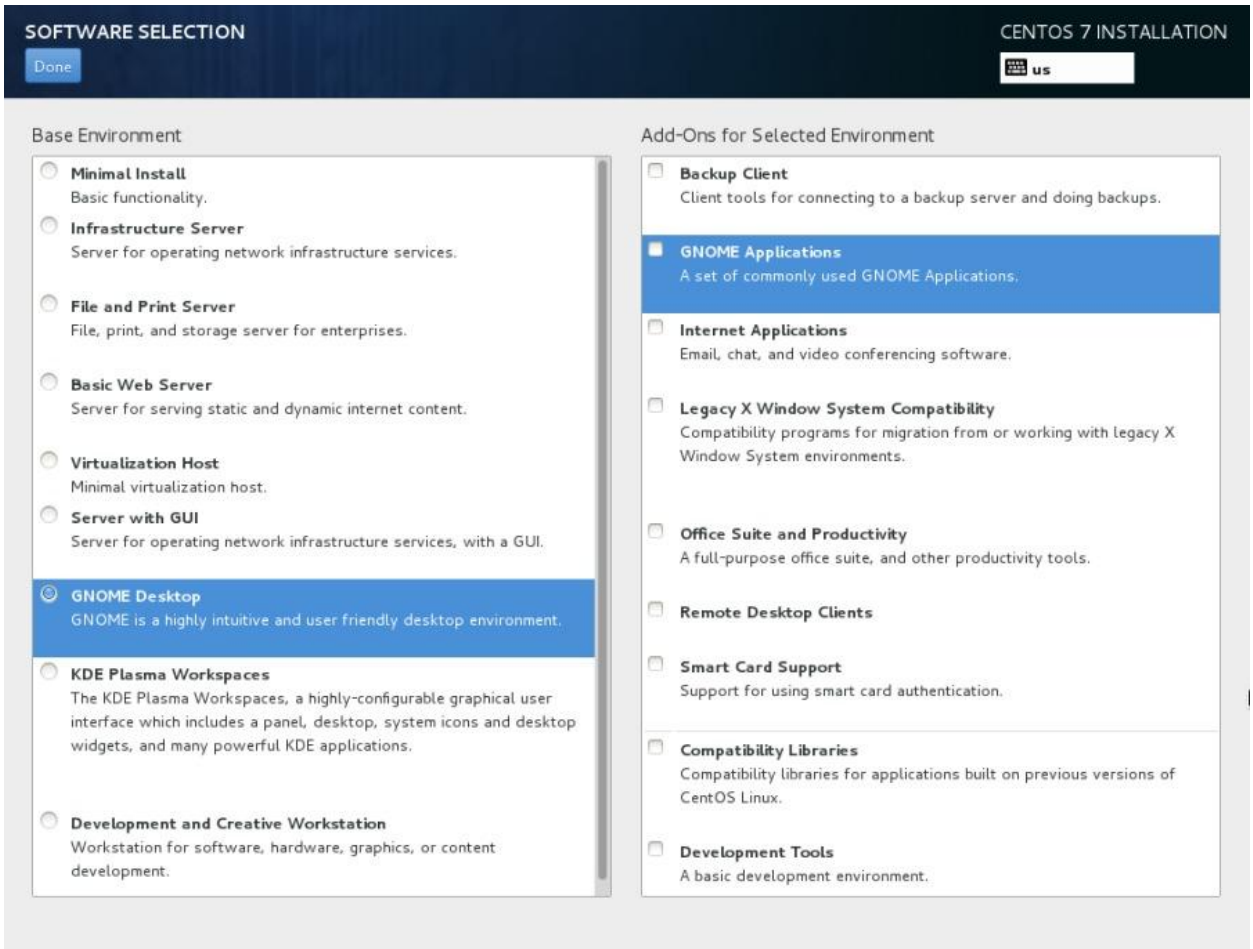
Step:6 Set Date & Time with your respective Zone



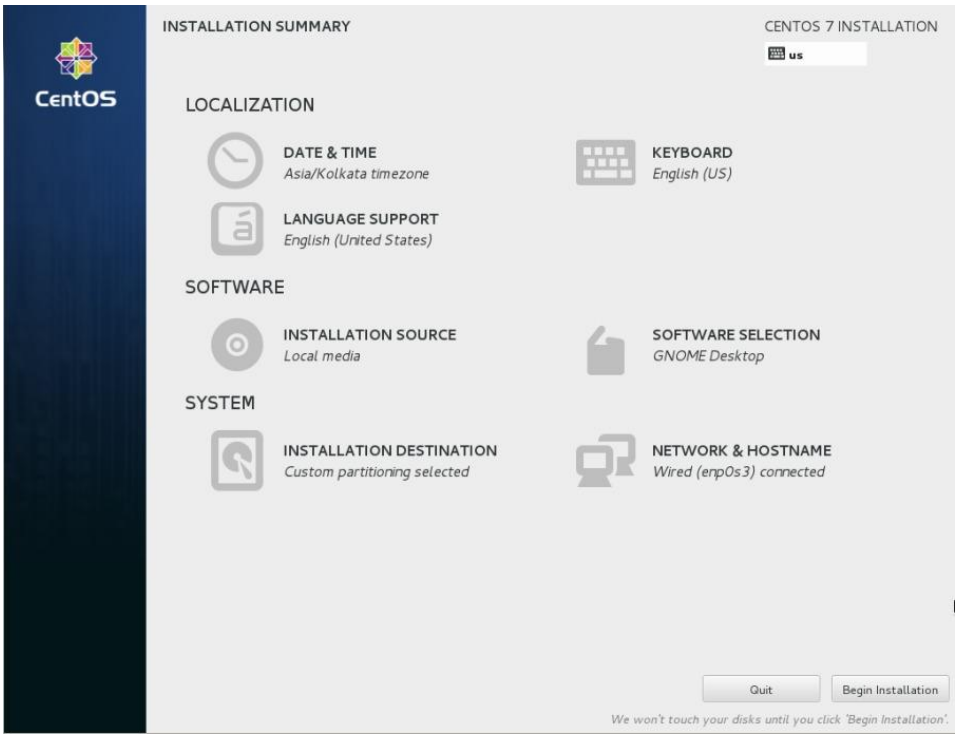
Step:7 Configure Networking and Set the hostname .



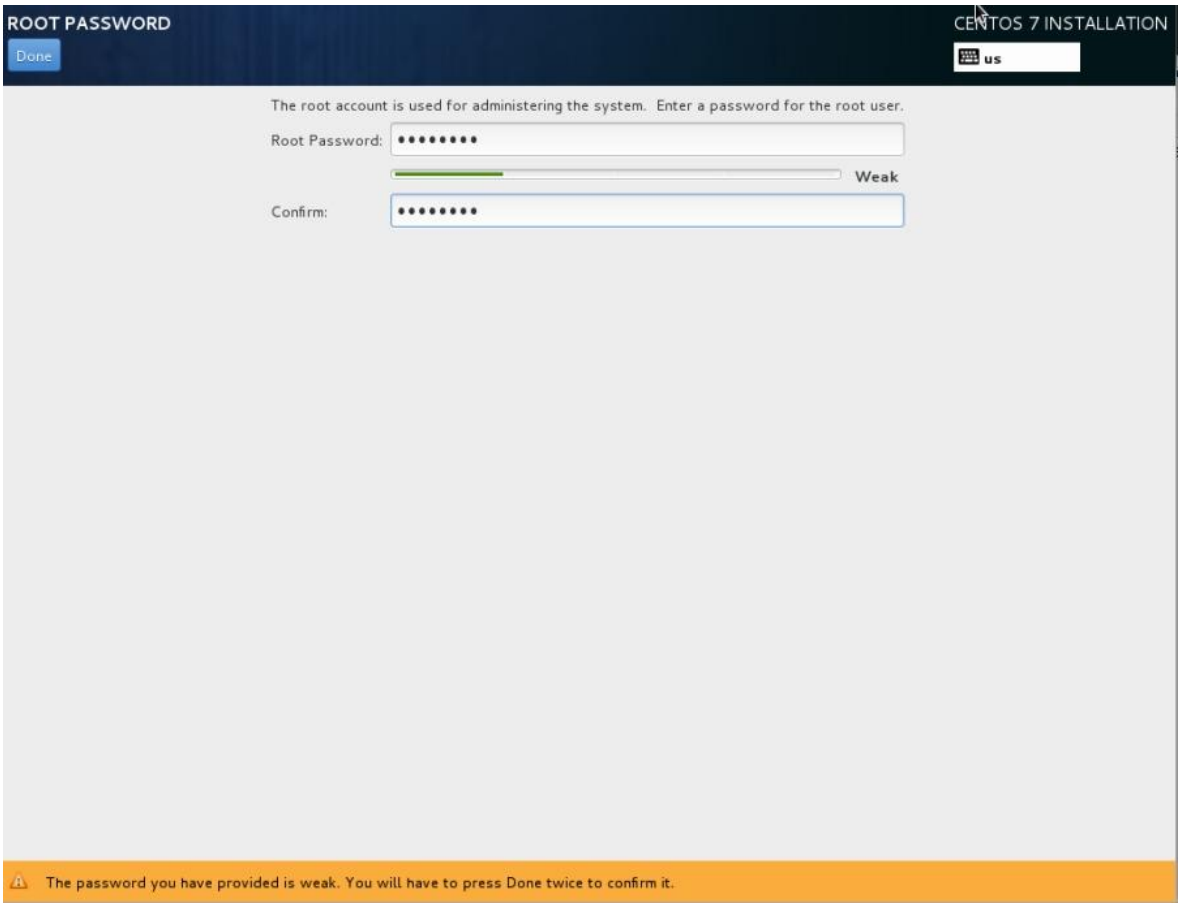
Step:8 Select the Software that you want to install. Click on "Software Selection". In my case i am selecting Gnome Desktop as shown below :



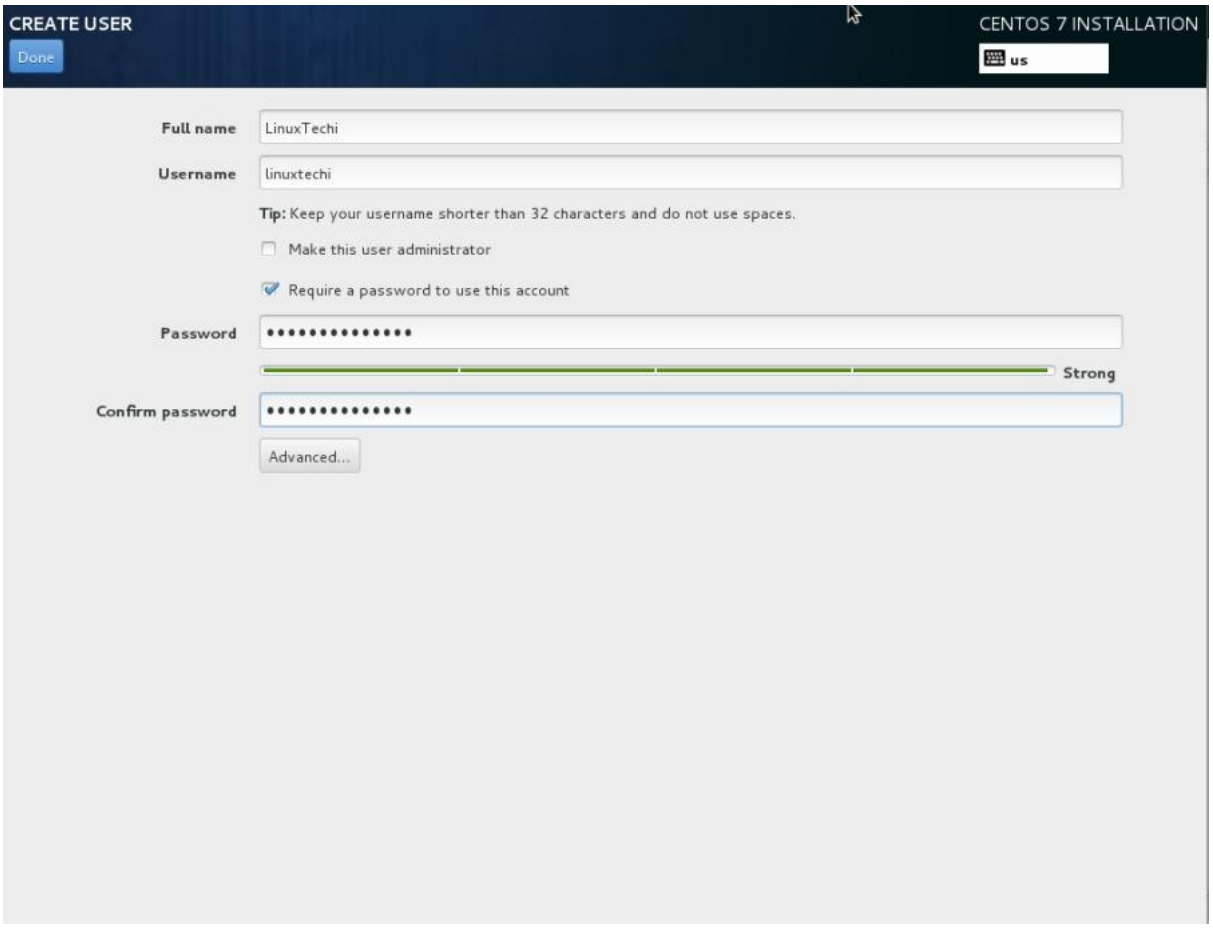
Step:9 Now Click on Begin Installation .



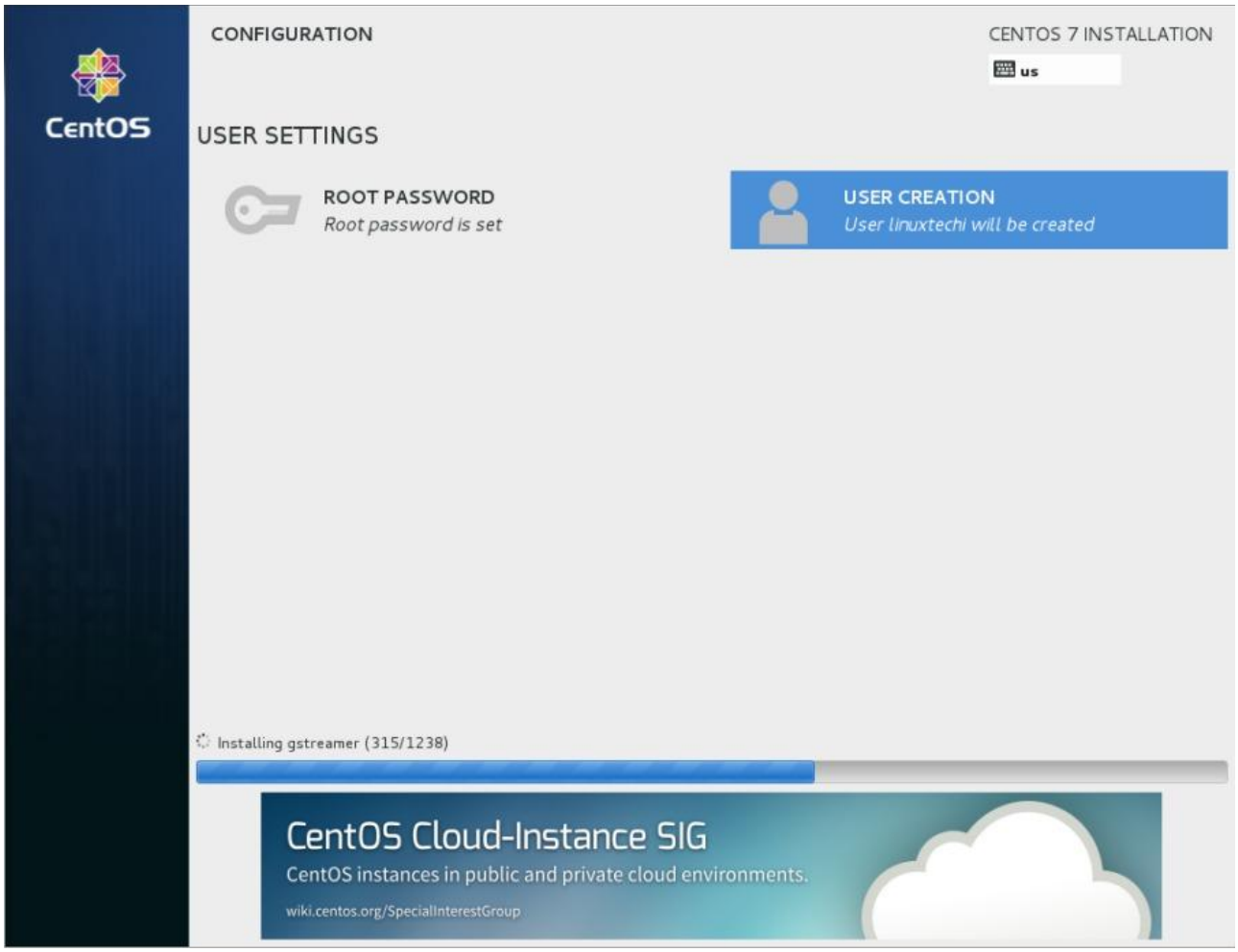
Step:10 Set root password



Step:11 Create a User

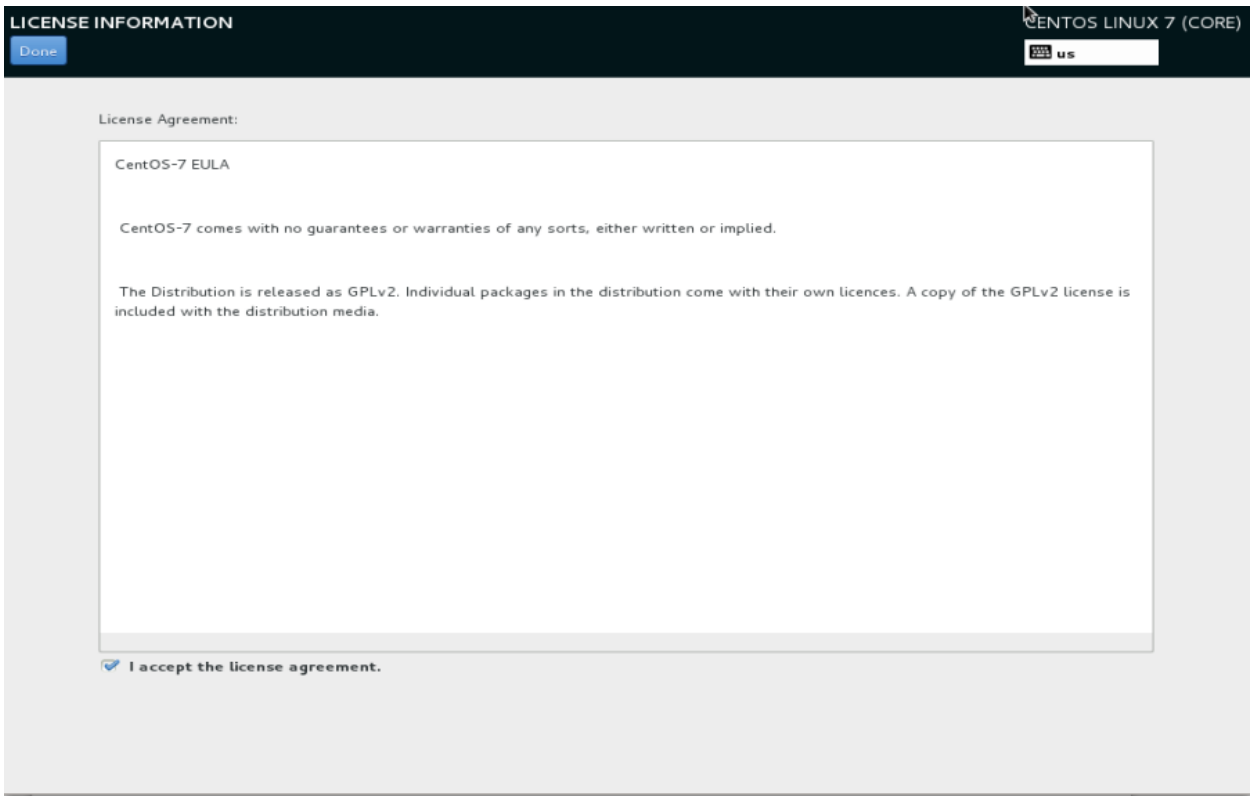


Step:12 Installation is in Progress as shown below

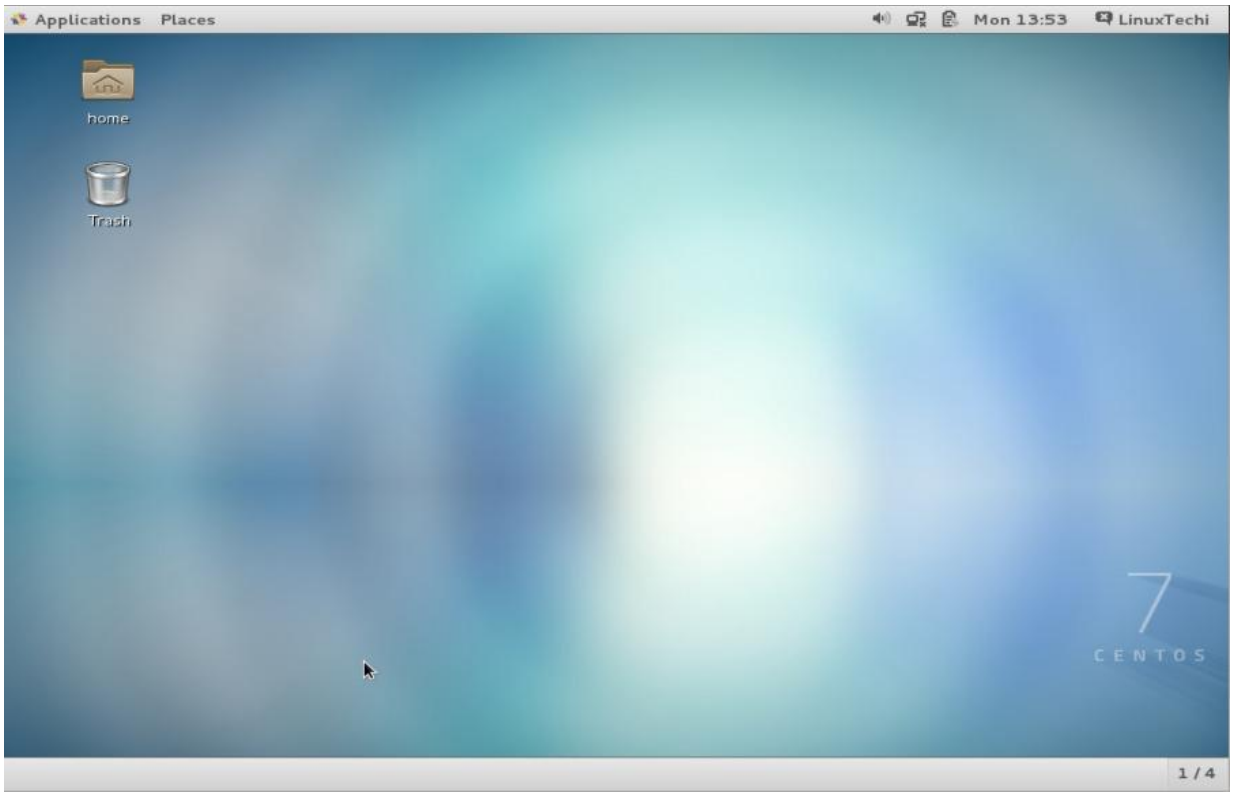


Once the installation is completed, you will be required to reboot the machine as shown below :

Step:13 When we first login to CentOS , Accept the EULA agreement



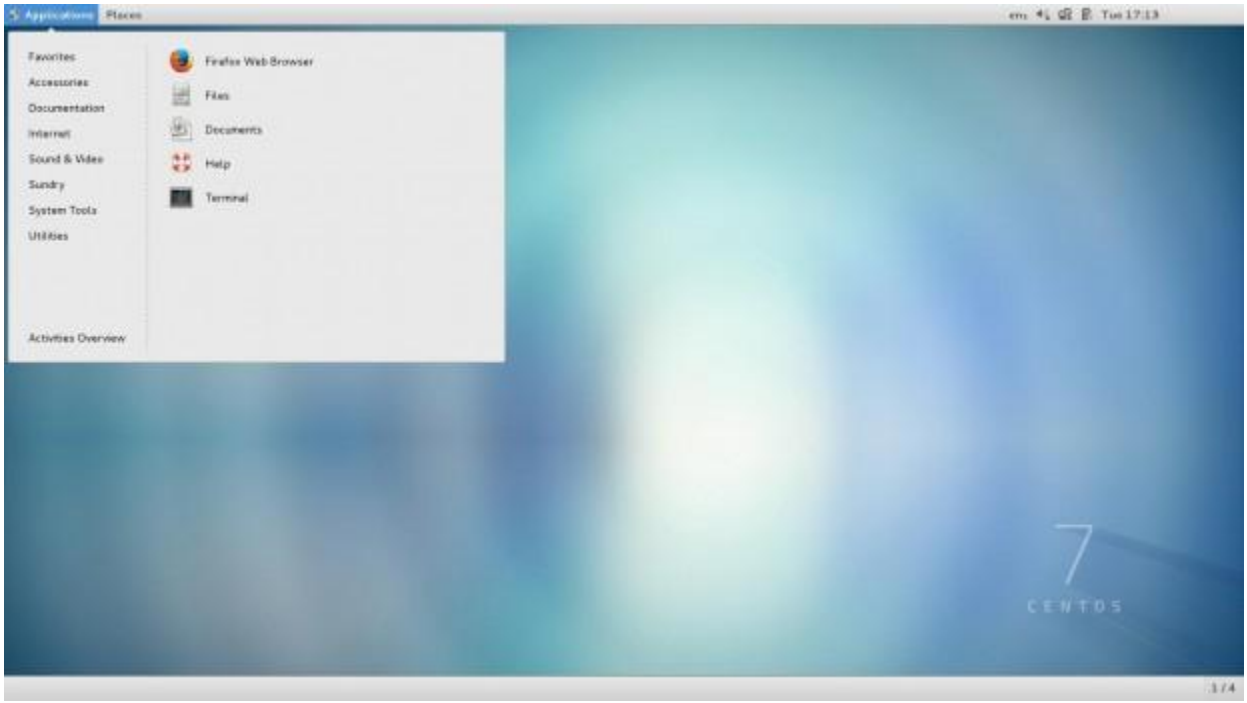
Below Screen will appear after login .



Another Example

Installation of “CentOS 7.0″ with Screenshots

This tutorial 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.



Installation of CentOS 7

If you want to find out more about what’s new in this release of **CentOS 7.0** holds and download links, I suggest reading the previous article on release announcements:

1. CentOS 7.0 Features and Download ISO Images

Requirements

1. CentOS 7.0 DVD ISO

CentOS 7.0 Installation Process

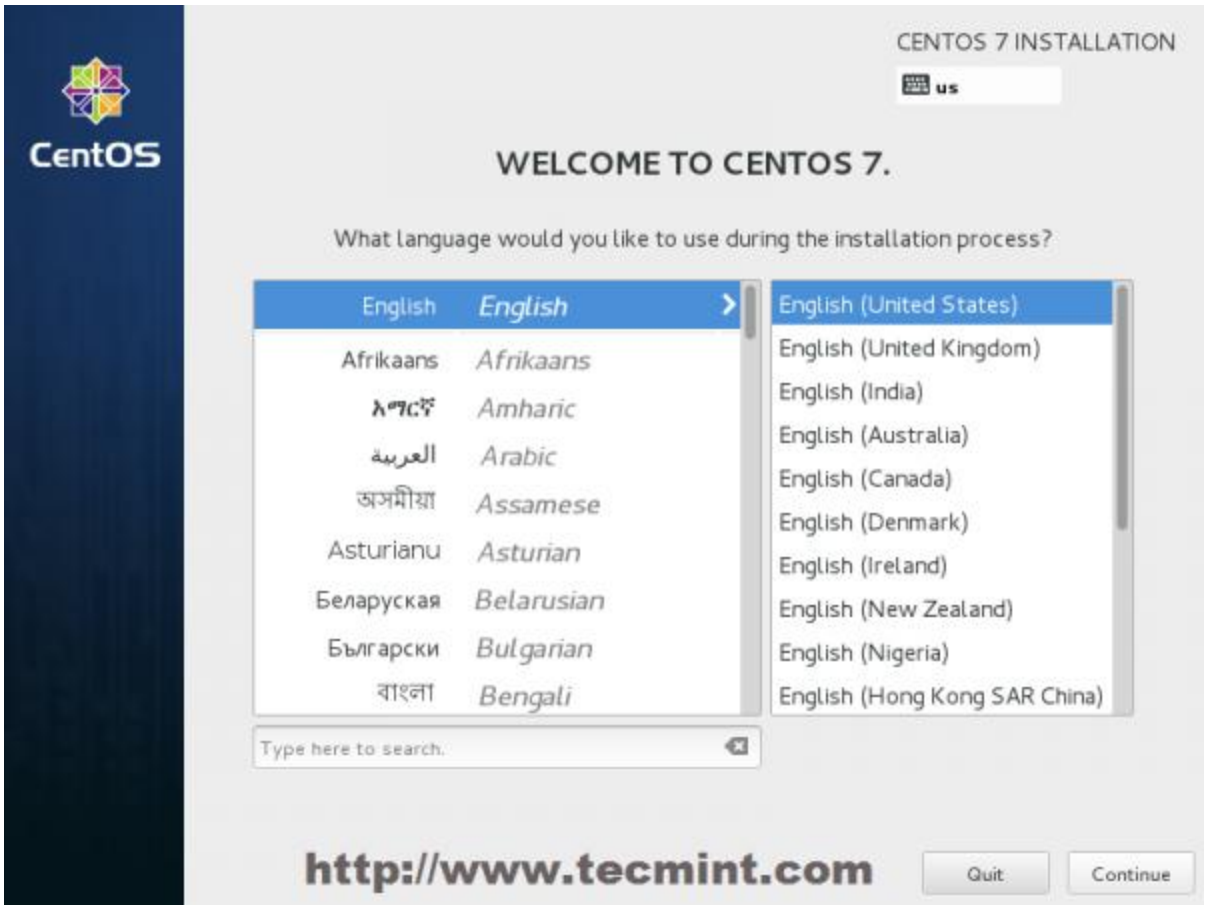
1. After downloading the last version of CentOS using above links or using official CentOS download page. Burn it to a DVD or create a bootable USB stick using **LiveUSB Creator** called Unetbootin.

2. After you have created the installer bootable media, place your DVD/USB into your system appropriate drive, start the computer, select your bootable unit and the first CentOS 7 prompt should appear. At the prompt choose **Install CentOS 7** and press [**Enter**] key.

CentOS 7 Boot Menu

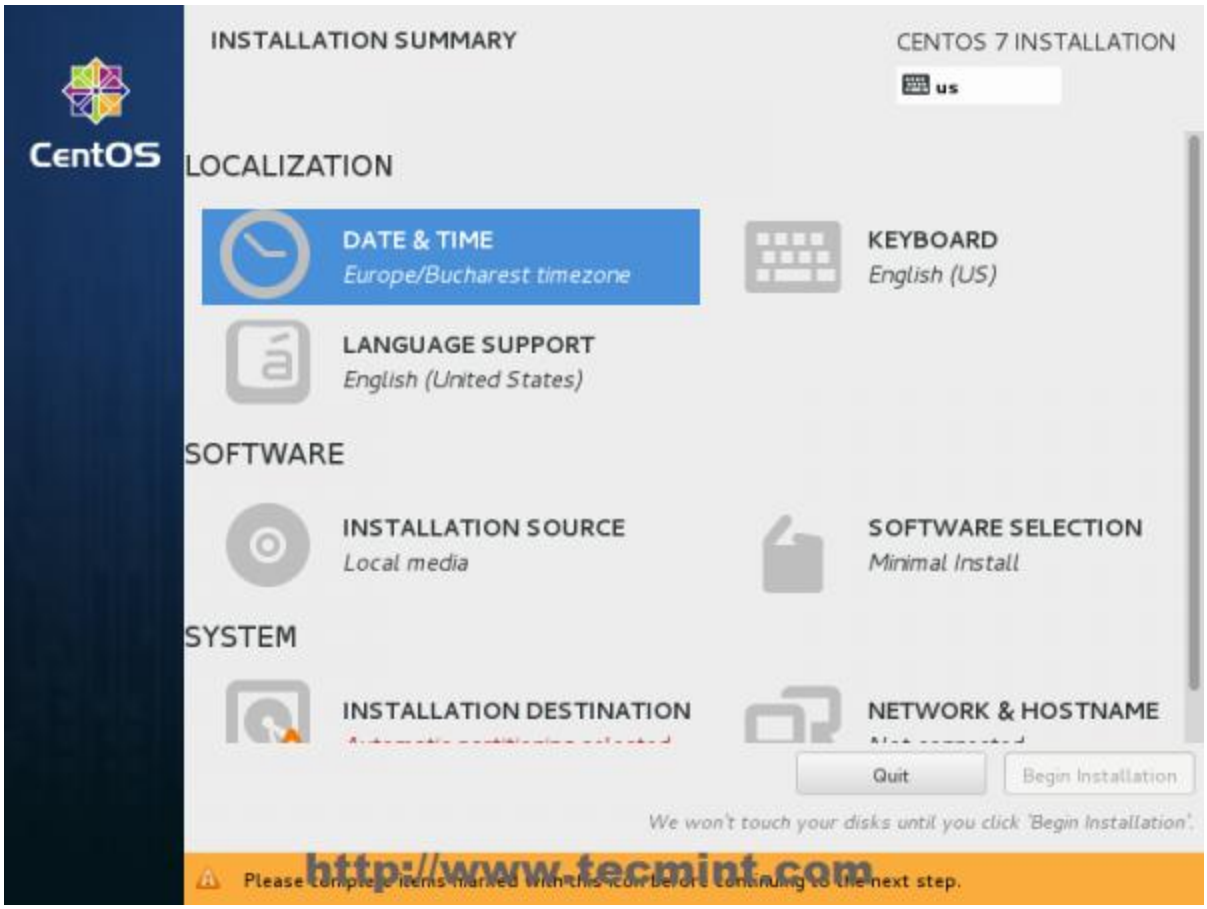
3. The system will start loading media installer and a Welcome screen should appear. Select your **Installation Process Language**, that will assist you through the entire installation procedure and click on **Continue**.

CentOS Installer Loading



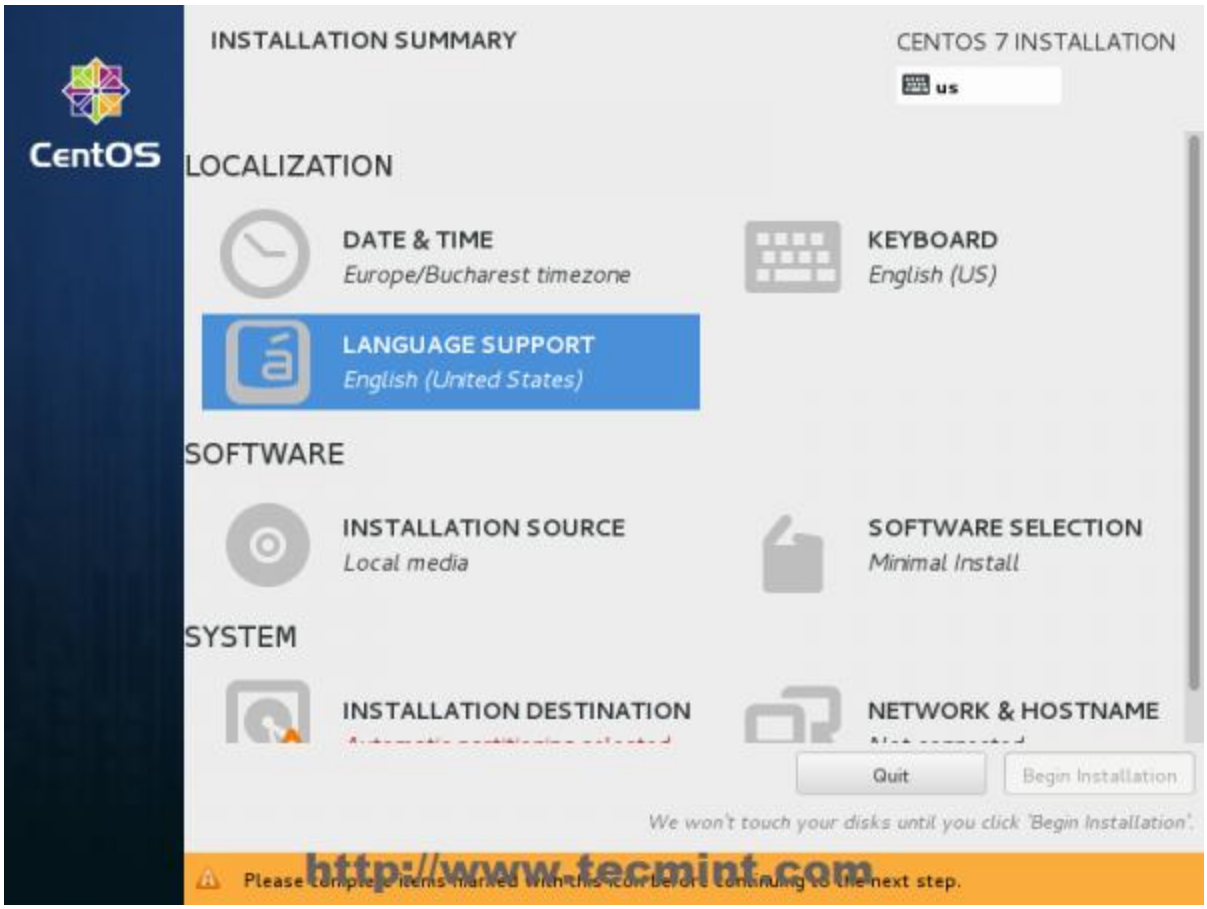
Select Installation Process Language

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.

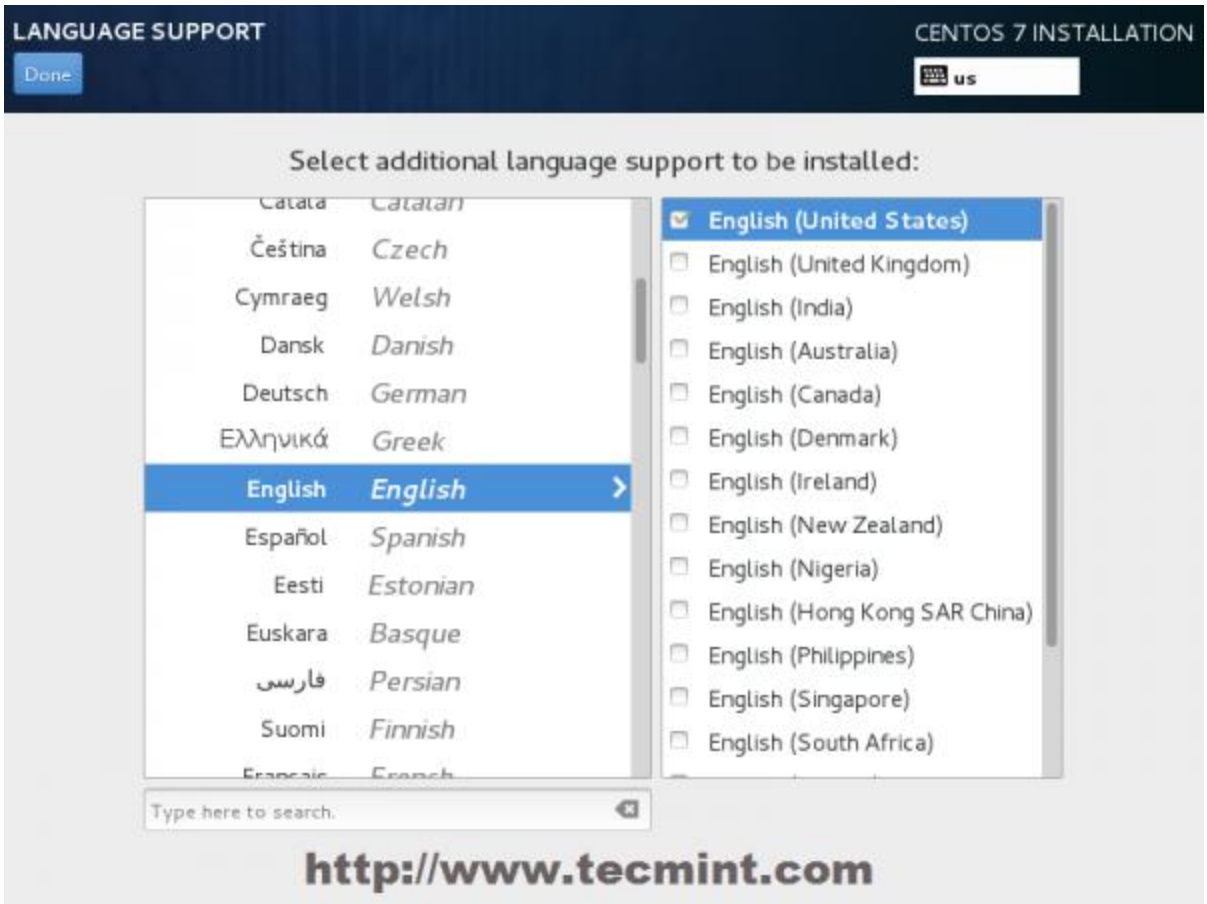


Select Date & Time and Location

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.

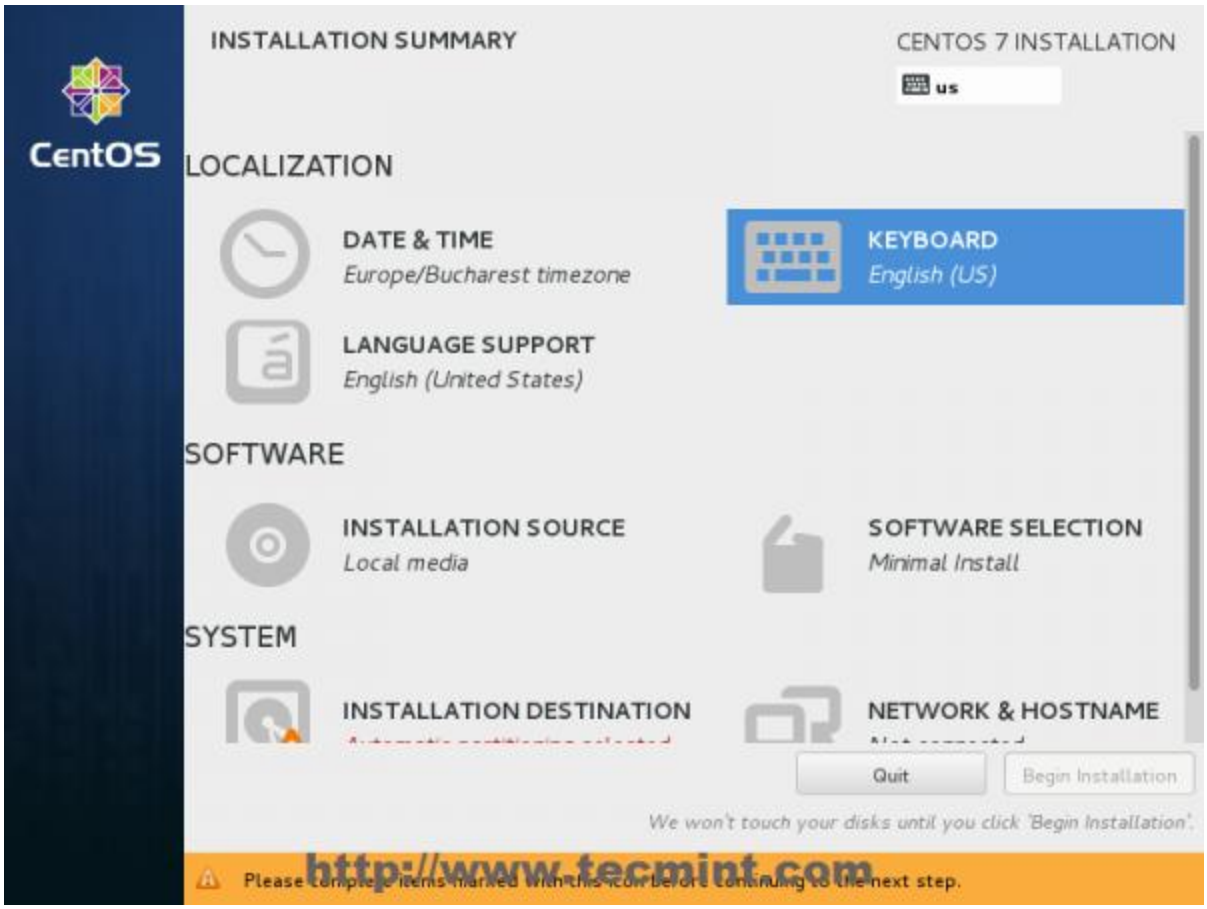


Select Language and Keyboard

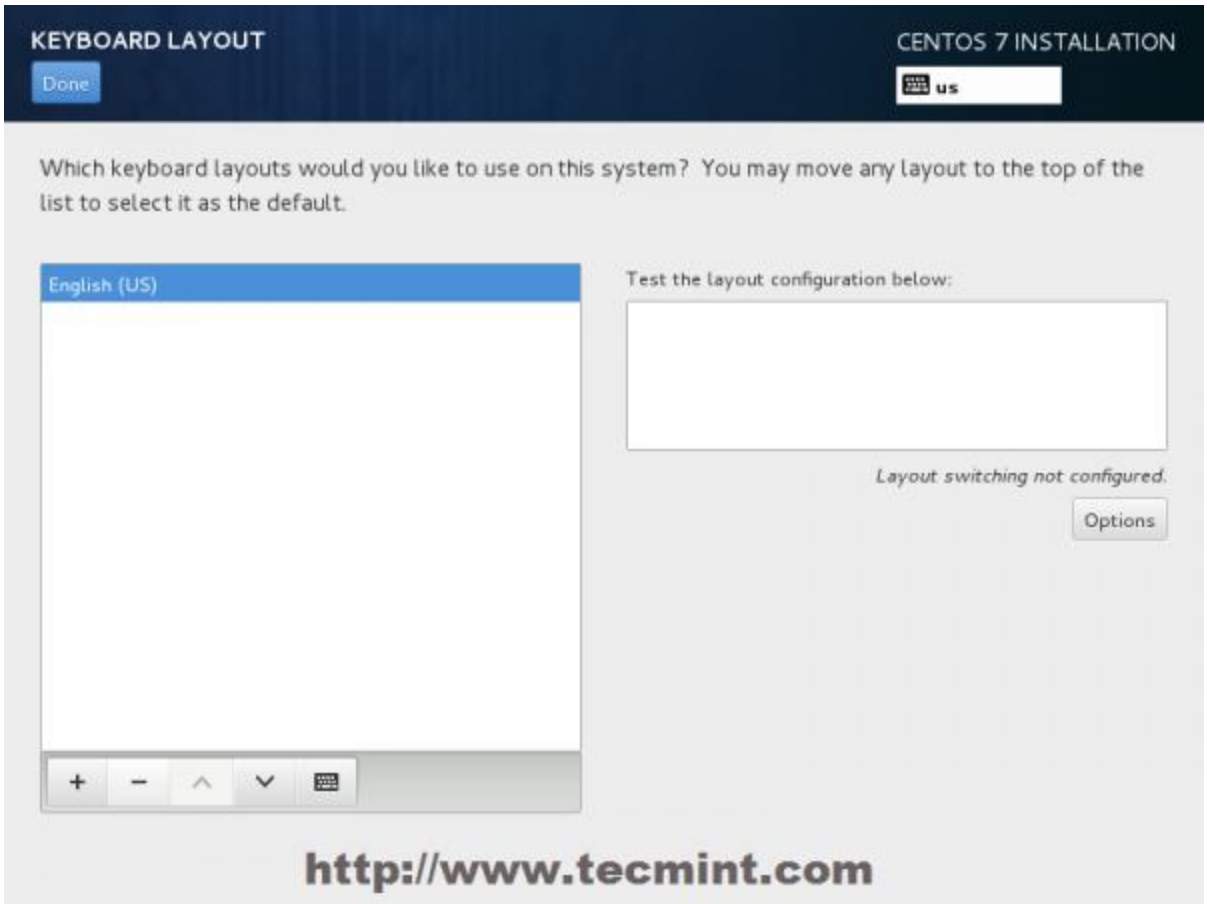


Select English Language

6. 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.



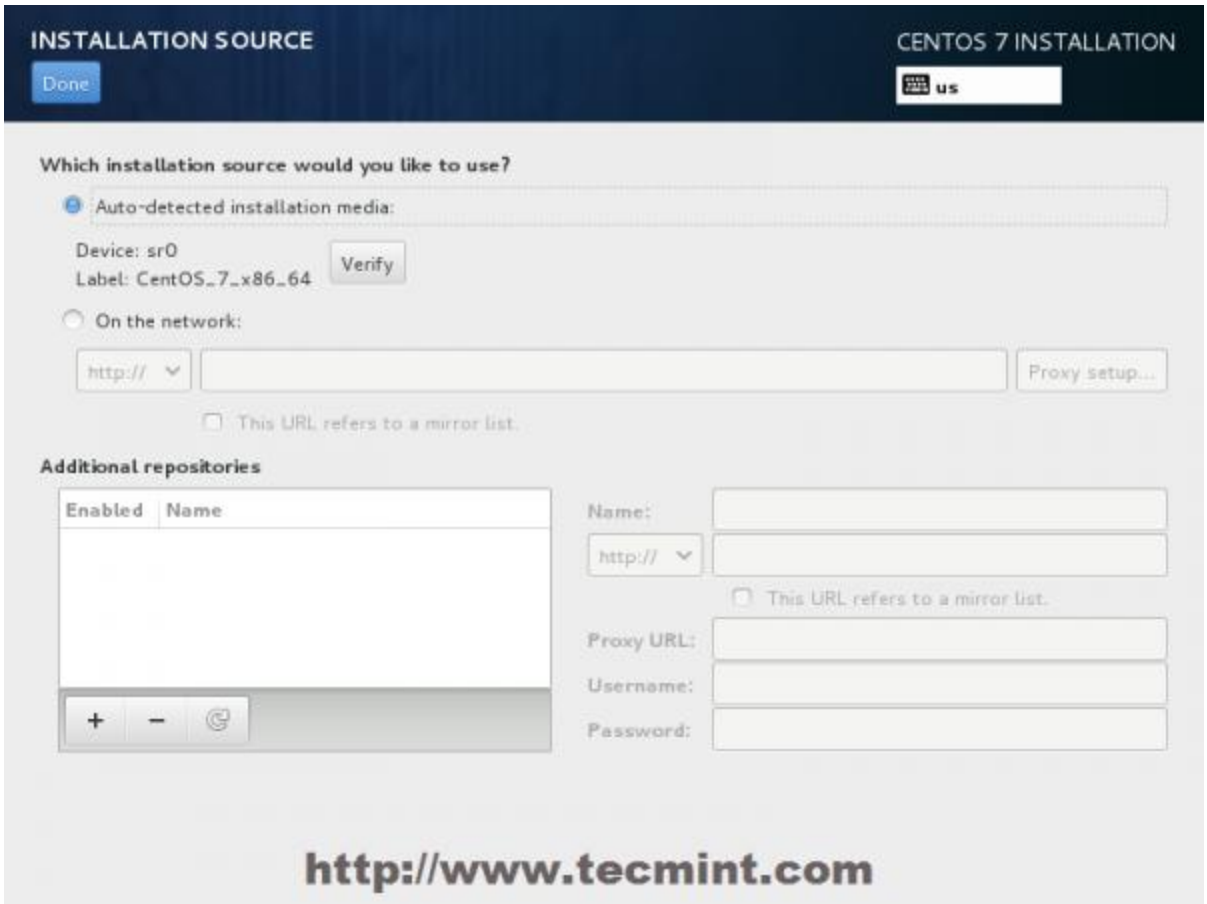
Choose Keyboard Layout



Choose English Keyboard

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.

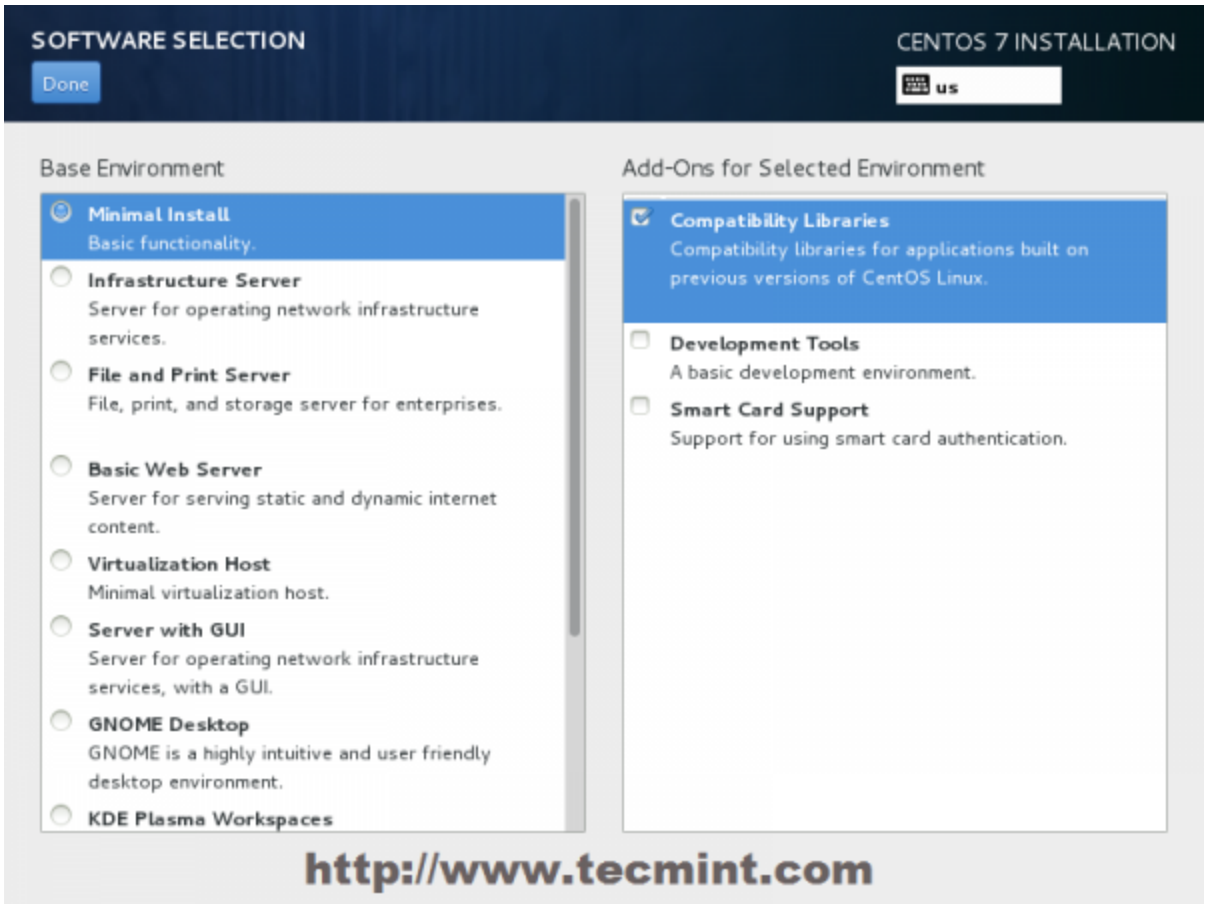
Choose Installation Sources



Auto Detect Installation Type

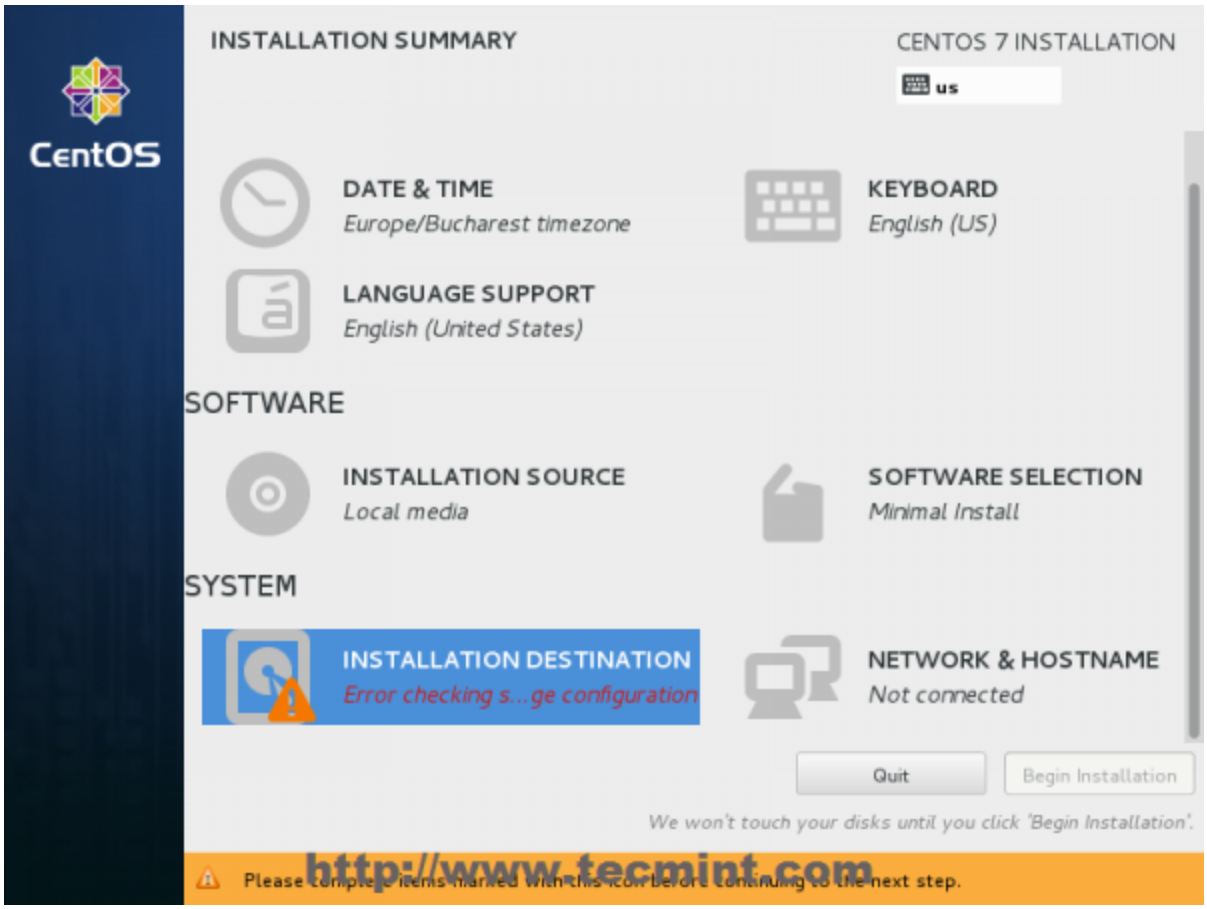
8. On the next step you can choose your system installation software. On this step CentOS offers a lot of Server and Desktop platform environments that you choose from, but, if you want a high degree of customization, especially if you are going to use CentOS 7 to run as a server platform, then I suggest 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.

Software Selection



Select CentOS 7 Minimal Install

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

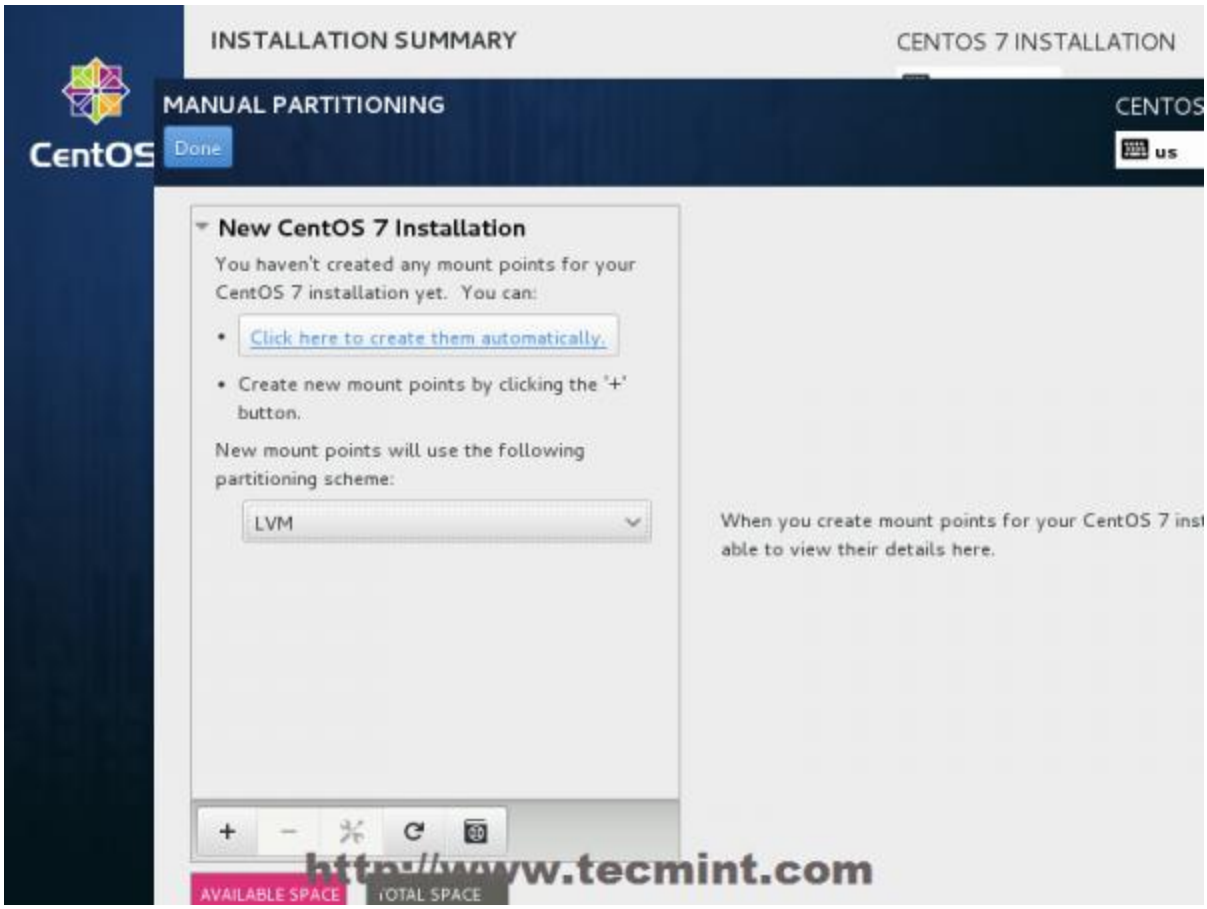


Choose Installation Destination

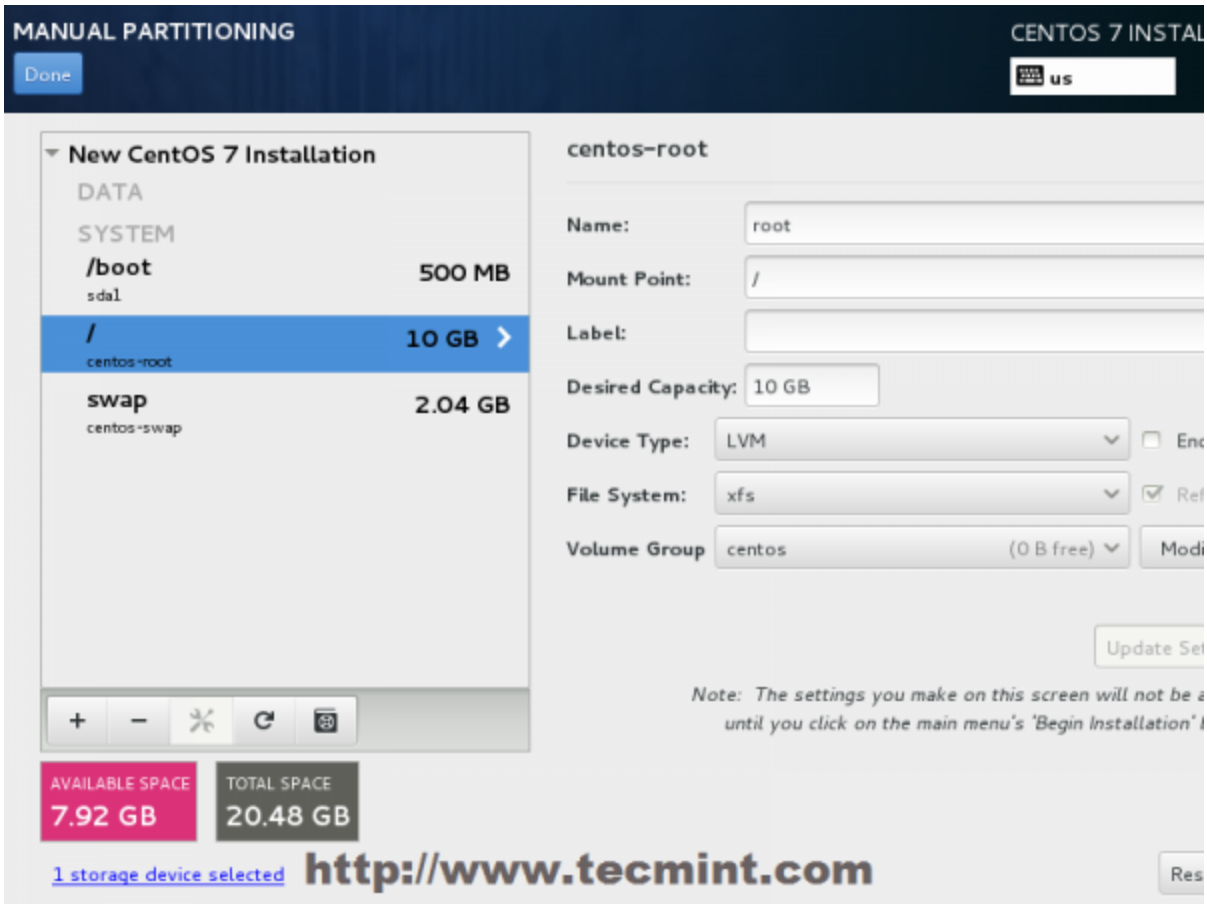
Installation Device Selection

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**.

1. **/boot** – Non LVM
2. **/(root)** – LVM
3. **Swap** – LVM

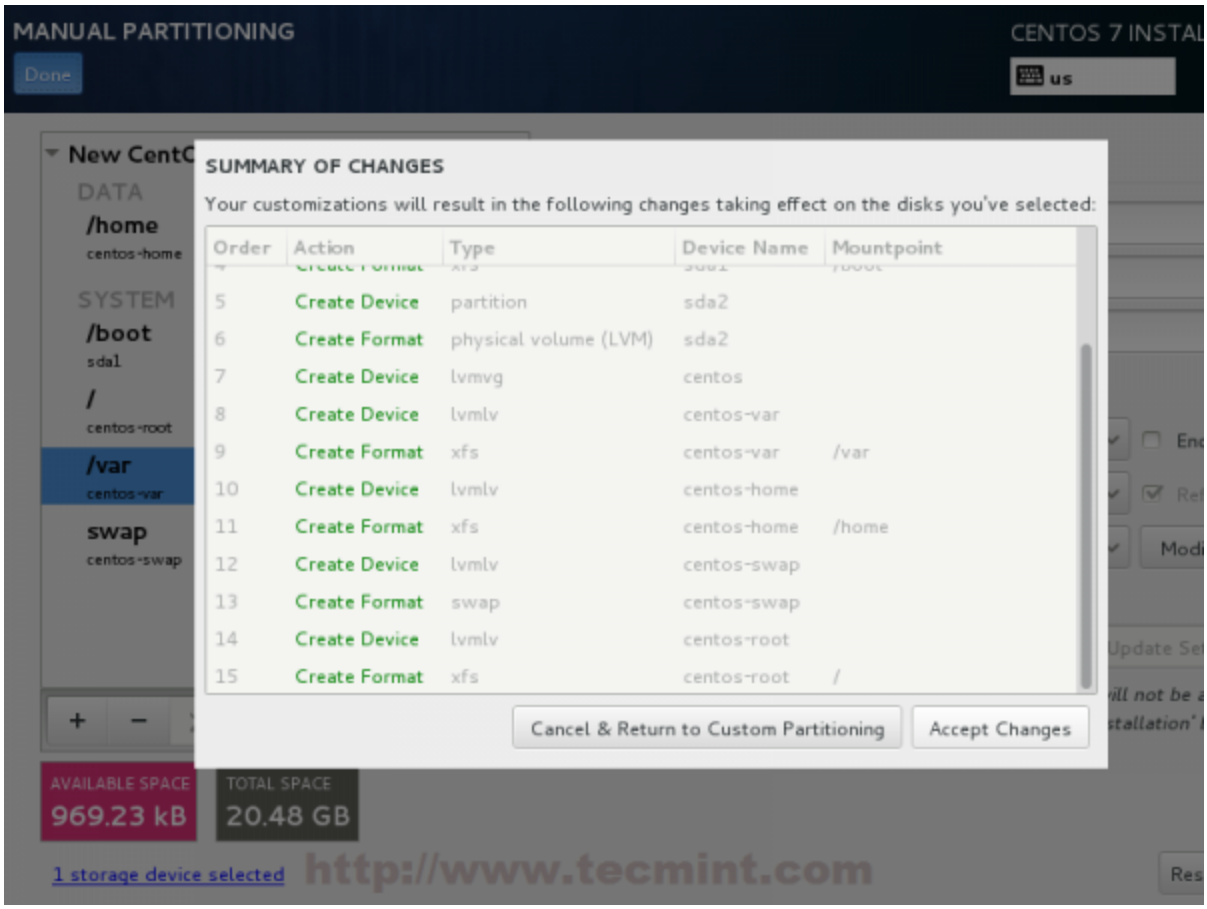


Select LVM Partition Type



Create Partitions

11. 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 and when you finish hit on **Done** button and **Accept Changes** on the Summary of Changes prompt.

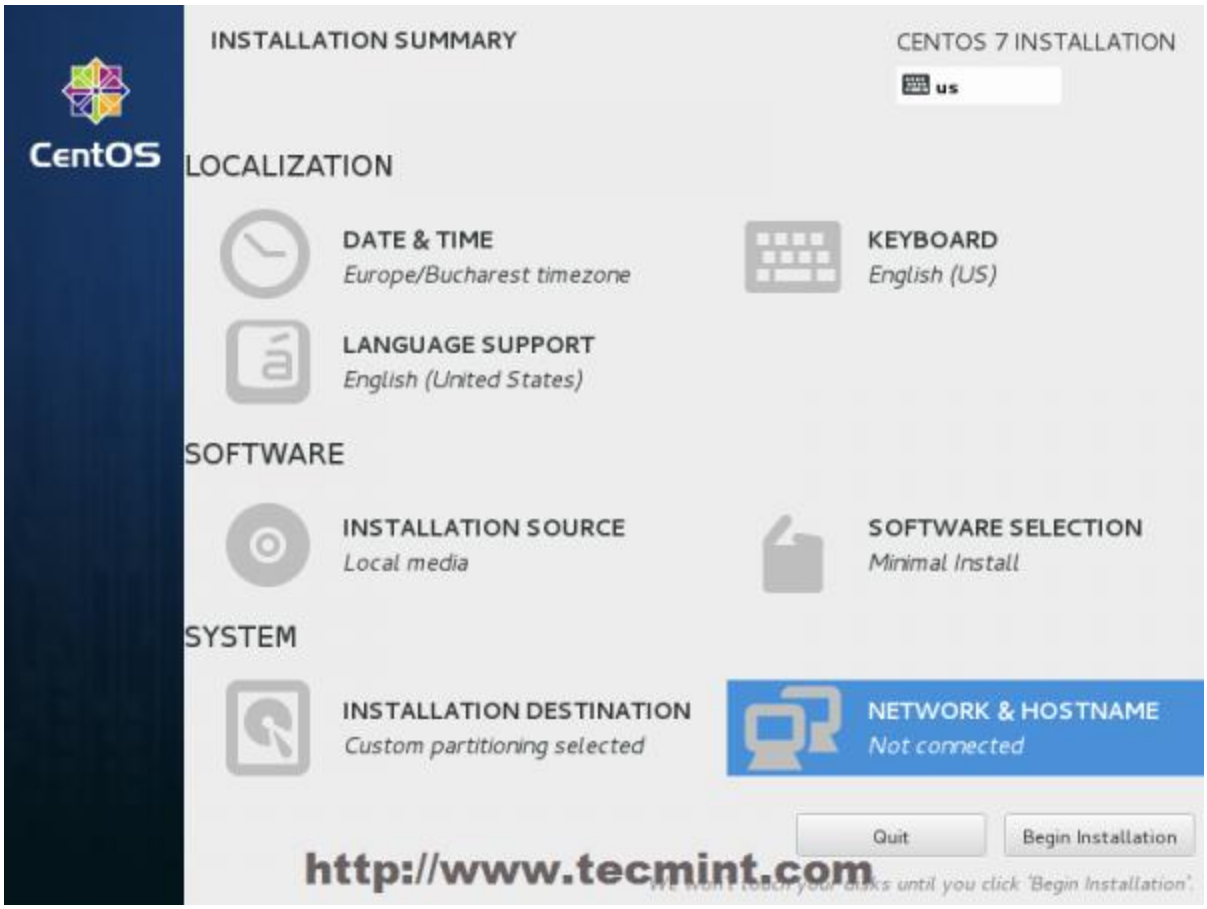


Summary of Partition Changes

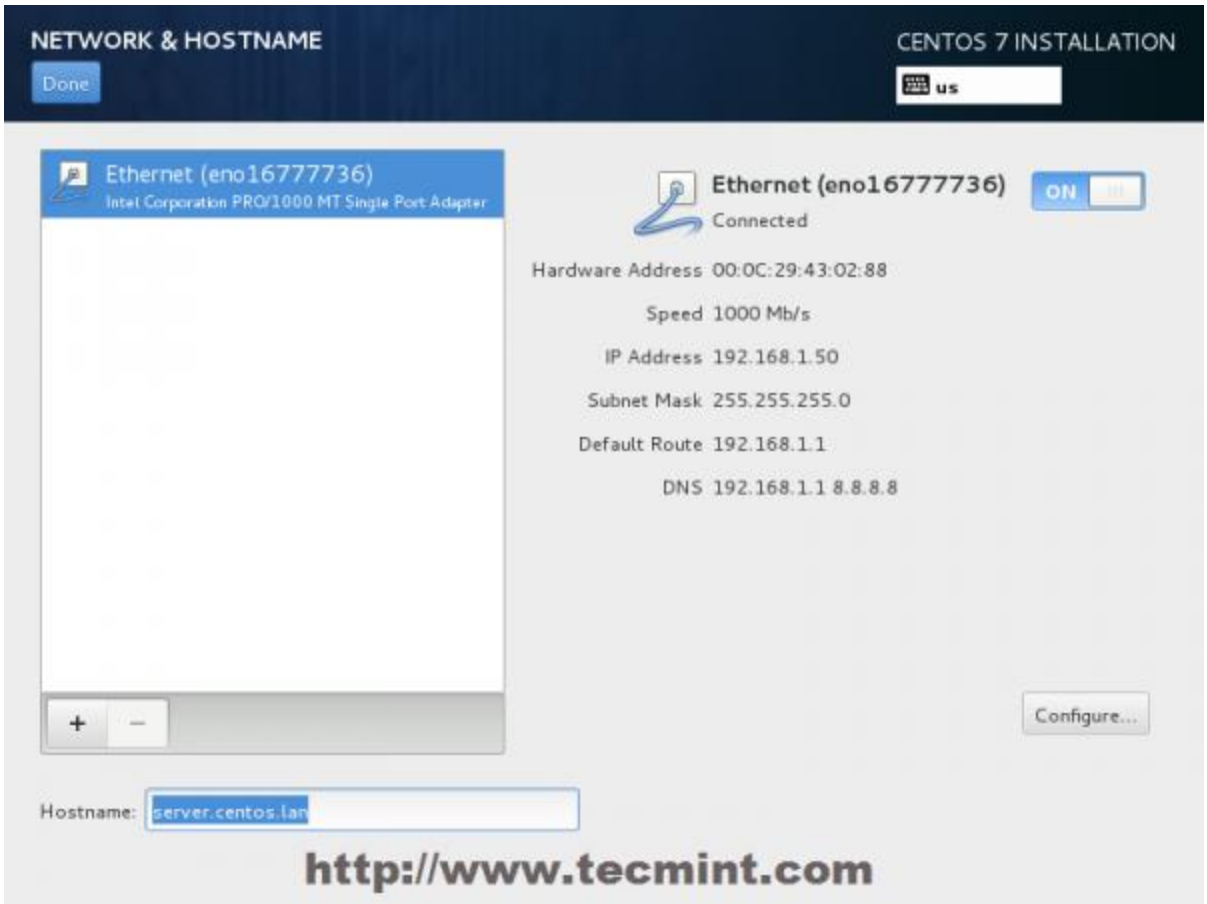
NOTE: For those users, who have hard-disks more than 2TB in size, the installer automatically will convert partition table to GPT, but if you wish to use GPT table on smaller disks than 2TB, then you should use the argument **inst.gpt** to the installer boot command line in order to change the default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

12. 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.

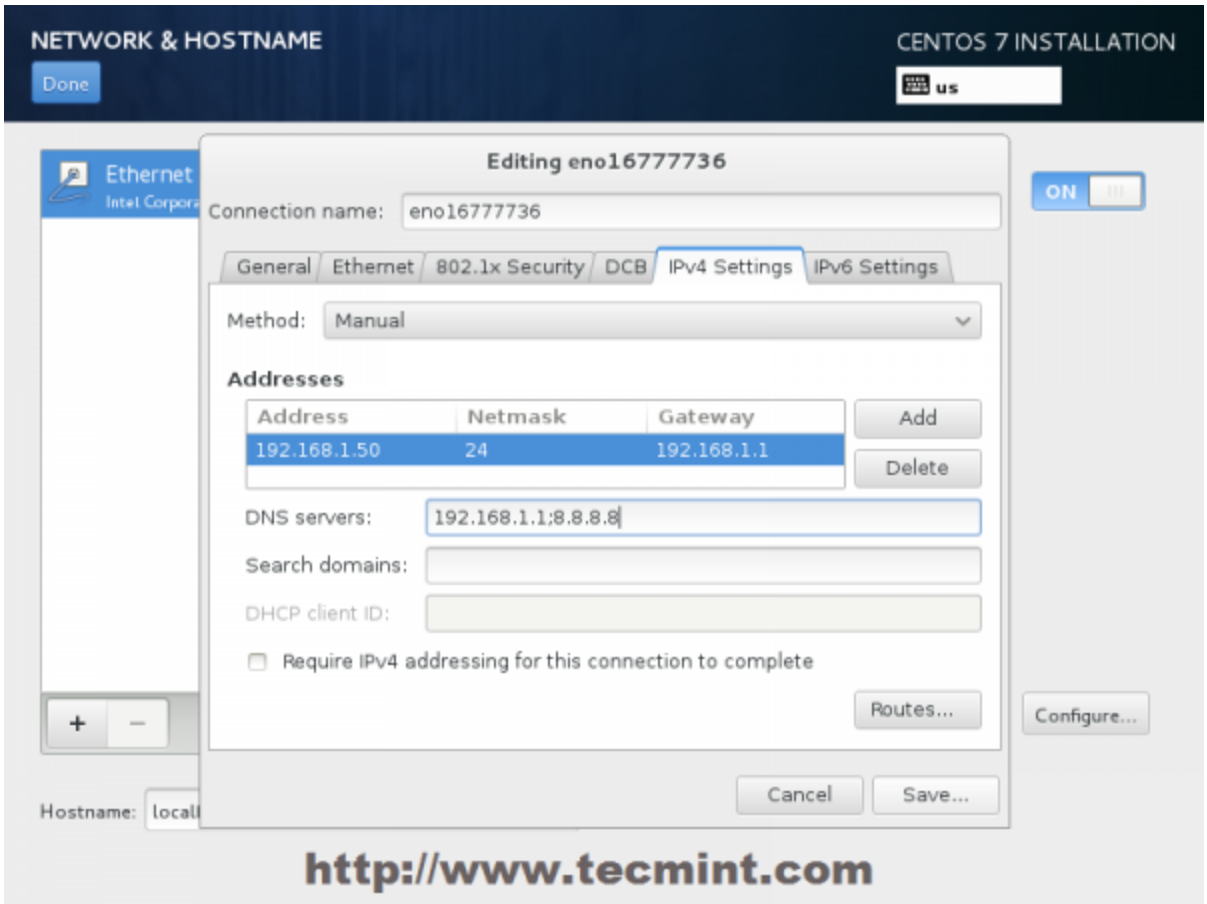


Set System Hostname



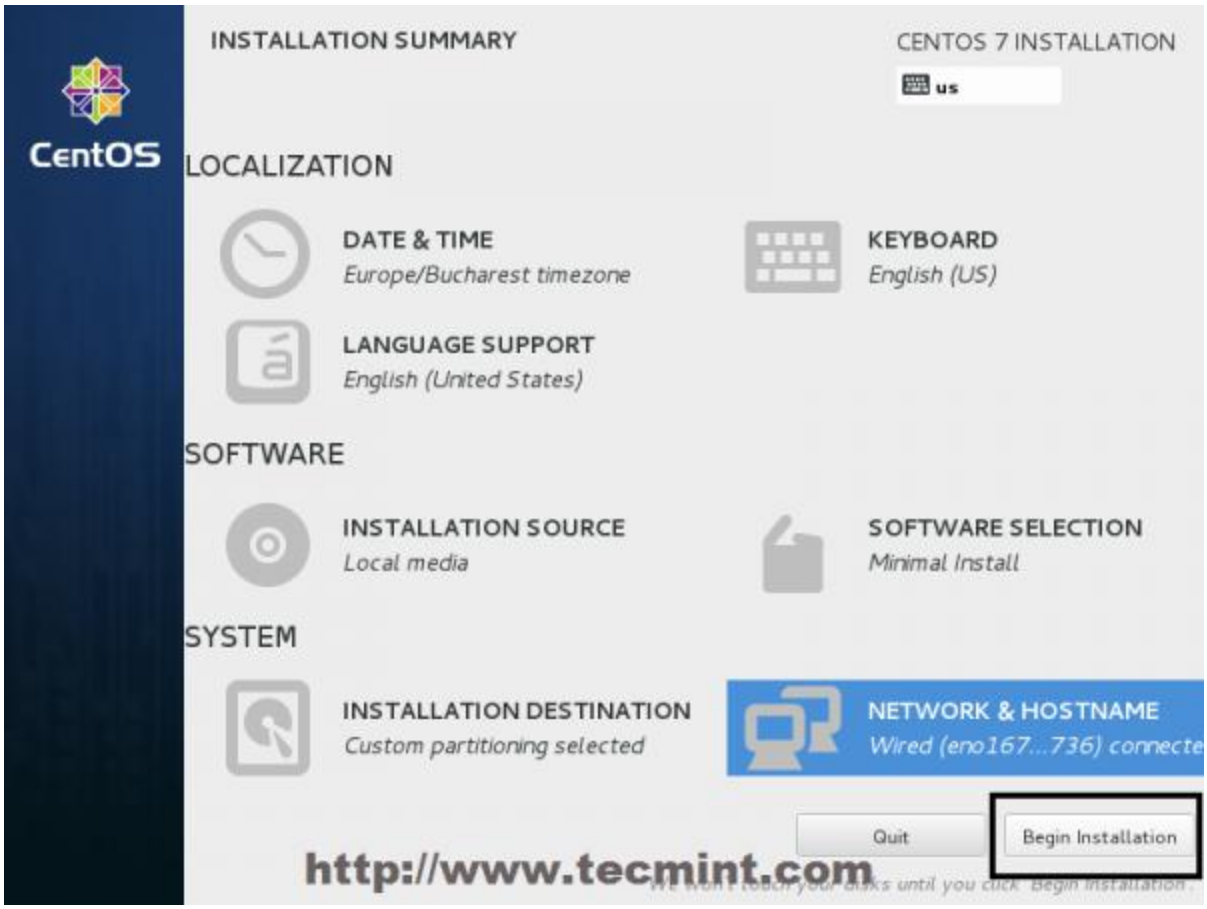
Enable Ethernet Interface

13. 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.

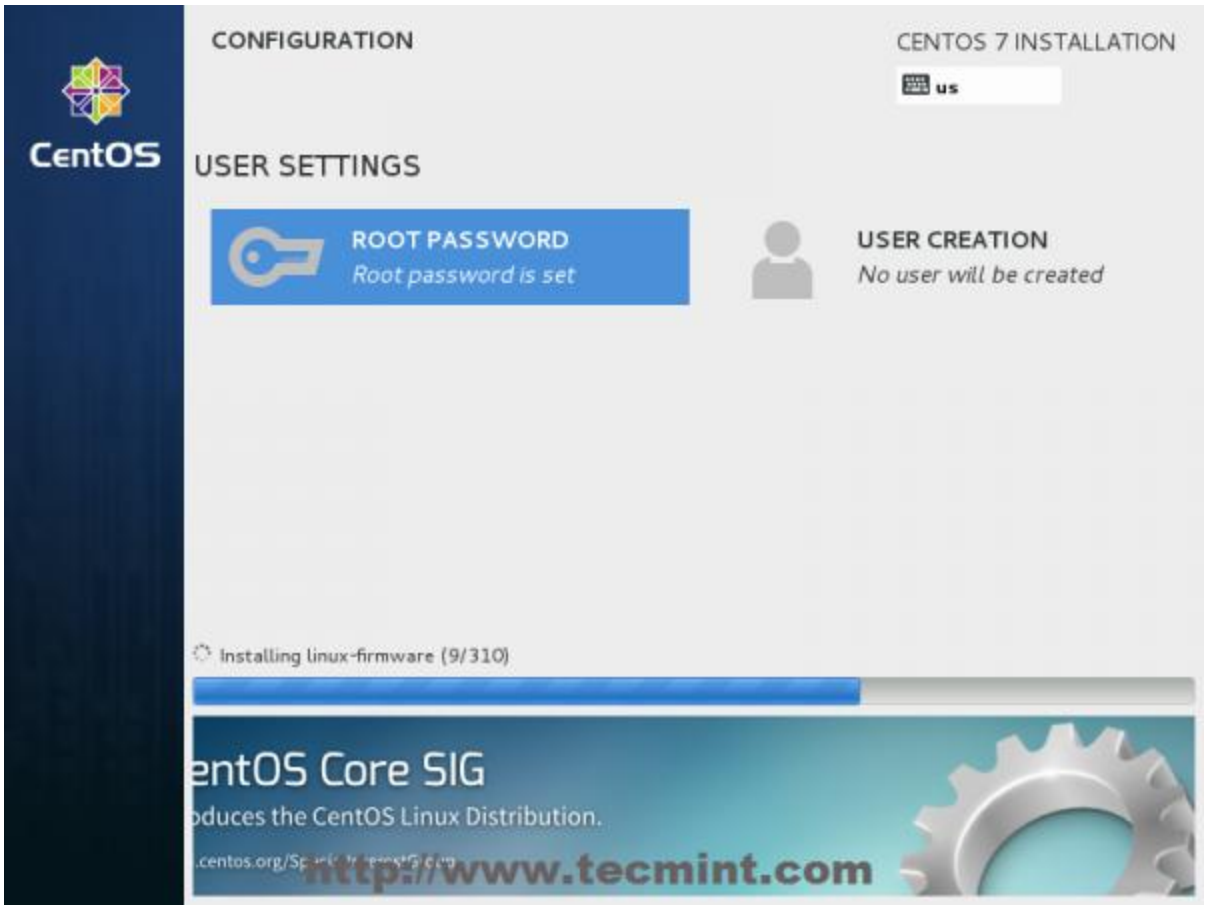


Enter Network Settings

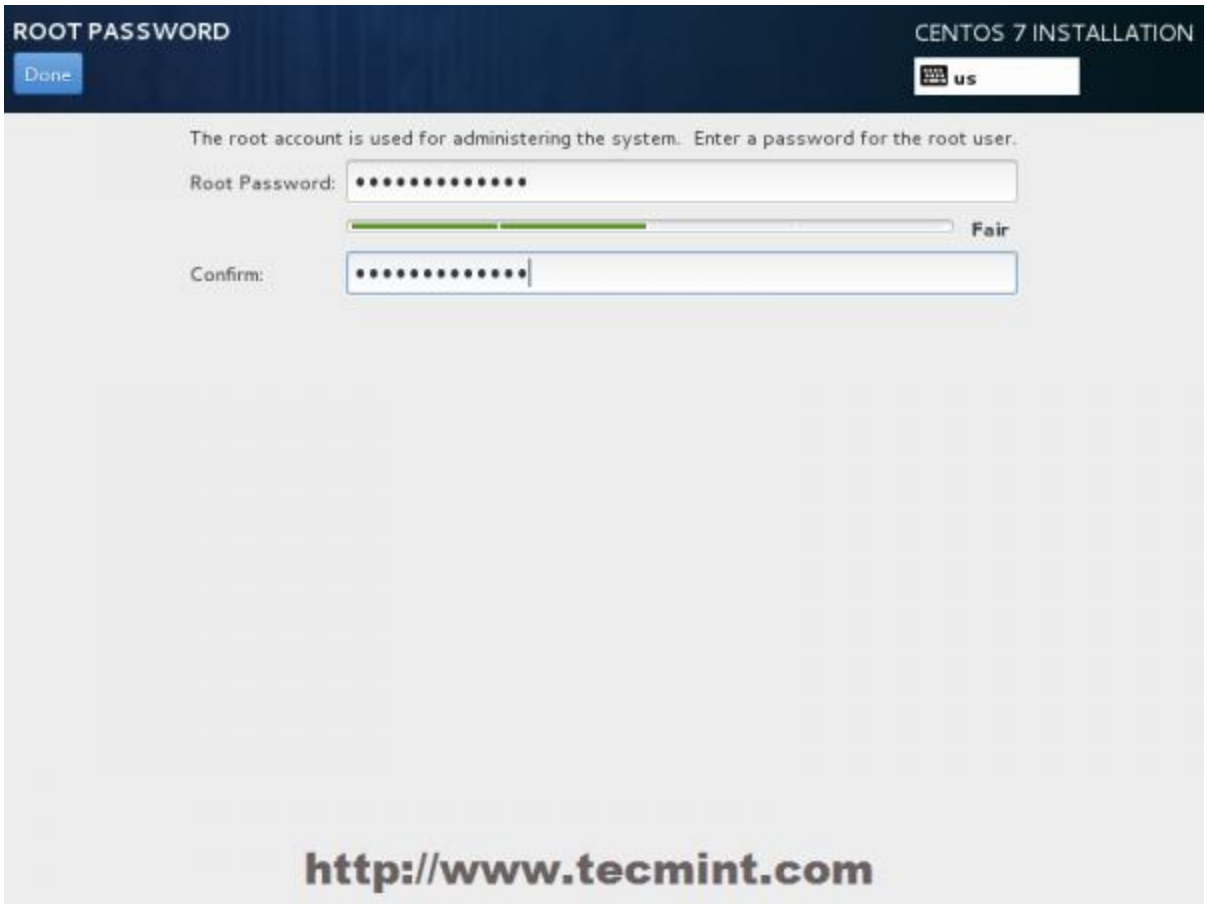
14. Now it's time to start installation process by pressing on **Begin Installation** button and set up a strong password for **root** account.



Click on Begin Installation

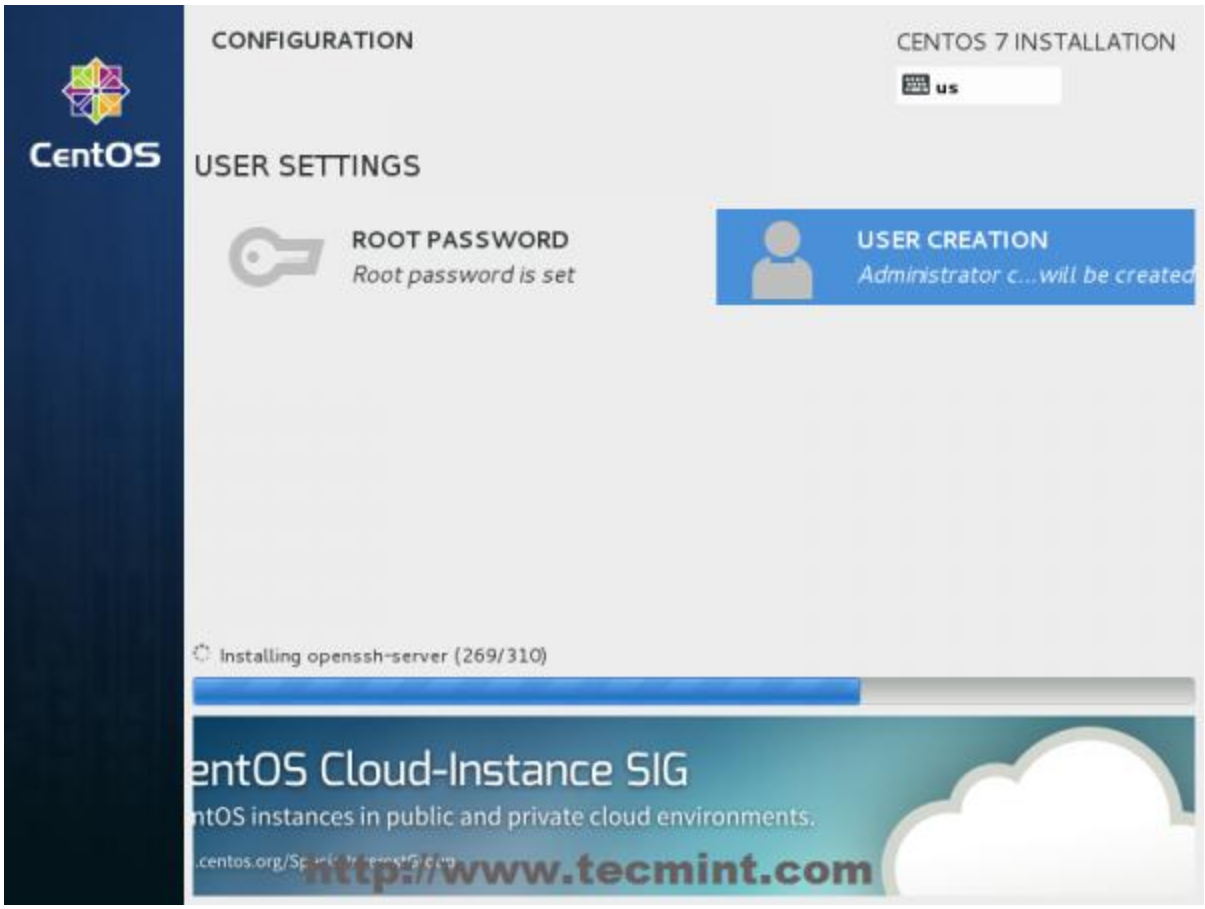


Select Root Password

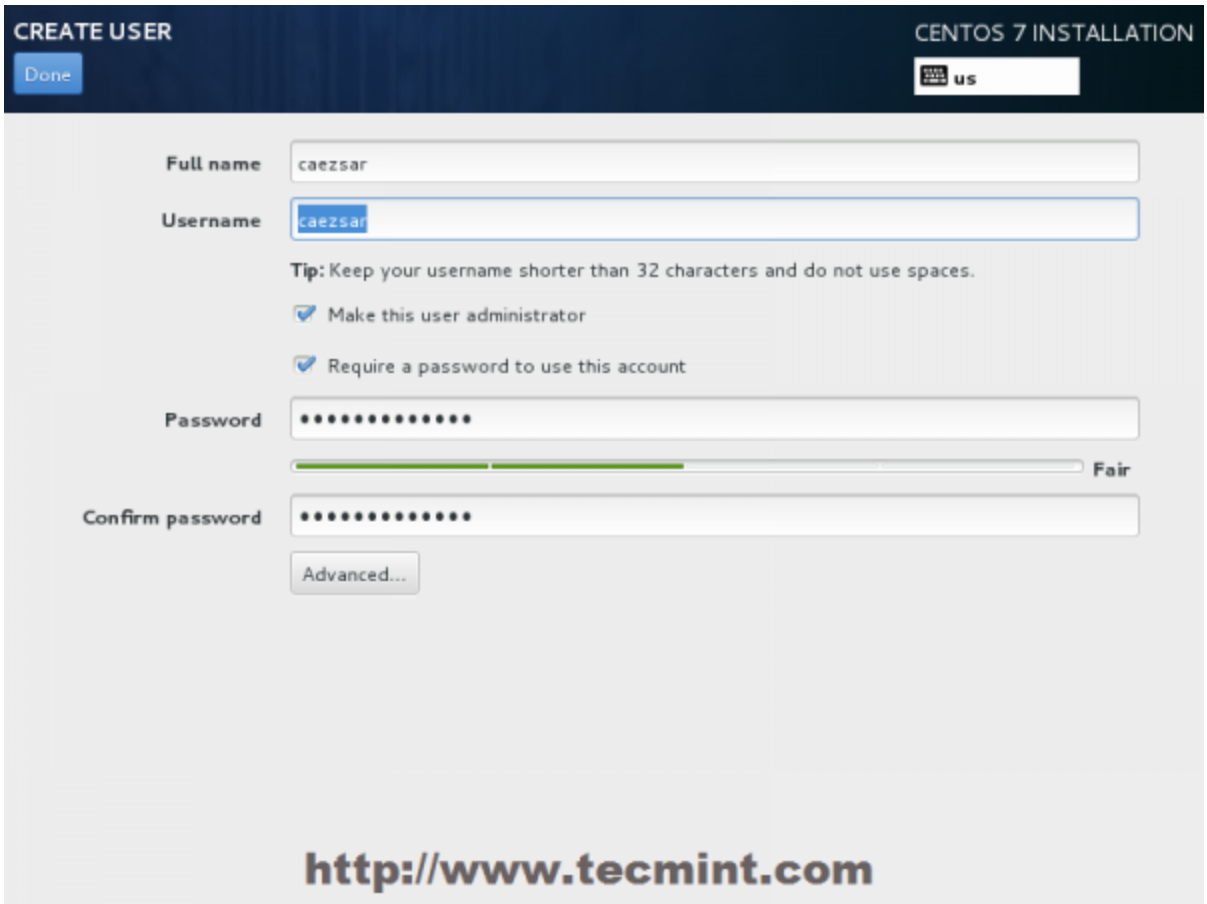


Enter Root Password

15. 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.

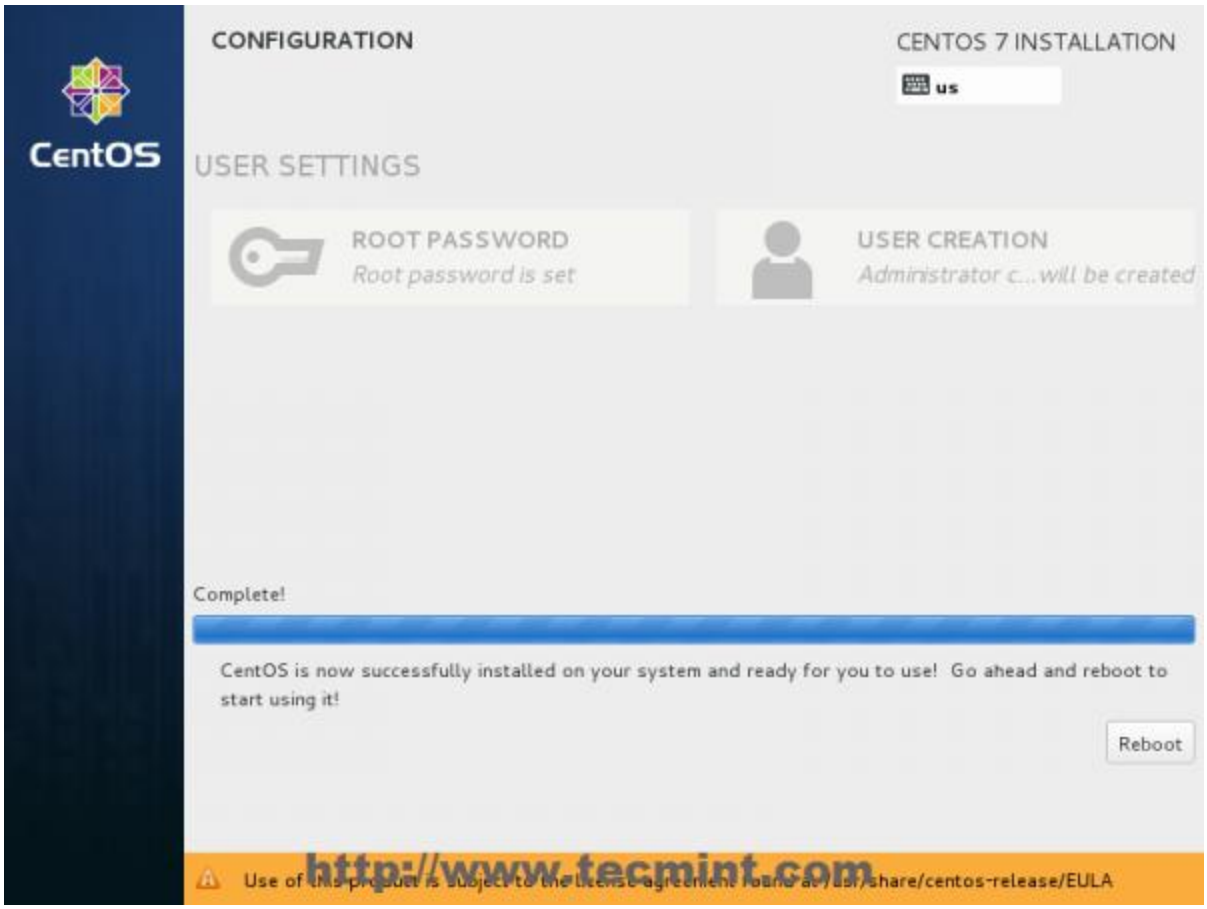


CentOS 7 Installation Process



User Creation and Set Password

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



CentOS 7 Installation Complete

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 perform other system tasks, such as update you system and install other useful software needed to run day to day tasks.

Disable Firewall:

Switch user to root (# su root) put password

How to Stop and Disable Firewalld on CentOS 7

It is **highly** recommended that you have another firewall protecting your network or server before, or immediately after, disabling firewalld.

Pre-Flight Check

- These instructions are intended specifically for stopping and disabling firewalld CentOS 7.
- I'll be working from a Liquid Web Self Managed CentOS 7 server, and I'll be logged in as root.

Disable Firewalld

To disable firewalld, run the following command as root:

```
systemctl disable firewalld
```

Stop Firewalld

To stop firewalld, run the following command as root:

```
systemctl stop firewalld
```

Check the Status of Firewalld

To check the status of firewalld, run the following command as root:

```
systemctl status firewalld
```

Wait, you actually wanted to Start and Enable Firewalld on CentOS 7? Then hit our tutorial on: How to Start and Enable Firewalld on CentOS 7!

How to Setup network on centos 7

Setup network on centos 7

let's start, Type "**nmcli d**" command in your terminal for quick identification of Ethernet cards installed in your machine.

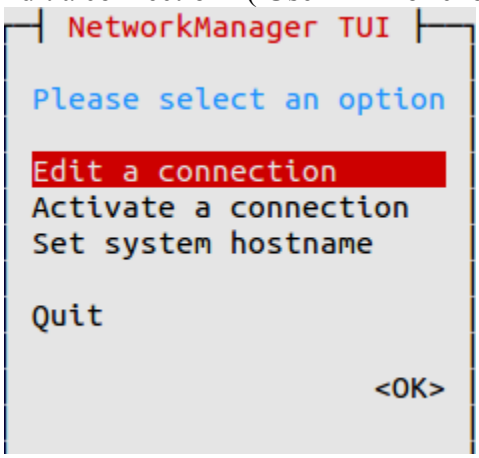
```
[root@krizna ~]# nmcli d
DEVICE   TYPE      STATE                                  CONNECTION
enp0s17  ethernet  disconnected                           --
enp0s18  ethernet  disconnected                           --
lo       loopback  unmanaged                              --
```

Here we have 2 interfaces named "**enp0s17**" and "**enp0s18**" . it might be different in your case (Eg: **em1** or **p4p1**).

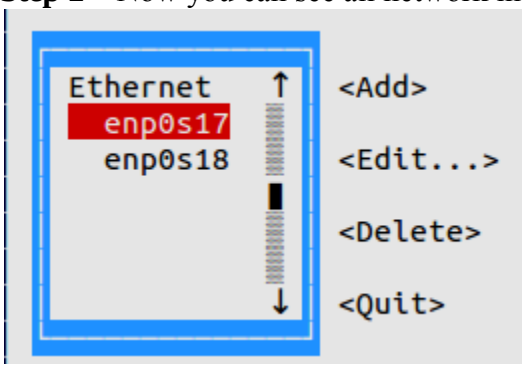
GUI Mode

Recommended for beginners

Step 1 » Type this command "**nmtui**" to open Network manager and press enter after choosing " Edit a connection" (Use TAB for choosing options) .



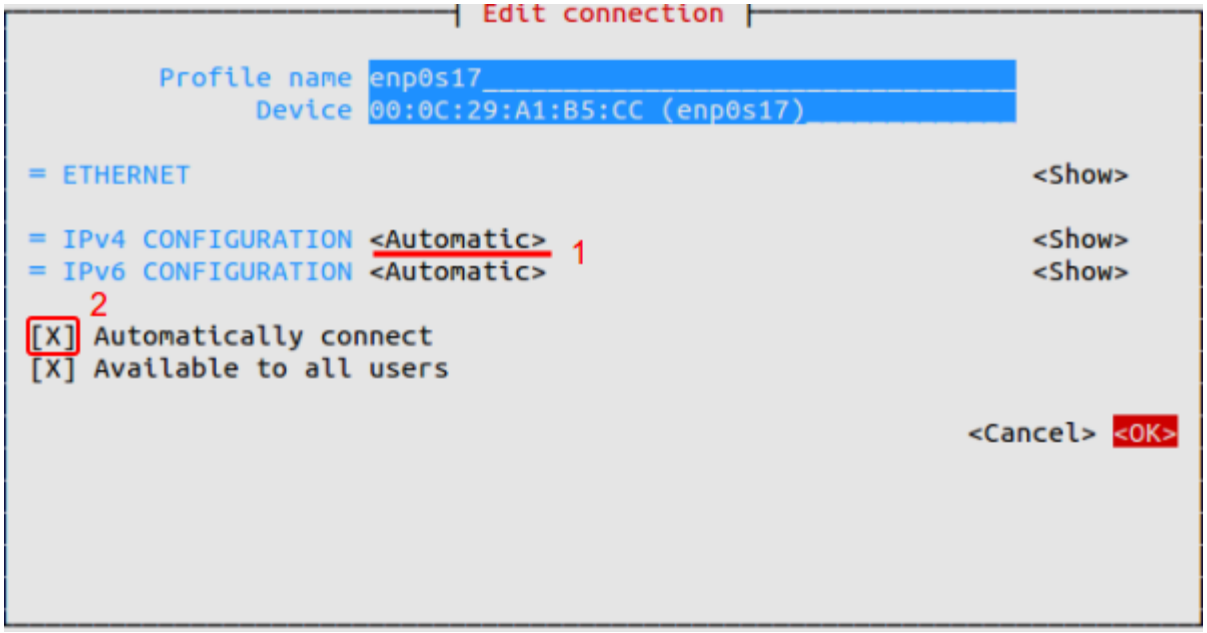
Step 2 » Now you can see all network interfaces, choose one and click "**Edit**".



» DHCP configuration

Step 3 » For DHCP,

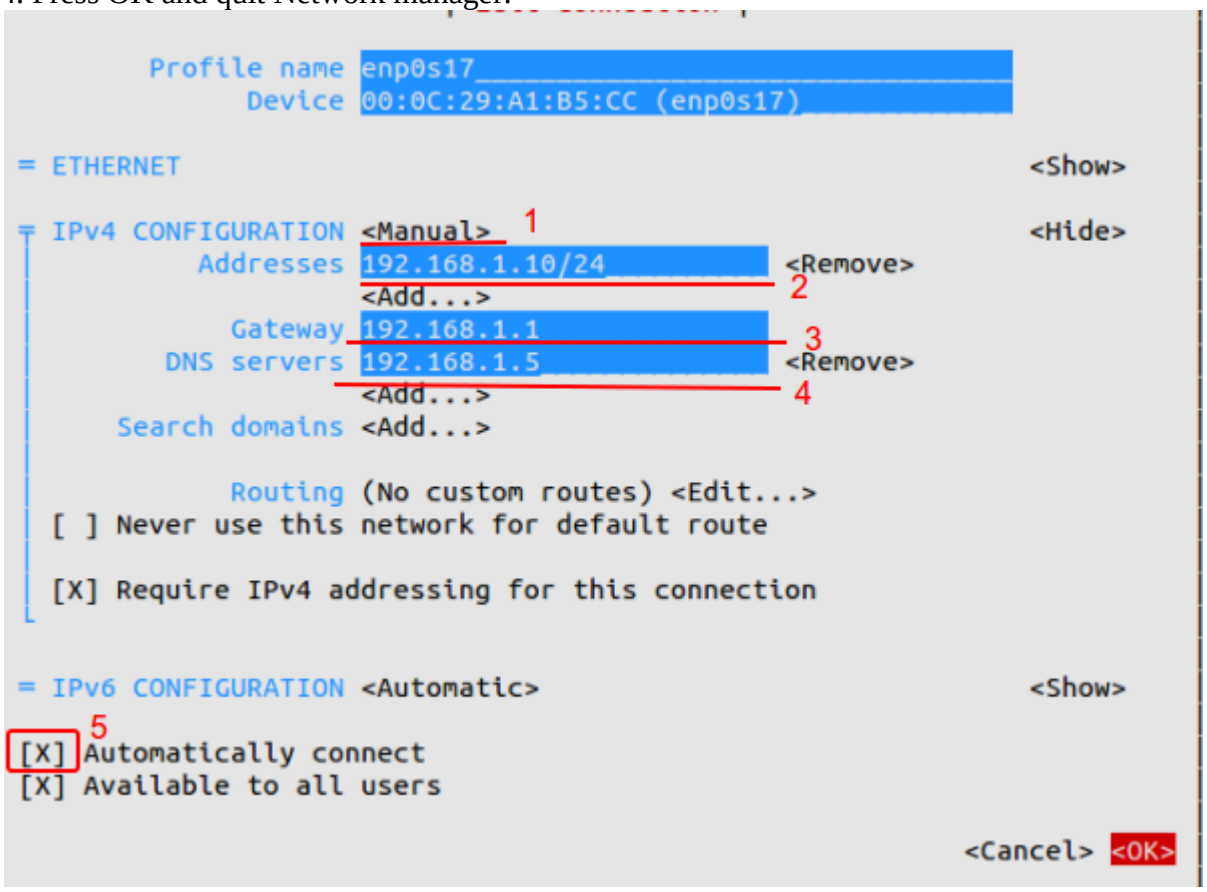
1. Choose "**Automatic**" in IPv4 CONFIGURATION.
2. Choose Automatic Connect check box.
3. Press OK and quit Network manager.



Now Restart network service by typing below command.
`systemctl restart network` Now your server will get IP Address from DHCP .

» Static configuration

Step 4 » For manual IP address,
1. Choose "**Manual**" in IPv4 CONFIGURATION.
2. Add IP Address with Subnet , Gateway and DNS server (Refer below image).
3. Choose Automatic Connect check box.
4. Press OK and quit Network manager.



Now Restart network service by typing below command.
`systemctl restart network` That's it, Interface will have static IP.

Command Mode

Step 1 » Network interface config files are located in **/etc/sysconfig/network-scripts/** directory. Open **ifcfg-enp0s17** file (For interface enp0s17) and you can see the content like below.

```
[root@krizna ~]# vi or nano /etc/sysconfig/network-scripts/ifcfg-enp0s17
TYPE=Ethernet
BOOTPROTO=none
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
NAME=enp0s17
UUID=7f1aff2d-b154-4436-9497-e3a4dedddcef
ONBOOT=no
HWADDR=00:0C:29:A1:B5:D6
PEERDNS=yes
PEERROUTES=yes
IPV6_PEERDNS=yes
IPV6_PEERROUTES=yes
```

» DHCP configuration

Step 2 » For DHCP
Find the below lines in config File.
`BOOTPROTO=none`
`ONBOOT=no` and replace with
`BOOTPROTO=dhcp`
`ONBOOT=yes`Now Restart network service by typing below command.
`systemctl restart network` Now your server will get IP Address from DHCP

» Static configuration

Step 3 » For Static IP.
Find the below lines in config File.
`BOOTPROTO=none`
`ONBOOT=no` and replace with
`BOOTPROTO=static`
`ONBOOT=yes` And add the below lines at the end of the file.

```
IPADDR=172.27.0.32
NETMASK=255.255.255.0
GATEWAY=172.27.0.1
DNS1=172.27.0.5
```

File will look like below after changes.

```
TYPE=Ethernet
BOOTPROTO=static
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
NAME=enp0s17
UUID=f0c5b37d-299a-43cb-b74b-618bb252d129
```

```
ONBOOT=yes
HWADDR=00:0C:29:A1:B5:CC
IPV6_PEERDNS=yes
IPV6_PEERROUTES=yes
IPADDR=192.168.1.10
NETMASK=255.255.255.0
GATEWAY=192.168.1.1
DNS1=192.168.1.5
```

Now Restart network service by typing below command.

`systemctl restart network`Now Interface will have static IP.

Additionally you can use **/etc/sysconfig/network** file for hostname and DNS .

```
HOSTNAME=server.krizna.com
DNS1=192.168.1.5
DNS2=8.8.8.8
SEARCH=krizna.com
```

Have a nice day

SSH Server and Client Setup

Configure SSH Server to login to a server from remote computer. SSH uses 22/TCP port.

OpenSSH is already installed by default even if you installed CentOS with "Minimal Install", so it's not necessarry to install new packages. You can login with Password Authentication by default, but change some settings for security like follows.

[root@client ~]# -y install openssh (Already installed)

[root@dlp ~]# vi or nano /etc/ssh/sshd_config
line 48: uncomment and change (prohibit root login remotely)

PermitRootLogin no
line 77: uncomment

PermitEmptyPasswords no
PasswordAuthentication yes
[root@dlp ~]# systemctl restart sshd

Configure SSH Client : CentOS

Configure SSH Client on CentOS.

Install SSH Client.

[root@client ~]# -y install openssh-clients

Connect to SSH server with a common user.

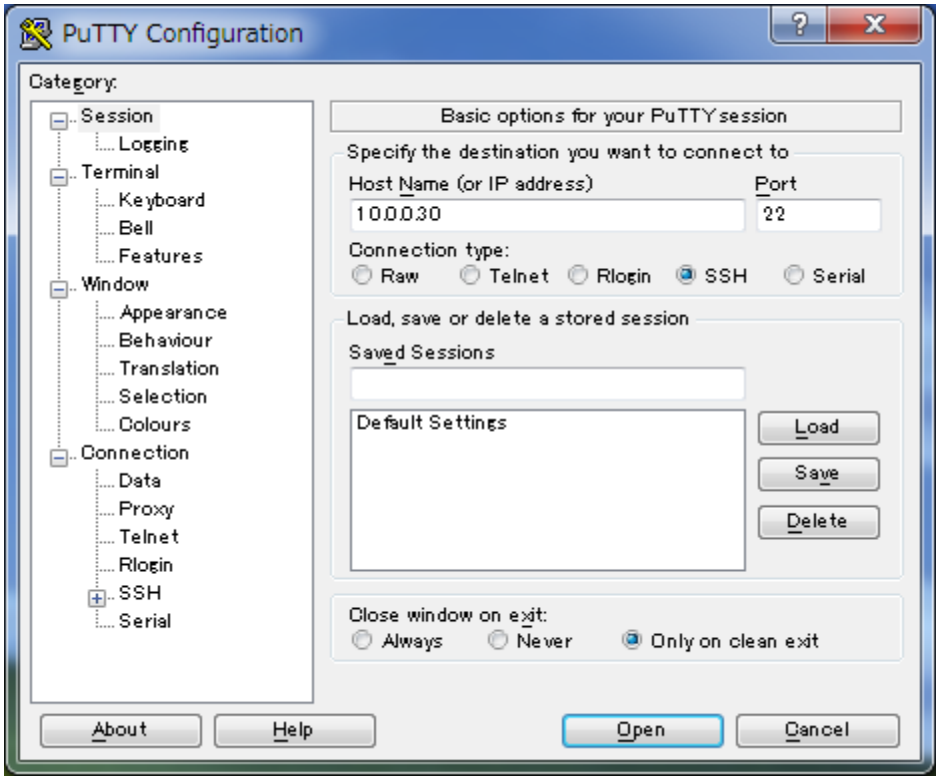
ssh [username@(hostname or IP address)]

[root@client ~]# ssh cent@dlp.server.world

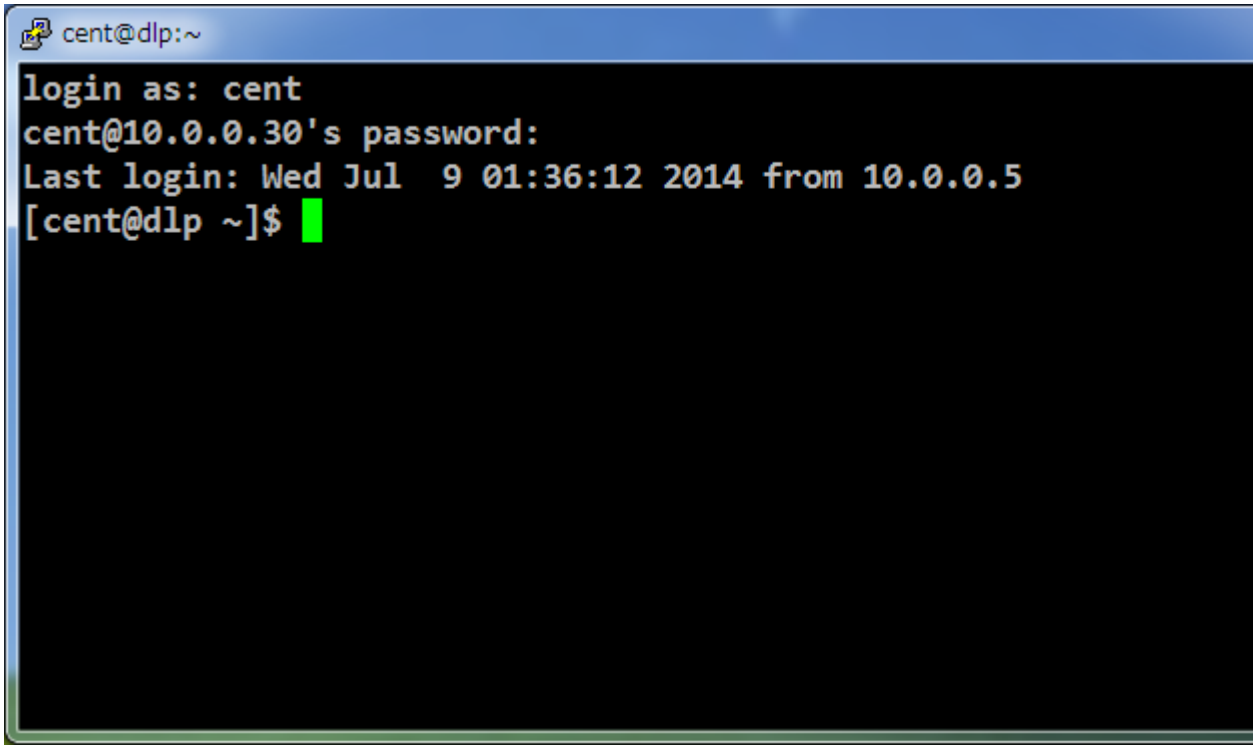
Configure SSH Client : Windows

Configure SSH Client on Windows.

Get a software which you can login with SSH from Windows clients. This example shows to use Putty. Install and start it and input your server's IP address and Click 'Open' button like follows.



[6] After succeccing authentication, it's possible to login like follows.



Exporting Proxy

yum add proxy on CentOS:

http_proxy= http://172.16.200.1:3128

export http_proxy

yum update

There is a standard way to do it by **adding a proxy directive** to **/etc/yum.conf**

However for some reason:

```
proxy=http://your-proxy-url.com:8080
proxy_username=yum-user
proxy_password=qwerty
```

To make yum work via proxy in gnome-terminal run first:

```
export http_proxy=http://your-proxy-server.com:8080
```

or **if proxy is protected by username / password** run instead:

```
export username='yum-user'
export password='qwerty'
export http_proxy="http://$username:$password@your-proxy-server:8080/
```

Afterwards **yum will work via the proxy**, i.e.:

```
yum update && yum upgrade
```

How to Install LAMP on CentOS 7

LAMP which originally stands for Linux, Apache, MySQL and PHP has now recently changed with the rise of MariaDB, a drop-in replacement for original MySQL. Long story short, MariaDB is a fork of MySQL and developed by MySQL developers itself. It has almost all features of what MySQL has and features library binary equivalency and exact matching with MySQL APIs and commands. It means if an app is able to run with MySQL and it also is able to on MariaDB without any glitch. I will not explain what is Apache and PHP as I've explained before and I believe you already knew what it is.

You may need:

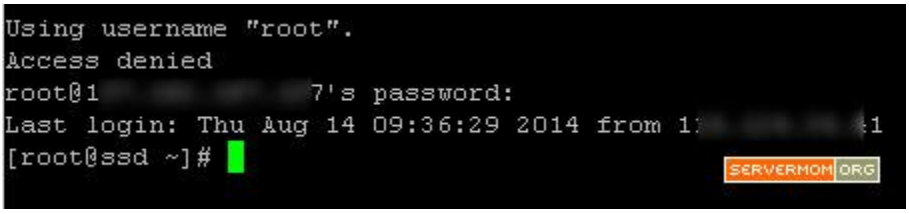
1. A server running CentOS 7. I recommend you to use CentOS 7 x86_64 minimal if available.
2. A knowledge on how to use Putty or Terminal to access a server via SSH.
3. I believe you knew *-at least part of-* most common Unix commands used to manage an unmanaged server.
4. A spare time of your life and a cup of coffee.

Install Apache Web Server

Being the most popular web server, Apache is commonly included in most of recent Linux Distro so installation will be very easy.

Before you proceed to the next steps, it is better to explain that all commands in this tutorial are written without the "sudo" prefix. In this tutorial I use root but you may also login as separate user with root privilege. However if you disabled root login and you logged in using another username with root privilege, you can add the "**sudo**" prefix all by your self. Alternatively you can simply type **su**, hit Enter and type in your password twice to switch as root.

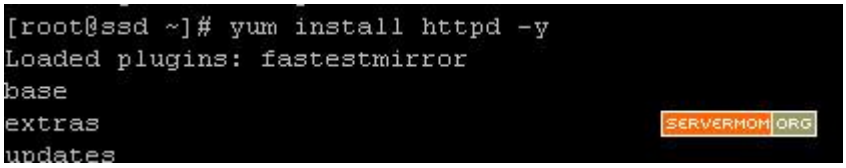
Step 1 – Login to your server via Putty or Terminal.



Step 2 – Now issue command below to install Apache 2.4 on your CentOS 7 server:

```
yum install httpd -y
```

As you can see the command is still the same.



And when the process finished, you'll see something like this:

```
Installed:
  httpd.x86_64 0:2.4.6-18.el7.centos

Dependency Installed:
  apr.x86_64 0:1.4.8-3.el7                          apr-util.x86_6
  centos-logos.noarch 0:70.0.6-1.el7.centos         httpd-tools.x8
  mailcap.noarch 0:2.1.41-2.el7

Complete!
[root@ssd ~]#
```

Step 3 – Now you have Apache 2.4 installed which you can then start the service by typing command below:

```
systemctl start httpd.service
```

or,..

```
service httpd start
```

```
[root@ssd ~]# systemctl start httpd.service
[root@ssd ~]#
```

Available commands:

```
systemctl status|start|stop|restart|reload httpd.service

# OR, old command :

service httpd status|start|stop|restart|reload
```

Step 4 – You can verify that Apache is really running by opening your favorite web browser and access your vps via its IP address:

```
http://xxx.xxx.xxx.xxx
```

and you'll see default Apache welcome page.

In current example I've installed Apache v2.4.6.

or, you can directly issue this command:

```
systemctl status httpd.service
```

you'll see something like this:

```
[root@ssd ~]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: active (running) since Fri 2014-08-15 11:58:08 EDT; 23min ago
 Main PID: 619 (httpd)
   Status: "Total requests: 9; Current requests/sec: 0; Current traffic:   0 B/sec"
   CGroup: /system.slice/httpd.service
           ãã619 /usr/sbin/httpd -DFOREGROUND
           ãã620 /usr/sbin/httpd -DFOREGROUND
           ãã621 /usr/sbin/httpd -DFOREGROUND
           ãã622 /usr/sbin/httpd -DFOREGROUND
           ãã623 /usr/sbin/httpd -DFOREGROUND
           ãã624 /usr/sbin/httpd -DFOREGROUND
           ãã628 /usr/sbin/httpd -DFOREGROUND
           ãã629 /usr/sbin/httpd -DFOREGROUND
           ãã630 /usr/sbin/httpd -DFOREGROUND
           ãã631 /usr/sbin/httpd -DFOREGROUND
           ãã632 /usr/sbin/httpd -DFOREGROUND

Aug 15 11:58:08 ssd.servermom.org systemd[1]: Started The Apache HTTP Server.
[root@ssd ~]#
```

Enable Apache to automatically run every time your server reboot:

```
systemctl enable httpd.service
```

Install PHP5

Step 5 – Now, it is time to install PHP5. Default command is:

```
yum install php -y
```

That' really is a simple command but since we will install MySQL then we'll need PHP MySQL modules plus any other PHP5 modules you website / app may need it. You can view all available modules using this command:

```
yum search php-
```

Confused? You can read my previous article or you can simply use command below that includes common PHP5 modules most websites can run with it.

```
yum install php php-common php-cli php-devel php-gd php-imap php-intl php-mysql php-process php-xml php-xmlrpc php-zts -y
```

And once done, you'll see something like this:

```
Installed:
  php.x86_64 0:5.4.16-23.el7_0                    php-cli.x86_64 0:5.4.16-23.el7_0
  php-common.x86_64 0:5.4.16-23.el7_0             php-devel.x86_64 0:5.4.16-23.el7_0
  php-gd.x86_64 0:5.4.16-23.el7_0                 php-intl.x86_64 0:5.4.16-23.el7_0
  php-mysql.x86_64 0:5.4.16-23.el7_0              php-process.x86_64 0:5.4.16-23.el7_0
  php-xml.x86_64 0:5.4.16-23.el7_0                php-xmlrpc.x86_64 0:5.4.16-23.el7_0

Dependency Installed:
  autoconf.noarch 0:2.69-11.el7                   automake.noarch 0:1.13.4-3.el7
  freetype.x86_64 0:2.4.11-9.el7                  libX11.x86_64 0:1.6.0-2.1.el7
  libX11-common.noarch 0:1.6.0-2.1.el7            libXau.x86_64 0:1.0.8-2.1.el7
  libXpm.x86_64 0:3.5.10-5.1.el7                  libicu.x86_64 0:50.1.2-11.el7
  libjpeg-turbo.x86_64 0:1.2.90-5.el7             libpng.x86_64 2:1.5.13-5.el7
  libxcb.x86_64 0:1.9-5.el7                       libxslt.x86_64 0:1.1.28-5.el7
  libzip.x86_64 0:0.10.1-8.el7                    m4.x86_64 0:1.4.16-9.el7
  pcre-devel.x86_64 0:8.32-12.el7                 perl-Test-Harness.noarch 0:3.28-2.el7
  perl-Thread-Queue.noarch 0:3.02-2.el7           php-pdo.x86_64 0:5.4.16-23.el7_0
  t1lib.x86_64 0:5.1.2-14.el7

Complete!
[root@ssd ~]#
```

You can test which version of PHP is installed by typing **php -v** command.

```
[root@ssd ~]# php -v
PHP 5.4.16 (cli) (built: Aug  6 2014 13:12:28)
Copyright (c) 1997-2013 The PHP Group
Zend Engine v2.4.0, Copyright (c) 1998-2013 Zend Technologies
[root@ssd ~]#
```

In my example it is **PHP v5.4.16**.

Install MariaDB MySQL Server

Step 6 – Installing MariaDB mysql server on CentOS 7 is pretty easy and once again we'll make us of yum package manager:

```
yum install mariadb-server mariadb -y
```

```
[root@ssd ~]# yum install mariadb-server mariadb -y
Loaded plugins: fastestmirror
base
extras
updates
```

and once done you'll see something like this:

```
  Installing : perl-DBI-1.627-4.el7.x86_64                                  8
  Installing : perl-DBD-MySQL-4.023-5.el7.x86_64                            9
  Installing : 1:mariadb-server-5.5.37-1.el7_0.x86_64                      10
  Verifying  : perl-Compress-Raw-Bzip2-2.061-3.el7.x86_64                   1
  Verifying  : perl-Net-Daemon-0.48-5.el7.noarch                            2
  Verifying  : 1:mariadb-5.5.37-1.el7_0.x86_64                              3
  Verifying  : perl-PlRPC-0.2020-14.el7.noarch                              4
  Verifying  : 1:mariadb-server-5.5.37-1.el7_0.x86_64                       5
  Verifying  : 1:perl-Compress-Raw-Zlib-2.061-4.el7.x86_64                  6
  Verifying  : libaio-0.3.109-12.el7.x86_64                                 7
  Verifying  : perl-DBI-1.627-4.el7.x86_64                                  8
  Verifying  : perl-IO-Compress-2.061-2.el7.noarch                          9
  Verifying  : perl-DBD-MySQL-4.023-5.el7.x86_64                           10

Installed:
  mariadb.x86_64 1:5.5.37-1.el7_0              mariadb-server.x86_64 1:5.5.37-1.el7_0

Dependency Installed:
  libaio.x86_64 0:0.3.109-12.el7                  perl-Compress-Raw-Bzip2.x86_64 0:2.061-3.el7
  perl-Compress-Raw-Zlib.x86_64 1:2.061-4.el7     perl-DBD-MySQL.x86_64 0:4.023-5.el7
  perl-DBI.x86_64 0:1.627-4.el7                   perl-IO-Compress.noarch 0:2.061-2.el7
  perl-Net-Daemon.noarch 0:0.48-5.el7             perl-PlRPC.noarch 0:0.2020-14.el7

Complete!
[root@ssd ~]#
```

Step 7 – Now you can start MariaDB server for the very first time using this simple systemctl command :

```
systemctl start mariadb.service
```

```
[root@ssd ~]# systemctl start mariadb.service
[root@ssd ~]# systemctl status mariadb
mariadb.service - MariaDB database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; disabled)
   Active: active (running) since Sat 2014-08-16 10:07:10 EDT; 38s ag
  Process: 954 ExecStartPost=/usr/libexec/mariadb-wait-ready $MAINPID
)
  Process: 875 ExecStartPre=/usr/libexec/mariadb-prepare-db-dir %n (c
 Main PID: 953 (mysqld_safe)
   CGroup: /system.slice/mariadb.service
           ââ 953 /bin/sh /usr/bin/mysqld_safe --basedir=/usr
           ââ1109 /usr/libexec/mysqld --basedir=/usr --datadir=/var/l

Aug 16 10:07:08 ssd.servermom.org mariadb-prepare-db-dir[875]: The la
Aug 16 10:07:08 ssd.servermom.org mariadb-prepare-db-dir[875]: You ca
Aug 16 10:07:08 ssd.servermom.org mariadb-prepare-db-dir[875]: http:/
Aug 16 10:07:08 ssd.servermom.org mariadb-prepare-db-dir[875]: Suppor
Aug 16 10:07:08 ssd.servermom.org mariadb-prepare-db-dir[875]: SkySQL
Aug 16 10:07:08 ssd.servermom.org mariadb-prepare-db-dir[875]: Altern
Aug 16 10:07:08 ssd.servermom.org mariadb-prepare-db-dir[875]: http:/
Aug 16 10:07:08 ssd.servermom.org mysqld_safe[953]: 140816 10:07:08 m
Aug 16 10:07:08 ssd.servermom.org mysqld_safe[953]: 140816 10:07:08 m
Aug 16 10:07:10 ssd.servermom.org systemd[1]: Started MariaDB databas
Hint: Some lines were ellipsized, use -l to show in full.
[root@ssd ~]#
```

You may also see the status of MariaDB by typing:

```
systemctl status mariadb
```

MariaDB Initial Configuration

Step 8 – So its service is now running but there is one thing you should do immediately: configuring MariaDB setup for the very first time like setting up your mysql root password. Issue this command:

```
mysql_secure_installation
```

Then you'll see a series of question, just answer it accordingly. The main important part is to define your root password while everything else is just up to you or you can simply hit the "ENTER" key through each prompt to accept the default values.

```
[root@ssd ~]# mysql_secure_installation
/usr/bin/mysql_secure_installation: line 379: find_mysql_client: command

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user.  If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

Set root password? [Y/n] y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
 ... Success!


By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] y
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] n
 ... skipping.

By default, MariaDB comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] y
 - Dropping test database...
 ... Success!
 - Removing privileges on test database...
 ... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] y
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
[root@ssd ~]#
```

If you need to automatically run MariaDB everytime your server boot, simply issue this command:

```
systemctl enable mariadb.service
```

```
[root@ssd ~]# systemctl enable mariadb.service
ln -s '/usr/lib/systemd/system/mariadb.service' '/etc/systemd/system/multi-use
.service'
[root@ssd ~]#
```

You may also need to test your newly installed MariaDB by logging in as root:

```
mysql -u root -p
```

```
[root@ssd ~]# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 9
Server version: 5.5.37-MariaDB MariaDB Server

Copyright (c) 2000, 2014, Oracle, Monty Program Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input

MariaDB [(none)]>
```

As you can see from the screenshot above, it is **Maria DB v5.5.37**.

Step 9 – Also test if Apache and PHP is running well and able to process any *.php files. Create a php info page using this command followed by restarting apache

```
echo "<?php phpinfo(); ?>" > /var/www/html/info.php
```

```
[root@ssd ~]# echo "<?php phpinfo(); ?>" > /var/www/html/info.php
[root@ssd ~]# cd /var/www/html
[root@ssd html]# ls
info.php
```

Restart apache:

```
systemctl restart httpd.service
```

Now open up your browser and access that newly created php page:

http://xxx.xxx.xxx.xxx/info.php

You'll see a page similar to this one:

PHP Version 5.4.16

System	Linux ssd.servermom.org 2.6.32-042stab092.3 #1 SMP Sun Jul 20 13:27:24 MSK 2014 x86_64
Build Date	Aug 6 2014 13:13:24
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php.d
Additional .ini files parsed	/etc/php.d/curl.ini, /etc/php.d/dom.ini, /etc/php.d/fileinfo.ini, /etc/php.d/gd.ini, /etc/php.d/intl.ini, /etc/php.d/json.ini, /etc/php.d/mysql.ini, /etc/php.d/mysqli.ini, /etc/php.d/pdo.ini, /etc/php.d/pdo_mysql.ini, /etc/php.d/pdo_sqlite.ini, /etc/php.d/phar.ini, /etc/php.d/posix.ini, /etc/php.d/sqlite3.ini, /etc/php.d/sysvmsg.ini, /etc/php.d/sysvsem.ini, /etc/php.d/sysvshm.ini, /etc/php.d/wddx.ini, /etc/php.d/xmlreader.ini, /etc/php.d/xmlrpc.ini, /etc/php.d/xmlwriter.ini, /etc/php.d/xsl.ini, /etc/php.d/zip.ini

Installing PhpMyAdmin (see another method if this method not work for you)

Step 10 – Now your server has Apace, PHP and MariaDB installed. It means it should be OK now to install PhpMyAdmin, a popular web-based database management system so you can easily manage your database without having to login via SSH and issuing several command lines. Unluckily, this piece of awesome software is not available in CentOS 7.0 default repositories. In this case you have to add / enable third-party repo like **EPEL** or **RPMForge**.

Method #1: RPMForge

First, download the rpm file.

```
wget http://pkgs.repoforge.org/rpmforge-release/rpmforge-release-0.5.3-1.el7.rf.x86_64.rpm
```

```
[root@ssd ~]# yum http://pkgs.repoforge.org/rpmforge-release/rpmforge-release-0.5.3-1.el7.rf.x86_64.
rpm
Loaded plugins: fastestmirror
No such command: http://pkgs.repoforge.org/rpmforge-release/rpmforge-release-0.5.3-1.el7.rf.x86_64.r
pm. Please use /usr/bin/yum --help
[root@ssd ~]# wget http://pkgs.repoforge.org/rpmforge-release/rpmforge-release-0.5.3-1.el7.rf.x86_64
.rpm
--2014-08-16 11:22:14--  http://pkgs.repoforge.org/rpmforge-release/rpmforge-release-0.5.3-1.el7.rf.
x86_64.rpm
Resolving pkgs.repoforge.org (pkgs.repoforge.org)... 78.46.17.228
Connecting to pkgs.repoforge.org (pkgs.repoforge.org)|78.46.17.228|:80... connected.
HTTP request sent, awaiting response... 302 Found
Location: http://rpmforge.sw.be/redhat/el7/en/x86_64/rpmforge/RPMS/rpmforge-release-0.5.3-1.el7.rf.x
86_64.rpm [following]
--2014-08-16 11:22:14--  http://rpmforge.sw.be/redhat/el7/en/x86_64/rpmforge/RPMS/rpmforge-release-0
.5.3-1.el7.rf.x86_64.rpm
Resolving rpmforge.sw.be (rpmforge.sw.be)... 78.46.17.228
Connecting to rpmforge.sw.be (rpmforge.sw.be)|78.46.17.228|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: http://tree.repoforge.org/redhat/el7/en/x86_64/rpmforge/RPMS/rpmforge-release-0
rf.x86_64.rpm [following]
```

then enable the repository and delete the .rpm file as it is not needed again.

```
rpm -ivh rpmforge-release-*

rm rpmforge-release-*
```

```
[root@ssd ~]# rpm -ivh rpmforge-release-*
warning: rpmforge-release-0.5.3-1.el7.rf.x86_64.rpm: Header V3 DSA/SHA1
NOKEY
Preparing...                          ##################################
Updating / installing...
   1:rpmforge-release-0.5.3-1.el7.rf  ##################################
[root@ssd ~]# rm rpmforge-release-*
rm: remove regular file 'rpmforge-release-0.5.3-1.el7.rf.x86_
[root@ssd ~]#
```

Method #2: EPEL

Download the .rpm file:

```
wget http://download.fedoraproject.org/pub/epel/beta/7/x86_64/epel-release-7-
0.2.noarch.rpm
```

The url above is still its beta version. However if the repository is out of beta status, the link most likely will be different. In case that happens, you can find out its latest download url at Fedora Project website.

enable the repository and delete the .rpm file:

```
rpm -ivh epel-release*

rm epel-release*
```

Step 11 – Next, install it using yum again:

```
yum install phpmyadmin -y
```

screenshot:

```
[root@ssd ~]# rpm -ivh epel-release*
warning: epel-release-7-0.2.noarch.rpm: Header V3 RSA/SHA256 Signature, key I
Preparing...                          ################################# [100%]
Updating / installing...
   1:epel-release-7-0.2               ################################# [100%]
[root@ssd ~]# yum install phpmyadmin -y
Loaded plugins: fastestmirror
epel/x86_64/metalink
epel
(1/2): epel/x86_64/group_gz
(2/2): epel/x86_64/primary_db
Loading mirror speeds from cached hostfile
 * base: repos.dfw.quadranet.com
 * epel: linux.mirrors.es.net
 * extras: mirrors.easynews.com
 * rpmforge: mirror.hmc.edu
 * updates: mirrors.kernel.org
Resolving Dependencies
--> Running transaction check
---> Package phpMyAdmin.noarch 0:4.2.7-1.el7 will be installed
--> Processing Dependency: php-mcrypt >= 5.3.0 for package: phpMyAdmin-4.2.7-
--> Processing Dependency: php-mbstring >= 5.3.0 for package: phpMyAdmin-4.2.
--> Processing Dependency: php-tcpdf-dejavu-sans-fonts for package: phpMyAdmi
--> Processing Dependency: php-tcpdf for package: phpMyAdmin-4.2.7-1.el7.noar
--> Processing Dependency: php-php-gettext for package: phpMyAdmin
```

Step 12 – That's it. Now you also have phpMyAdmin (PMA) installed but you should be better if you change its default configuration before using it. First, you'll need to backup default PMA's config file:

```
cp /etc/httpd/conf.d/phpMyAdmin.conf /etc/httpd/conf.d/phpMyAdmin.conf.old
```

then edit file phpMyAdmin.conf file using your favorite editor. In this example I use Nano editor:

```
nano /etc/httpd/conf.d/phpMyAdmin.conf
```

Step 13 – You'll now see the content of phpMyAdmin.conf, next you have to allow connections from remote hosts by editing few lines inside section **<Directory "/usr/share/phpMyAdmin">**.

Before changes:

```
<Directory /usr/share/phpMyAdmin/>
   <IfModule mod_authz_core.c>
     # Apache 2.4
     <RequireAny>
       Require ip 127.0.0.1
       Require ip ::1
     </RequireAny>
   </IfModule>
```

After:

```
<Directory /usr/share/phpMyAdmin/>
   <IfModule mod_authz_core.c>
     # Apache 2.4
     <RequireAny>
       Require all granted
     </RequireAny>
   </IfModule>
   <IfModule !mod_authz_core.c>
     # Apache 2.2
```

Also you'll need to edit few lines next:

Before:

```
     # Apache 2.4
     <RequireAny>
       Require ip all granted
     </RequireAny>
   </IfModule>
   <IfModule !mod_authz_core.c>
     # Apache 2.2
     Order Deny,Allow
     Deny from All
     Allow from 127.0.0.1
     Allow from ::1
   </IfModule>
</Directory>
```

After:

```
     # Apache 2.4
     <RequireAny>
       Require ip all granted
     </RequireAny>
   </IfModule>
   <IfModule !mod_authz_core.c>
     # Apache 2.2
     #Order Deny,Allow
     #Deny from All
     AllowOverride None
     Options None
     Allow from All
     Require all granted
   </IfModule>
</Directory>
```

Shortly it should look like this:

```
<Directory /usr/share/phpMyAdmin/>

   <IfModule mod_authz_core.c>

     # Apache 2.4

     <RequireAny>
```

```
        Require all granted
      </RequireAny>
    </IfModule>
    <IfModule !mod_authz_core.c>
      # Apache 2.2
      #Order Deny,Allow
      #Deny from All
      AllowOverride None
      Options None
      Allow from All
      Require all granted
    </IfModule>
</Directory>
```

Once done, save and exit editor (In Nano it is Control+O then Control+X).

Step 14 – Restart Apache again:

```
systemctl restart httpd.service
```

Now you can test opening PMA on your browser via your server's IP address:

http://xxx.xxx.xxx.xxx/phpmyadmin

and default login page of phpMyAdmin should be displayed:

That's all. Now you can host your websites or blogs in that server, even WordPress.

Do not forget to follow me on twitter to get faster update or download my official Android app. Enjoy..

phpMyAdmin configuration working methods

How To Install and Secure phpMyAdmin with Apache on a CentOS 7 Server

Introduction

Relational database management systems like MySQL and MariaDB are needed for a significant portion of web sites and applications. However, not all users feel comfortable administering their data from the command line.

To solve this problem, a project called phpMyAdmin was created in order to offer an alternative in the form of a web-based management interface. In this guide, we will demonstrate how to install and secure a phpMyAdmin configuration on a CentOS 7 server. We will build this setup on top of the Apache web server, the most popular web server in the world.

Prerequisites

Before we begin, there are a few requirements that need to be settled.

To ensure that you have a solid base to build this system upon, you should run through our initial server setup guide for CentOS 7. Among other things, this will walk you through setting up a non-root user with `sudo` access for administrative commands.

The second prerequisite that must be fulfilled in order to start on this guide is to install a LAMP (Linux, Apache, MariaDB, and PHP) stack on your CentOS 7 server. This is the platform that we will use to serve our phpMyAdmin interface (MariaDB is also the database management software that we are wishing to manage). If you do not yet have a LAMP installation on your server, follow our tutorial on installing LAMP on CentOS 7.

When your server is in a properly functioning state after following these guides, you can continue on with the rest of this page.

Step One — Install phpMyAdmin

With our LAMP platform already in place, we can begin right away with installing the phpMyAdmin software. Unfortunately, phpMyAdmin is not available in CentOS 7's default repository.

To get the packages we need, we'll have to add an additional repo to our system. The EPEL repo (**E**xtra **P**ackages for **E**nterprise **L**inux) contains many additional packages, including the phpMyAdmin package we are looking for.

The EPEL repository can be made available to your server by installing a special package called `epel-release`. This will reconfigure your repository list and give you access to the EPEL packages.

To install, just type:

```
sudo yum install epel-release
```

Now that the EPEL repo is configured, you can install the phpMyAdmin package using the `yum` packaging system by typing:

```
sudo yum install phpmyadmin
```

The installation will now complete. The installation included an Apache configuration file that has already been put into place. We will need to modify this a bit to get it to work correctly for our installation.

Open the file in your text editor now so that we can make a few changes:

```
sudo nano /etc/httpd/conf.d/phpMyAdmin.conf
```

Inside, we see some directory blocks with some conditional logic to explain the access policy for our directory. There are two distinct directories that are defined, and within these, configurations that will be valid for both Apache 2.2 and Apache 2.4 (which we are running).

Currently, this setup is configured to deny access to any connection not being made from the server itself. Since we are working on our server remotely, we need to modify some lines to specify the IP address of your *home* connection.

Change any lines that read `Require ip 127.0.0.1` or `Allow from 127.0.0.1` to refer to your home connection's IP address. If you need help finding the IP address of your home connection, check out the next section. There should be four locations in the file that must be changed:

```
. . .
Require ip your_workstation_IP_address
. . .
Allow from your_workstation_IP_address
. . .
Require ip your_workstation_IP_address
. . .
Allow from your_workstation_IP_address
. . .
```

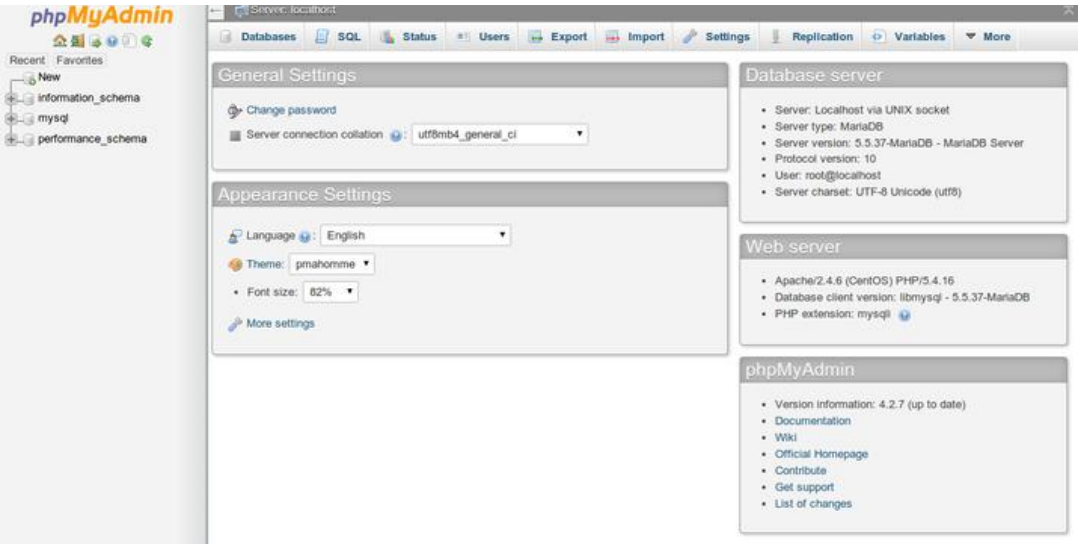
When you are finished, restart the Apache web server to implement your modifications by typing:

```
sudo systemctl restart httpd.service
```

With that, our phpMyAdmin installation is now operational. To access the interface, go to your server's domain name or public IP address followed by `/phpMyAdmin`, in your web browser:

```
http://server_domain_or_IP/phpMyAdmin
```

To sign in, use a username/password pair of a valid MariaDB user. The `root` user and the MariaDB administrative password is a good choice to get started. You will then be able to access the administrative interface:



Find Your IP Address

You will need to know the IP address of the computer you are using to access your databases in order to complete the step above. This is a security precaution so that unauthorized people cannot connect to your server.

Note: This is *not* the IP address of your VPS, it is the IP address of your home or work computer.

You can find out how the greater web sees your IP address by visiting one of these sites in your web browser:

- What's My IP Address?
- What's My IP?
- My IP Address

Compare a few different sites and make sure they all give you the same value. Use this value in the configuration file above.

Step Two — Secure your phpMyAdmin Instance

The phpMyAdmin instance installed on our server should be completely usable at this point. However, by installing a web interface, we have exposed our MySQL system to the outside world.

Even with the included authentication screen, this is quite a problem. Because of phpMyAdmin's popularity combined with the large amount of data it provides access to, installations like these are common targets for attackers.

We will implement two simple strategies to lessen the chances of our installation being targeted and compromised. We will change the location of the interface from `/phpMyAdmin` to something else to sidestep some of the automated bot brute-force attempts. We will also create an additional, web server-level authentication gateway that must be passed before even getting to the phpMyAdmin login screen.

Changing the Application's Access Location

In order for our Apache web server to work with phpMyAdmin, our phpMyAdmin Apache configuration file uses an alias to point to the directory location of the files.

To change the URL where our phpMyAdmin interface can be accessed, we simply need to rename the alias. Open the phpMyAdmin Apache configuration file now:

```
sudo nano /etc/httpd/conf.d/phpMyAdmin.conf
```

Toward the top of the file, you will see two lines that look like this:

```
Alias /phpMyAdmin /usr/share/phpMyAdmin
Alias /phpmyadmin /usr/share/phpMyAdmin
```

These two lines are our aliases, which means that if we access our site's domain name or IP address, followed by either `/phpMyAdmin` or `/phpmyadmin`, we will be served the content at `/usr/share/phpMyAdmin`.

We want to disable these specific aliases since they are heavily targeted by bots and malicious users. Instead, we should decide on our own alias. It should be easy to remember, but not easy to guess. It shouldn't indicate the purpose of the URL location. In our case, we'll go with `/nothingtosee`.

To apply our intended changes, we should remove or comment out the existing lines and add our own:

```
# Alias /phpMyAdmin /usr/share/phpMyAdmin
# Alias /phpmyadmin /usr/share/phpMyAdmin
Alias /nothingtosee /usr/share/phpMyAdmin
```

When you are finished, save and close the file.

To implement the changes, restart the web service:

```
sudo systemctl restart httpd.service
```

Now, if you go to the previous location of your phpMyAdmin installation, you will get a 404 error:

```
http://server_domain_or_IP/phpMyAdmin
```

:

Not Found

The requested URL /phpMyAdmin was not found on this server.

However, your phpMyAdmin interface will be available at the new location we selected:

```
http://server_domain_or_IP/nothingtosee
```

Setting up a Web Server Authentication Gate

The next feature we wanted for our installation was an authentication prompt that a user would be required to pass before ever seeing the phpMyAdmin login screen.

Fortunately, most web servers, including Apache, provide this capability natively. We will just need to modify our Apache configuration file to use an authorization file.

Open the phpMyAdmin Apache configuration file in your text editor again:

```
sudo nano /etc/httpd/conf.d/phpMyAdmin.conf
```

Within the `/usr/share/phpMyAdmin` directory block, but outside of any of the blocks inside, we need to add an override directive. It will look like this:

```
. . .
<Directory /usr/share/phpMyAdmin/>
   AllowOverride All
   <IfModule mod_authz_core.c>
   . . .
</Directory>
. . .
```

This will allow us to specify additional configuration details in a file called `.htaccess` located *within* the phpMyAdmin directory itself. We will use this file to set up our password authentication.

Save and close the file when you are finished.

Restart the web service to implement this change:

```
sudo systemctl restart httpd.service
```

Create an .htaccess File

Now that we have the override directive in our configuration, Apache will look for a file called `.htaccess` within the `/usr/share/phpMyAdmin` directory. If it finds one, it will use the directives contained within to supplement its previous configuration data.

Our next step is to create the `.htaccess` file within that directory. Use your text editor to do so now:

```
sudo nano /usr/share/phpMyAdmin/.htaccess
```

Within this file, we need to enter the following information:

```
AuthType Basic
AuthName "Admin Login"
AuthUserFile /etc/httpd/pma_pass
Require valid-user
```

Let's go over what each of these lines mean:

- **AuthType Basic**: This line specifies the authentication type that we are implementing. This type will implement password authentication using a password file.
- **AuthName**: This sets the message for the authentication dialog box. You should keep this generic so that unauthorized users won't gain knowledge about what is being protected.
- **AuthUserFile**: This sets the location of the actual password file that will be used for authentication. This should be outside of the directories that are being served. We will create this file in a moment.
- **Require valid-user**: This specifies that only authenticated users should be given access to this resource. This is what actually stops unauthorized users from entering.

When you are finished entering this information, save and close the file.

Create the Password File for Authentication

Now that we have specified the location for our password file through the use of the `AuthUserFile` directive in our `.htaccess` file, we need to create and populate the password file.

This can be accomplished through the use of an Apache utility called `htpasswd`. We invoke the command by passing it the location where we would like to create the file and the username we would like to enter authentication details for:

```
sudo htpasswd -c /etc/httpd/pma_pass username
```

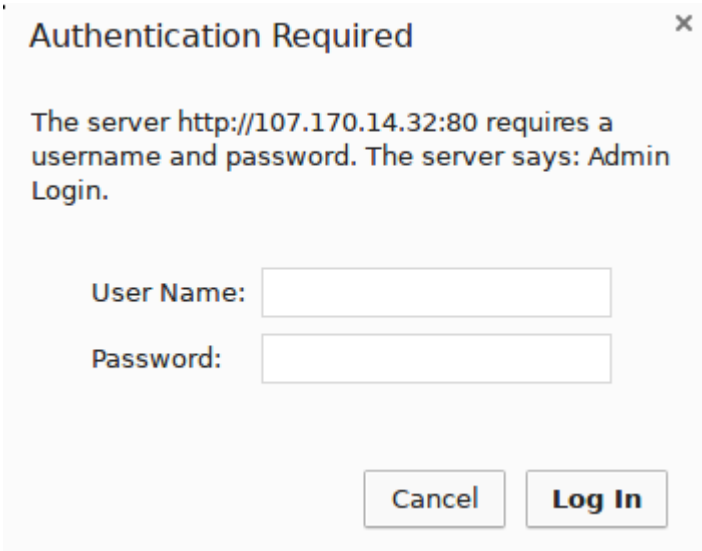
The `-c` flag indicates that this will create an initial file. The directory location is the path and filename that will be used for the file. The username is the first user we would like to add. You will be prompted to enter and confirm a password for the user.

If you want to add additional users to authenticate, you can call the same command again **without** the `-c` flag, and with a new username:

```
sudo htpasswd /etc/httpd/pma_pass seconduser
```

With our password file created, an authentication gateway has been implemented and we should now see a password prompt the next time we visit our site:

```
http://server_domain_or_IP/nothingtosee
```



Once you enter your credentials, you will be taken to the normal phpMyAdmin login page. This added layer of protection will help keep your MySQL logs clean of authentication attempts in addition to the added security benefit.

Conclusion

You can now manage your MySQL databases from a reasonably secure web interface. This UI exposes most of the functionality that is available from the MySQL command prompt. You can view databases and schema, execute queries, and create new data sets and structures.

The configuration file will be like that

```
Alias /phpMyAdmin /usr/share/phpMyAdmin
Alias /phpmyadmin /usr/share/phpMyAdmin

<Directory /usr/share/phpMyAdmin/>
  AddDefaultCharset UTF-8

  <IfModule mod_authz_core.c>
    # Apache 2.4
    <RequireAny>
      # Require ip 172.16.151.15
      # Require ip 172.16.151.26
      # Require ip ::1
```

```
      Require all granted
    </RequireAny>
  </IfModule>
  <IfModule !mod_authz_core.c>
    # Apache 2.2
    Order Deny,Allow
    # Deny from All
    Allow from 172.16.151.15
    Allow from 172.16.151.26
    Allow from ::1
  </IfModule>
</Directory>

<Directory /usr/share/phpMyAdmin/setup/>
  <IfModule mod_authz_core.c>
    # Apache 2.4
    <RequireAny>
      Require ip 172.16.151.15
      Require ip 172.16.151.26
      Require ip ::1
    </RequireAny>
  </IfModule>
  <IfModule !mod_authz_core.c>
    # Apache 2.2
    Order Deny,Allow
    # Deny from All
    Allow from 172.16.151.15
    Allow from 172.16.151.26
    Allow from ::1
  </IfModule>
</Directory>

# These directories do not require access over HTTP - taken from the original
# phpMyAdmin upstream tarball
#
<Directory /usr/share/phpMyAdmin/libraries/>
   Order Deny,Allow
   # Deny from All
   # Allow from None
   Allow from all
</Directory>

<Directory /usr/share/phpMyAdmin/setup/lib/>
   # Order Deny,Allow
   # Deny from All
   # Allow from None
```

```
   Allow from all
</Directory>

<Directory /usr/share/phpMyAdmin/setup/frames/>
   Order Deny,Allow
   # Deny from All
   # Allow from None
   Allow from all
</Directory>

# This configuration prevents mod_security at phpMyAdmin directories from
# filtering SQL etc.  This may break your mod_security implementation.
#
#<IfModule mod_security.c>
#    <Directory /usr/share/phpMyAdmin/>
#        SecRuleInheritance Off
#    </Directory>
#</IfModule>
```

Now install WordPress……

Happy Server Configuration

Engr. Md. Nazim Uddin

nazim.cse.kuet@gmail.com