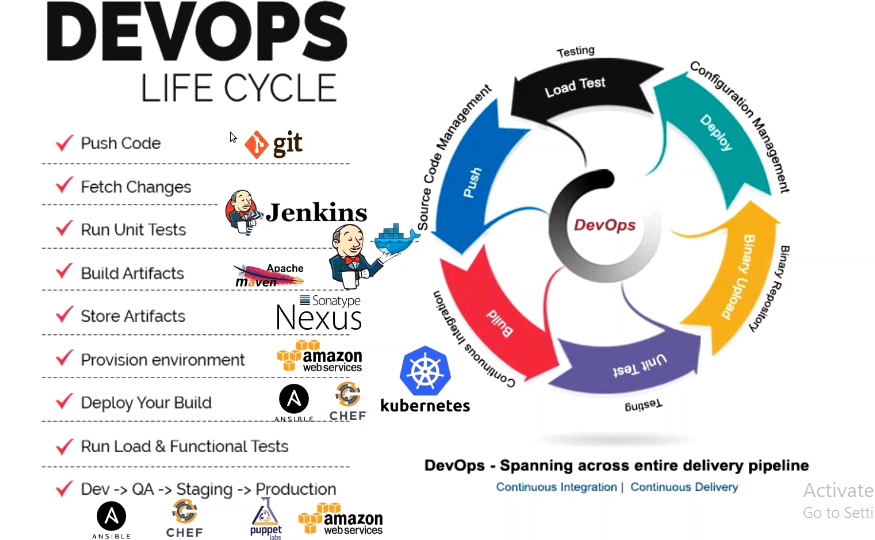
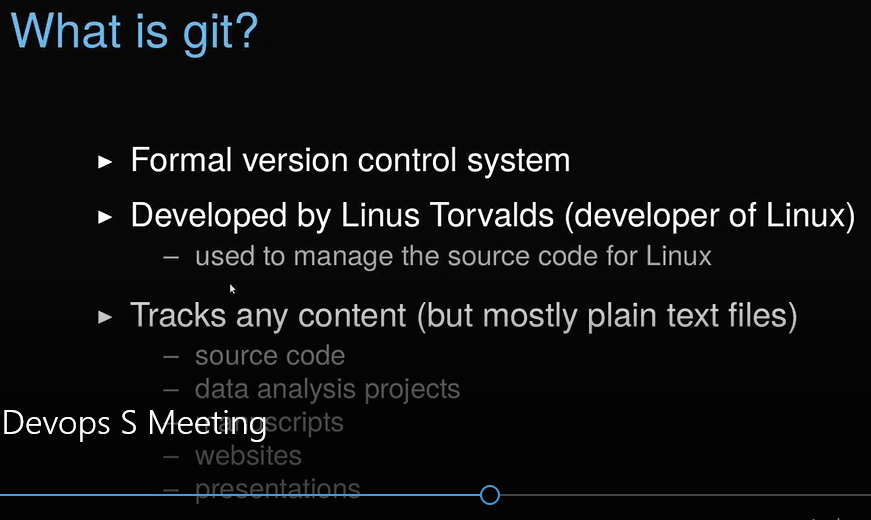
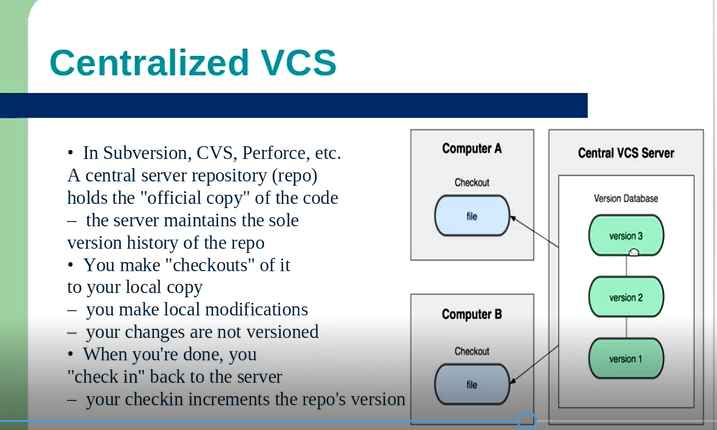
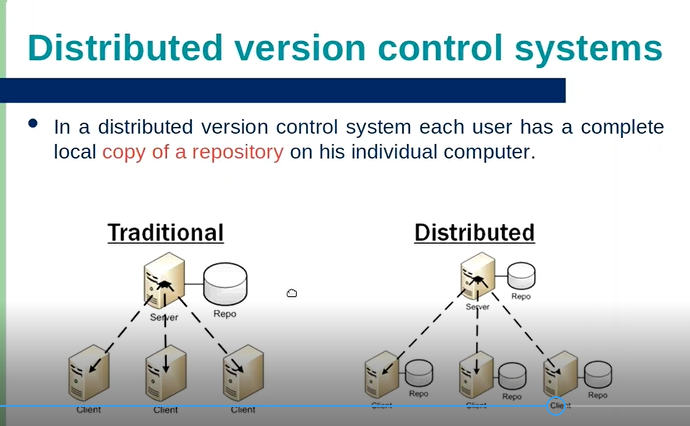
***GitHUB***

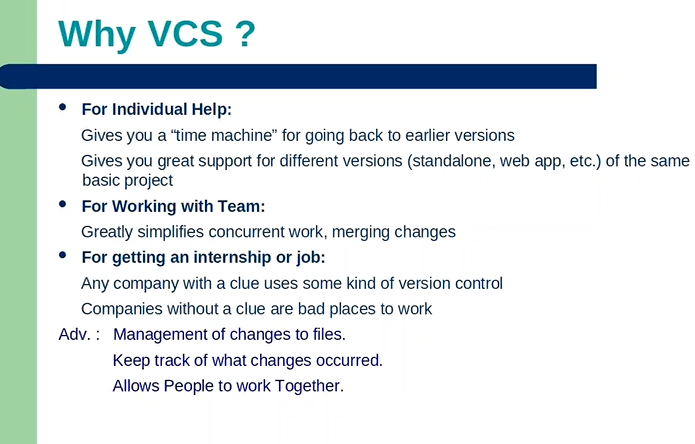


It begins when the developer push the code into a version control system. Fetch the code, execute some tests, generate the artefacts,









root@ubuntu-xenial:~/Labs/gitlearn/linuxgeek-newmeaning# history

1 touch file1

2 cp file1 file1.`(date +%F)`

5 apt-get install git

6 mkdir -p Labs/gitlearn

7 cd Labs/gitlearn/

8 git clone <https://github.com/anchaubey/linuxgeek-source.git> --recursive

10 cd linuxgeek-source/

12 cat README.md

13 cd .git/

17 cat .git/config

18 mkdir dir1 dir2 dir2

19 ll

20 mkdir dir3

21 touch ll

22 ls -la

23 rm -rf dir\*

24 ll

25 mkdir dir1 dir2 dir3

26 touch file1 file2 file3

27 ll

28 cp file\* dir\*

29 cp file\* dir1

30 cp file\* dir2

31 cp file\* dir3

32 ll dir2/

33 ls

34 git add .

35 ls -la

36 touch file4

37 git add file4

38 ls -la

39 git commit -m "first commit"

40 git config --gloabal user.email "ankit.chaubey@gmail.com"

41 git config --global user.email "ankit.chaubey@gmail.com"

42 git config --global user.name "anchaubey"

43 git commit -m "first commit"

44 git log

45 date

46 git status

47 git log

48 git config --system --list

49 pwd

50 ll /root/Labs/gitlearn/

51 ll /root/Labs/

52 ll /root/

53 ll

54 pwd

55 ln -s /root/Labs/gitlearn/linuxgeek-source/.git/config /etc/gitconfig

56 git config --system --list

57 cat .git/config

58 git config --global user.name "Ankit Chaubey"

59 git config --global user.email "ankit.chaubey@gmail.com"

60 git config --global --list

61 git config --global core.editor vim

62 git config --global --list

63 apt-get install vim

64 vim test

65 rm test

66 ll

67 git config --list

68 git status

69 git log --stat

70 git log --shortstat

73 rm -rf Labs/\*

74 ll

75 rm -rf Labs

76 ll

77 mkdir Labs

78 mkdir Labs/gitlearn

79 cd Labs/gitlearn/

80 ll /etc/gitconfig

81 rm /etc/gitconfig

82 ll /etc/gitconfig

83 pwd

84 git clone https://github.com/anchaubey/linuxgeek-newmeaning.git

85 ll

86 cd linuxgeek-newmeaning/

87 ll

88 ls -la

89 mkdir dir1 dir2 dir3

90 touch file{1..3}

91 ll

92 cp file\* dir1/

93 cp file\* dir2/

94 cp file\* dir3/

95 ll dir1/

96 pwd

97 git add .

98 git commit -m "latest"

99 git log

100 date

101 git log

102 git rm file2

103 ll

104 git add .

105 git commit -m "removed file2"

106 git log

107 git mv file3 file33

108 git add .

109 git commit -m "file3 renamed to file33"

110 git log

111 cat .git/config

112 git push origin master

113 git log

114 mkdir kandhenu

115 git add kandhenu/

116 git commit -m "added kamdhenu"

117 history | grep push

118 git push origin master

119 history

120 git add .

121 git push origin master

122 ll

123 cat file1

124 ls

125 git pull

126 cat file1

127 git log

128 date

129 git log

130 pwd

131 git pull

132 git branch

133 git checkout development

134 git branch

135 ll

136 git rm file1

137 vi file33

138 ll

139 git add .

140 git commit -m "made changes to file33"

141 ls

142 git status

143 git checkout master

144 cat file33

145 git branch

146 git checkout development

147 git push origin development

148 get log --oneline

150 git log

152 git diff 8f1688e..103282e

155 git checkout master

156 git branch

157 git merge development

158 ll

159 touch file1

160 rm file1

161 vi file22

162 ll

163 git add .

164 git commit -m "added file22"

165 git branch

166 git log

167 cat file22

168 git log

169 date

170 git log

171 git log --oneline

172 git reset --hard 8f1688e

173 ll

174 history

***The below is from Imran Sir:-***

1. Sign up for the github account.

2. Create public repository.

3. Copy the URL of the public repo.

4. Install git client on your system.

W

​indows:

      - ​

I

​nstall git software

<https://git-scm.com/download/win>

      - Open Git Bash prompt

​L

inux​

:-

​

     # sudo apt-get install git (Ubuntu) or

     # yum install git (Centos) or

5. Clone git repository on your local system

     # git clone <Git Repo URL>

6. Create few directories and files in the Repo directory. (Directories should not be empty)

7. Go inside the repo directory

    # cd <Repo name>

8. Update the index using the current content found in the working tree.

  # git add .

9.  Stores the current contents of the index in a new commit along with a log message from the user describing the changes.

   # git commit -m "<Put some message>"

10. Push the changes to the github.

    # git push origin master

11. Make changes to some files from github UI and pull the changes back to the local repo

    # git pull

12. Create new branch from master branch and pull the changes again to local repo.

    # git pull

    Switch to new branch

    # git status (Shows the current branch)

    # git checkout <branch name>

    # git status

13. Make some changes to files and push it back to the latest branch

     # git add .

     # git commit -m "<Some message>"

     # git push origin <branchname>

​

14.Remove/Rename the content and push

    # git rm <filename>

    # git mv <filename>​ <newnameoffile>

    # git commit -m "<message>"

    # git push

***Vagrant:-***

$ vagrant plugin list

$ vagrant plugin install vagrant-vbguest

c**onfig.vm.synced\_folder** ".", "/var/www/html",type:"virtualbox" (dot is the current working directory of vagrant machine)

Few commands

vagrant reload (after changing any vm setting)

vagrant destroy (will remove all traces of the guest machine from system)

vagrant box list

vagrant box add centos/7 (we can download a box to use it later)

vagrant list-commands - will list all vagrant commands

$ vagrant global-status (gives status of all vms)

**Multiple VM’s**

**$ cat Vagrantfile**

**Vagrant.configure("2") do |config|**

**config.vm.define "web" do |web|**

**web.vm.box = "ubuntu/trusty64"**

**web.vm.hostname = 'web'**

**end**

**config.vm.define "db" do |db|**

**db.vm.box = "nrel/CentOS-6.5-x86\_64"**

**db.vm.hostname = 'db'**

**end**

**end**

**How to configure vagrant to use a shell script saved at your default location and also using port forwarding:-**

# Enable provisioning with a shell script. Additional provisioners such as

# Puppet, Chef, Ansible, Salt, and Docker are also available. Please see the

# documentation for more information about their specific syntax and use.

# config.vm.provision "shell", inline: <<-SHELL

# apt-get update

# apt-get install -y apache2

# SHELL

**config.vm.provision :shell, path: "bootstrap.sh"**

config.vm.network :forwarded\_port, guest: 80, host: 4567

end

**All in One where we are getting 2 vm’s up togerther with script and port forwarding also: -**

**$ cat Vagrantfile**

**Vagrant.configure("2") do |config|**

**config.vm.define "web" do |web|**

**web.vm.box = "ubuntu/trusty64"**

**web.vm.hostname = 'web'**

**web.vm.network "public\_network", bridge: "wlo1"**

**web.vm.provision :shell, path: "bootstrap.sh"**

**web.vm.network :forwarded\_port, guest: 80, host: 4567**

**web.vm.provider :virtualbox do |v|**

**v.customize ["modifyvm", :id, "--memory", 512]**

**end**

**end**

**config.vm.define "db" do |db|**

**db.vm.box = "ubuntu/trusty64"**

**db.vm.hostname = 'db'**

**db.vm.network "public\_network", bridge: "wlo1"**

**db.vm.provider :virtualbox do |v|**

**v.customize ["modifyvm", :id, "--memory", 512]**

**end**

**end**

**end**

**for bridge networking:-**

config.vm.network "public\_network", bridge: "eth0"

**How to setup cron:-**

 crontab -l > mycron

 echo "00 09 \* \* 1-5 echo hello" >> mycron

 crontab mycron

 rm mycron

**Vagrant for Ansible:-**

#Webserver node, ubuntu OS

config.vm.define "web" do |web|

web.vm.box = "nrel/CentOS-6.5-x86\_64"

web.vm.hostname = 'web'

web.vm.network "public\_network"

end

#DBserver node, ubuntu OS

config.vm.define "db" do |db|

db.vm.box = "nrel/CentOS-6.5-x86\_64"

db.vm.hostname = 'db'

db.vm.network "public\_network"

end

#Control Server

config.vm.define "control" do |control|

control.vm.box = "ubuntu/trusty64"

control.vm.hostname = 'control'

control.vm.network "public\_network"

end

end

**VAGRANT FILE FOR PUPPET VM’S**

$ cat Vagrantfile

# -\*- mode: ruby -\*-

# vi: set ft=ruby :

# README

#

# Getting Started:

# 1. vagrant plugin install vagrant-hostmanager

# 2. vagrant up

# 8. vagrant ssh

#

# This should put you at the control host

# with access, by name, to other vms

Vagrant.configure(2) do |config|

config.hostmanager.enabled = true

config.vm.define "puppetmaster", primary: true do |h|

h.vm.box = "aerospike/centos-6.5"

h.vm.network "private\_network", ip: "192.168.8.10"

h.vm.provider :virtualbox do |vb|

vb.customize ["modifyvm", :id, "--memory", "2048"]

end

h.vm.hostname = 'puppetmaster'

h.vm.synced\_folder ".", "/etc/puppet"

end

config.vm.define "wiki" do |h|

h.vm.box = "aerospike/centos-6.5"

h.vm.network "private\_network", ip: "192.168.8.11"

h.vm.hostname = 'wiki'

end

config.vm.define "wikitest" do |h|

h.vm.box = "ubuntu/trusty64"

h.vm.network "private\_network", ip: "192.168.8.12"

h.vm.hostname = 'wikitest'

end

end

**How to install JAVA on Centos7**

**[root@dlp ~]# yum -y install java-1.8.0-openjdk java-1.8.0-openjdk-devel**

**# make sure the PATH**

**[root@dlp ~]# dirname $(readlink $(readlink $(which java)))**

**/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.102-1.b14.el7\_2.x86\_64/jre/bin**

**[root@dlp ~]# vi /etc/profile**

**# add follow to the end**

**export JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.102-1.b14.el7\_2.x86\_64**

**export PATH=$PATH:$JAVA\_HOME/bin**

**export CLASSPATH=.:$JAVA\_HOME/jre/lib:$JAVA\_HOME/lib:$JAVA\_HOME/lib/tools.jar**

**[root@dlp ~]# source /etc/profile**

**How to Install Java on ubuntu?**

add-apt-repository ppa:openjdk-r/ppa (first add PPA repository)

apt-get update

apt-get install openjdk-8-jdk

java -version

**Installing Jenkins website**

wget -q -O - https://pkg.jenkins.io/debian/jenkins-ci.org.key | sudo apt-key add -

sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'

sudo apt-get update

sudo apt-get install jenkins

https://github.com/wkhanvisualpath/VProfile

before building a project

need maven version 3.3 and java version 1.8

**Nexus works on port 8081 (Important)**

**Installing Nexus**

**25 yum search java**

**27 yum install java-1.8.0-openjdk.x86\_64**

**28 java -version**

**29 yum install wget -y**

**30 wget http://www.sonatype.org/downloads/nexus-latest-bundle.tar.gz**

**31 mv nexus-latest-bundle.tar.gz /usr/local/**

**32 cd /usr/local/**

**35 tar -xvzf nexus-latest-bundle.tar.gz**

**36 ll**

**37 cd nexus-2.14.5-02/**

**38 ll**

**39 cd bin/**

**40 ll**

**41 ./nexus start**

**42 export RUN\_AS\_USER=root**

**43 ./nexus start**

making log server and client

NFS server and client

DNS

jenkins Advanced tab- Proxy settings

**To find a process using a specific port user**

**[root@localhost ~]# fuser -v 80/tcp**

configuring multiple nginx first to redirect any website running on a specific port to port 80.

Go to /etc/nginx/sites-available directory and create a file tomcat.app and the below:-

server{

listen 80;

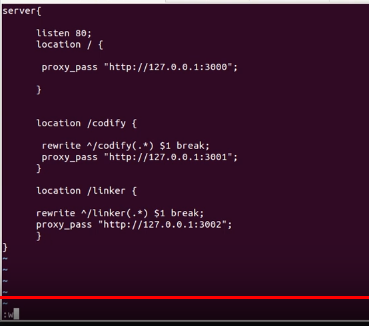
location / {

proxy\_pass "http://127.0.0.1:8080/manager/html";

}

}

This is how we add multiple sites pointing to different ports added to tomcat.app file:-



In the above we are serving tomcat default page on port which nginx redirects to port 80.

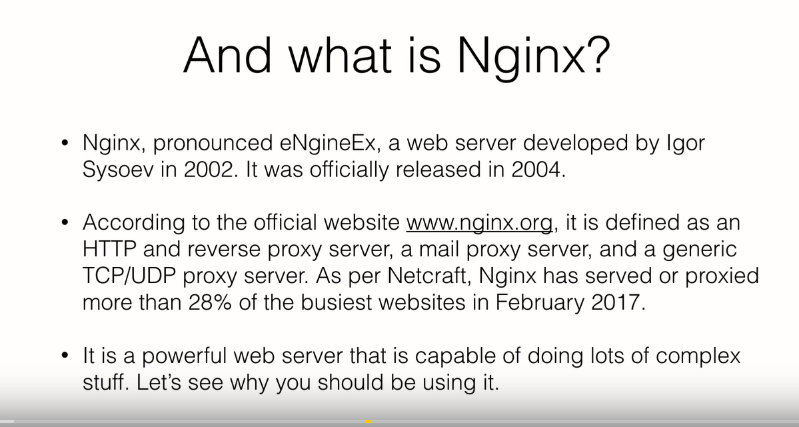
Also, we need to delete the /etc/nginx/sites-enabled/default file and would need to create a soft link of our tomcat.app file to this directory

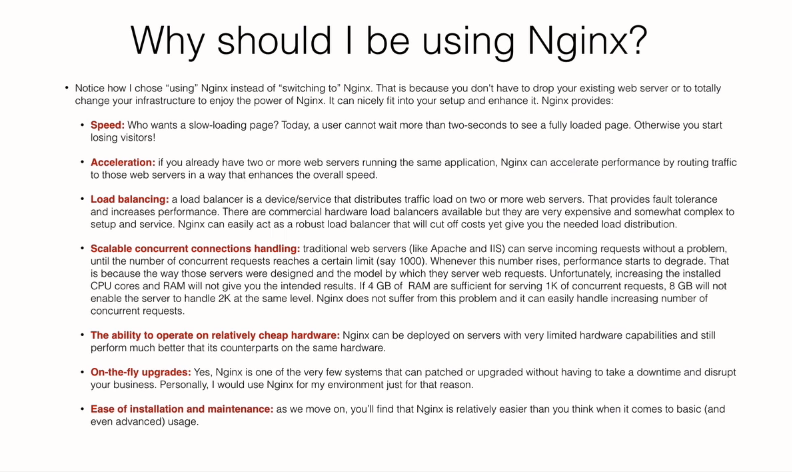
ln -s /etc/nginx/sites-available/tomcat.app /etc/nginx/sites-enabled/tomcat.app

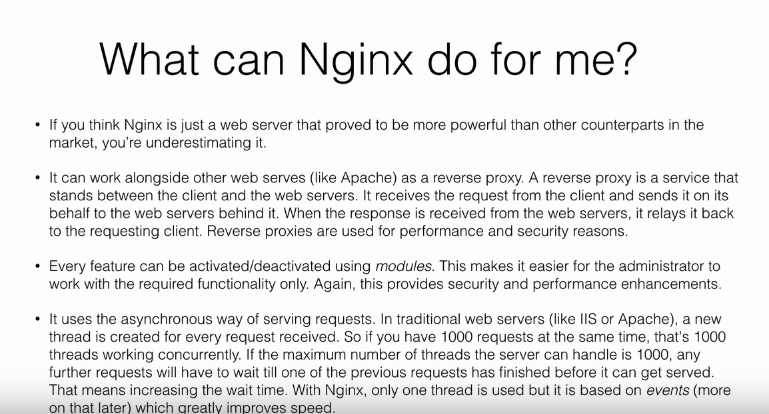
then restart nginx

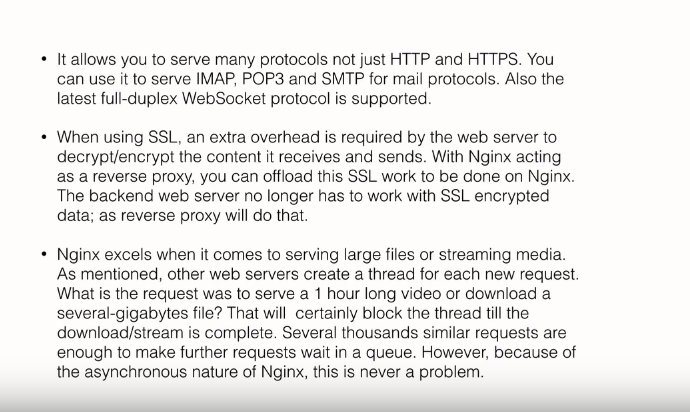
/etc/init.d/nginx restart

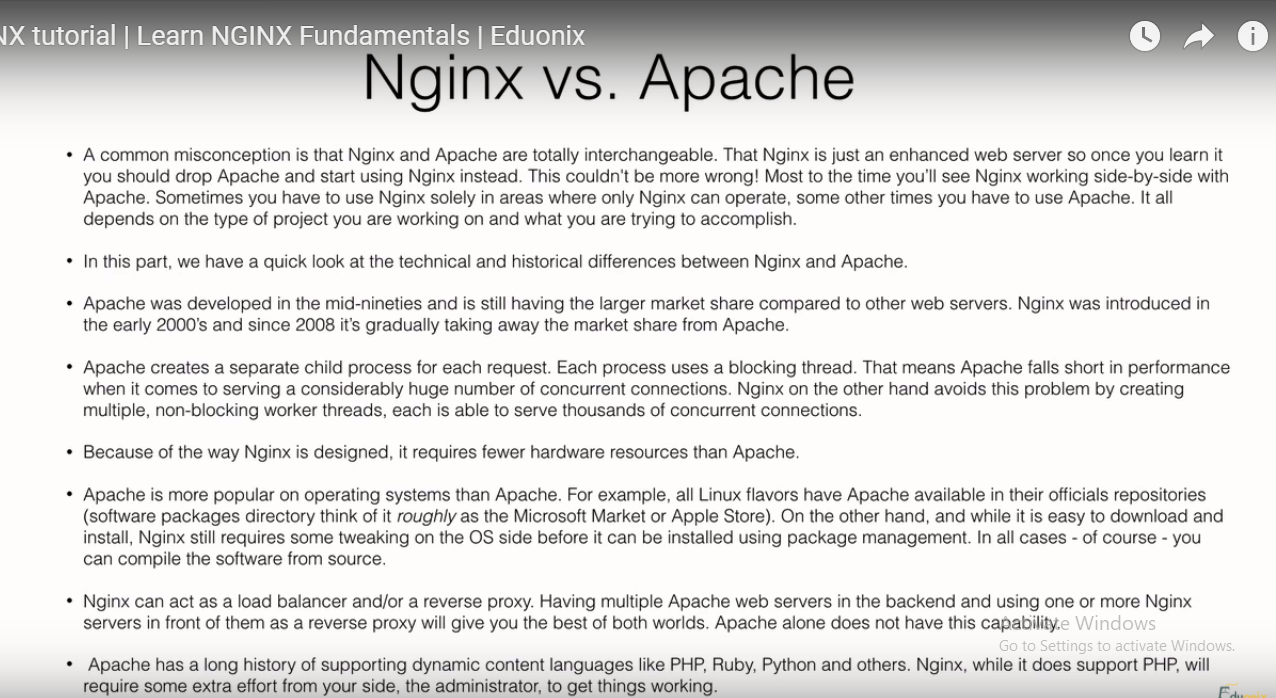
https://www.digitalocean.com/community/tutorials/how-to-configure-nginx-as-a-web-server-and-reverse-proxy-for-apache-on-one-ubuntu-16-04-server

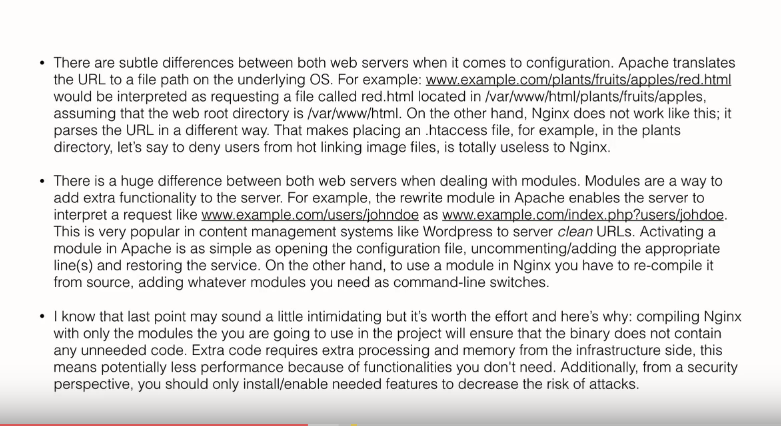


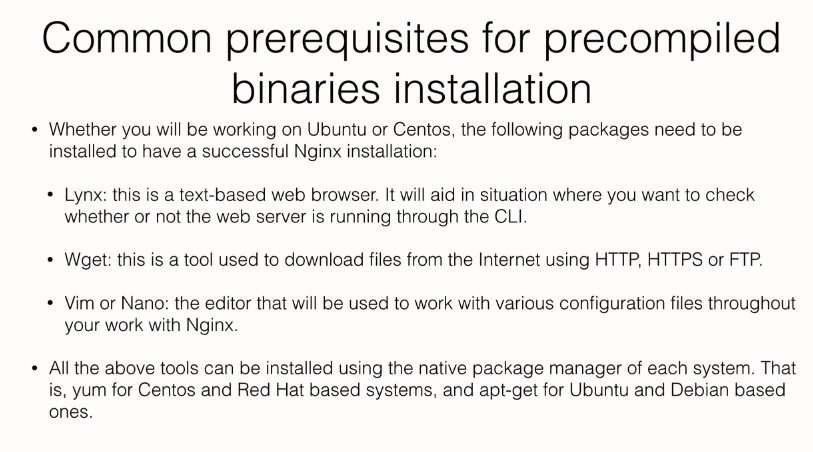


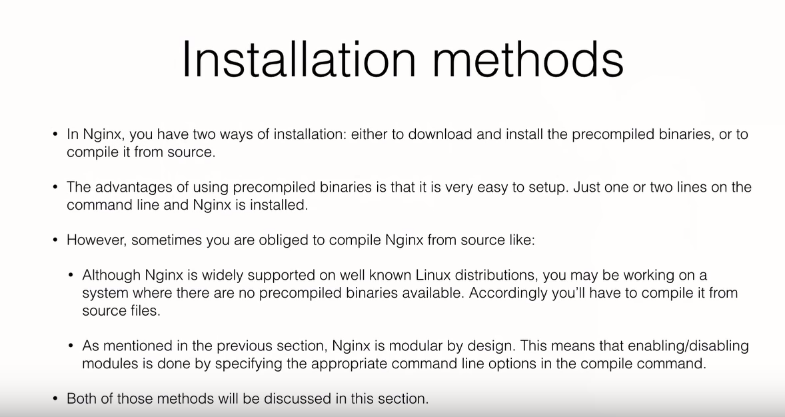










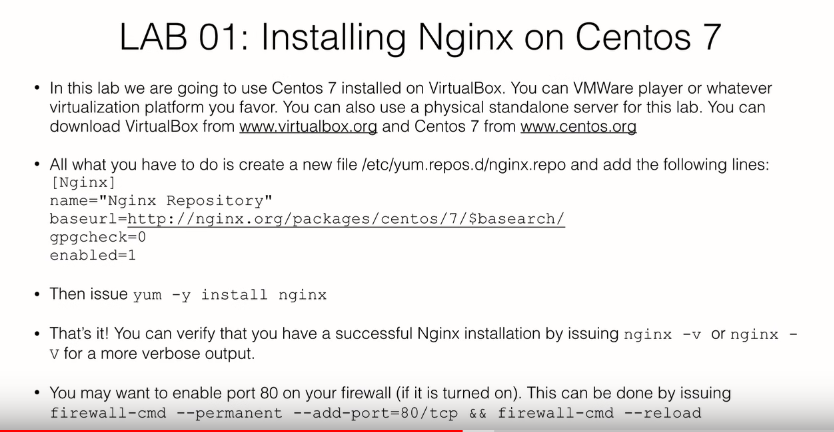


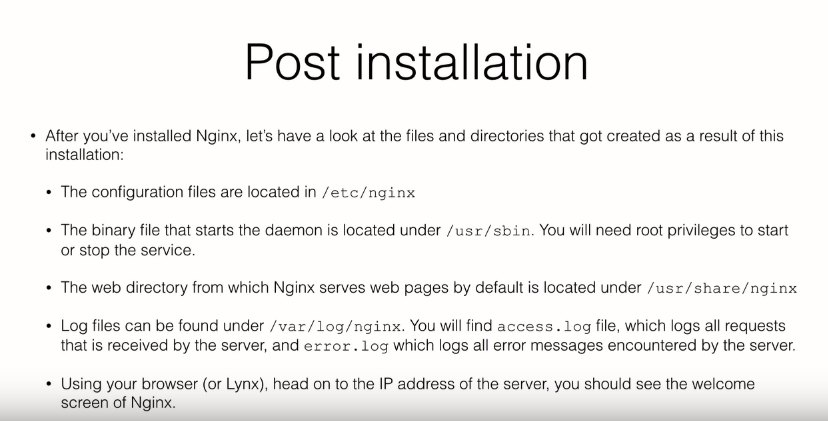
We have 2 servers ubuntu and centos

Yum install lynx wget vim on both the machines

Ubuntu VM- 192.168.38.77

Centos VM – 192.168.38.79





Service start/stop



