AWS:

https://us-east-2.console.aws.amazon.com/console/home?region=us-east-2#

Gyanvriksh login:

gvpl\_act1

gvipl@123

gvpl\_act2

gvipl@234

gvipl@321

TIKONA -> gvipl$123

Amazon default user:

ec2-user

pwd: LaxmanE2DevOps

sudo yum install httpd -> to install s/w in linux machine

raghuopsdev@gmail.com

Sub: GVK Morning 8 AM

Linux commands:

domain name : hostname -f

IP details : ipconfig

To know public IP : curl ifconfig.co

To check whether TCP/IP is properly confired or not: ping <loop back IP>

To check whether there is any n/w issue -> if some user from a perticular place is not able to access a website.

$trace

$traceroute google.co.in

To know the space: df -h

Present working directory: pwd

Current user: $whoami

To chk the logged in user details: $w

To switch to root user: $sudo su -

To know the all repositories(where we can save authorized softwares and data) list in the machine: # yum repolist

To know what are the softwares present int the repositories: # yum list

We can create a new repository if a s/w is not available in the existing repositories.

To know # cd /etc/yum.repos.d

ls -l

To create a file: touch abc.txt

ls -l -> gives type of file, permissions(user group others)

Change permissions: chmod 664 abc.txt -> 4->read 2-> write 1-> execute)

Change ownership: chown ec2-user:ec2-user abc.txt (user pwd)

To know all the users present: cat /etc/passwd

To know which type of shell we are using: echo $SHELL

Google -> linux cheat sheets

Start httpd server: service httpd start

To check IP address: curl ifconfig.co

**06/09/2018:**

$top -> gives you all details of various tasks running and CPU usage. (like task manager in windows)

* Swap is temporary memory of hard disk when RAM is full. It makes performance degradation.

$iostat -> give i/o statistics.

$ ps -ef | grep ssh ->

$nslookup -> gives ip address of a DNS name

$mkdir -p linux/practice/ssh -> to create folders inside other folders even the first folder is not there.

$cat abc.txt > abc2.txt -> creates a new file and copy the data into it.

$yum list | grep httpd

$yum install httpd

$netstat -plan | grep httpd -> to check on which port httpd is running.

**Where https is installed and index.htm???**

1. $which httpd
2. Check configuration file for root directory
3. Place your index.html here.

**Add DNS to an IP:**

Add IP address and DNS name of yours in C:\Windows\System32\drivers\etc\hosts file. It is used for local testing.

**Searching for files: find and locate???**

**Partitions:**

**Add a new volume and partition**

**Make directory and mount it.**

[root@default]# fdisk -l | grep '^Disk'

1. To partition the disk - /dev/sdb, enter:

**[root@default root]# fdisk /dev/sdb**

1. Format the new disk using mkfs.ext3 command:  
     
   To format Linux partions using ext2fs on the new disk, issue the following command:

|  |
| --- |
|  |
|  | [root@default]#mkfs.ext3 /dev/sdb1 |
| 1. Mount the new disk using the mount command:   First, you'll need to create a mount point. We'll use /disk1. This is where we'll mount /dev/sdb1. Enter the following commands:  |  | | --- | |  |      |  |  | | --- | --- | |  | [root@default]# mkdir /disk1 [root@default]# mount /dev/sdb1 /disk1 [root@default]# df -H | |  |

1. Edit /etc/fstab so the new drive will automatically mount to /disk1 on reboot

**#vi /etc/fstab**

**linux system admin cheat sheet**

**/dev/sdf**

/dev/xvdfansi

**07/09/2018:**

**Connecting two linux servers with ssh protocol**

* They will connected using ssh protocol.

we need to generate a key for the source using “$ssh-keygen -t rsa -b 4096

* Add this key in .ssh file of target linux machine’s .ssh/ authorized\_keys file.
* Now you can use $ssh <host\_name> to connect

**How to keep an IP address as a DNS in linux machine:**

**Add IP address with DNS name in hosts file using below command**

* **$sudo vi /etc/hosts**

**Hosting two dns’s on same httpd server with same port 80-> same ip**

**Read conf file details in /etc/httpd**

**#Yum install elinks -> to install linux browser**

**10/09/2018:**

**How to install one more apache:**

1. Download from apache site 2.2.3.
2. Download by using wget <web link>
3. Unzip using tar command
4. Go to that folder
5. Compile this by using global C compiler

#sh configure -> u will get failure as C compiler is not there, install it

1. #Yum install CC
2. Now execute step 5
3. #Make
4. #make install
5. #cd /usr/local/apache2 -> where apache was installed
6. Need to start this service by checking starter file in bin of installation location
7. #cd bin
8. #./apachectl start

Port already in use

1. #netstat -plan | grep http
2. #netstat -….
3. #cd conf
4. #cat httpd.conf | grep Listen
5. #vi httpd.conf
6. #/Listen -> enter to search Listen work
7. Edit the post number to some other
8. #cd ../bin
9. #./apachectl start
10. #ps …
11. Diff b/n http and https -> https will use a security certificate like CA entrust. This certificate encrypt the traffic with that website. 2048 RSA algorithm will encrypt 2048 times. Browser will check whether the certificate is provided by valid CA.
12. We can also generate our certificate using open SSL to test our application but browser can’t recognize it.
13. #cd /etc/http/conf
14. #yuminsttal openssl
15. #generate ssl certificate with proper command
16. #vi httpd.conf
17. Some where you can see all the Load Modules to support various languages and SSL certificates….
18. #cd../modules/
19. You can see all the modules supported by httpd
20. If SSL is not there need to install SSL module
21. #yum install mod\_ssl
22. There 3 types of installtions

* Yum install -> which takes from amazon existing repository
* RPM -> no need of internet and repository
* Download tar file->unzip->compile->install

1. #vi http.conf -> chk for SSL module
2. Copy the virtual host of 443 port at the end
3. SSLCertificaateFile -> check certificate path and name. Also key path and name
4. Save
5. Restart httpd service
6. You can redirect the http to https in config file to mandatorily use https.
7. Enable Name Virtual host as we are having more than one virtual hosts.

**11/09/2018:**

1. Webserver is used for static content and app server for dynamic content.
2. Webserver is to understand http requests, app server is used to connect with business logic.
3. Increasing CPU and memory is scale up and down. increasing machines is scale out and in.

Install **app server tomcat:**

1. It needs java to install.
2. Server.xml -> there are 3 imp ports. Shutdown port, connector port, AJP port(LB).
3. 8443 port is used to create SSL secure connection b/n web server and app server.

Install an app inside webapps -> **yum install tomcat7-webapps**

1. Connection b/n web server and app server
2. For one to one connection we can use proxy pass.
3. Set up this in httpd.conf for to and fro requests.

ProxyPass /examples <http://172.31.45.203:8080/exmaples/>

ProxyPassReverse <http://172.31.45.203:8080/exmaples/> /examples

1. Add host in hosts file.
2. Stop and start webserver and app server and play around various requests -> what are all the issues. -> not found, 503 error, 404 error, not reachable….
3. **Server is too busy:**
4. Netstat -plan | grep 80 | grep ESTABLISHED
5. If this command gives too many time outs, there is an issue.
6. Is the issue with web server or app server or db server?
7. **Mod\_jk is the popular load balancer. It sits behind apache, so need to take the compatible version for this.**

**12/09/18:**

1. Mod\_jk is a open source used to connect with app servers.(Business Continuity Process)
2. Secure configuration -> keep application in DMZ ()
3. Prefork MPM, worker MPM.
4. SSL b/n webserver and app server for more secure applications -> port is 8443 on app server
5. Proxy pass uses http protocol where as mod\_jk uses AJP 1.3 protocol
6. Sticky session -> to avoid session time out, one request will go to same app server from the same ip
7. Failover-> if one server is down, request won’t go to that server
8. Domain model clustering ->

**How mod\_jk LB works?**

1. Will put all the entries of all app servers in workers.properties configuration file.
2. Deploying different modules of an application in different nodes is calld **Domain Clustering.** Eg., offline reports on node2 and UI layer on node1.
3. How to do deployment during production hours. Disable one node and deploy into that and do the same in node2. The other node will serve the requests.
4. Jk statistics will help us to which node we are connecting
5. Difference b/n proxy pass and mod\_jk???
6. Httpd -V -> to know whether worker mpi or prefork api is being used by our httpd.
7. Install compatable mod\_jk from google
8. Google->download mod\_jk for httpd 2.2.4 linux. You can copy link and get in linux machine.
9. Add this module in httpd.conf file to load it.
10. Check for mulesoft documentation for alias name.
11. Provide the workers.properties file path as well here.
12. Define the worker which is given in http.conf in worker.properties

**17/09/2018:**

1. Practice linux and go through questions and answers.???
2. Creation of machine, http.conf, apache, tomcat, ssl, redirection, tomcat, jdk, gcc, installation of apache, ports in tomcat, app deployment, proxy pass, mod\_jk, load balancer, external module downloads.
3. **Creation of two tomcats and one Apache and load balancing.**
4. Yum install -y tomcat-webapps
5. Httpd-conf -> LoadModule jk\_module modules/mod\_jk.so
6. Mod\_jk configuration documentation and copy it.
7. Set up worker.properties.
8. Take a new machine and install only tomcat
9. Install example application as well.
10. Configure worker.properties.= in etc/httpd/conf in first machine
11. Configure load balancer in above file.
12. Check the documentation.(mod\_jk load balancing)

**LoadModule jk\_module /etc/httpd/modules/mod\_jk.so**

**# Declare the module for use with the <IfModule directive> element. (This only applies to versions of HTTPD below 2.x. For 2.x and above, REMOVE THIS LINE.)**

**#AddModule mod\_jk.c**

**# Set path to workers.properties. We will create this file in the next step. The file will be placed in the same directory as httpd.conf.**

**JkWorkersFile /etc/httpd/conf/workers.properties**

**# Set path to jk shared memory. Generally, you'll want this to point to your local state or logs directory.**

**JkShmFile /etc/httpd/logs/mod\_jk.shm**

**# Where to put jk logs**

**# Set path to jk logs. This path should point to the same logs directory as the HTTPD access\_log.**

**JkLogFile /etc/httpd/logs/mod\_jk.log**

**# Set the jk log level. Valid values for this setting are 'debug', 'error', or 'info'.**

**JkLogLevel info**

**# Set timestamp log format. Use provided variables to customize.**

**JkLogStampFormat "[%a %b %d %H:%M:%S %Y]"**

**JkMount /examples/\* loadbalancer**

1. Etc/httpd/conf/workers.properties
2. Mount load balancder in httpd.conf(workers file???) instead of workers under modules section.
3. Create mod-jk.conf file and include this in httpd.conf file as below.

# Include mod\_jk's specific configuration file

Include conf/mod-jk.conf

1. Write the URL configuration in mod-jk.conf file as below.
2. Configure these IP and port numbers which are configured in workers.properties file in server.xml of tomcat to give connectivity.

<!-- A AJP 1.3 Connector on port 8009 -->

<Connector port="8009" address="18.218.9.174"

enableLookups="false" redirectPort="8443" debug="0"

maxThreads="150" minSpareThreads="25"

protocol="AJP/1.3"/>

<Engine name="standalone" defaultHost="localhost"

jvmRoute="host1">

1. Stop one tomcat and check.
2. Check with two apaches and two tomcats.
3. **LB for web layer…. -> it should be at AWS level**
4. AWS-> left side load balancer.
5. F5 LB is very popular.
6. It is just a device which distribute the load.
7. LTF -> Local Traffic Filter
8. F5 uses different types of algorithm, Round robin is most common. Also F5 keeps on check whether all the nodes are alive. If any one of the nodes is down, request won’t send to that.
9. We need to raise req for F5 ip address and configure that in our config files and raise requests for domains for all our webservers.

**18/09/2018:**

**Monitoring tools – Nagios:**

1. Google -> Nagios monitoring tool.
2. Nagios is the best in infrastructure monitoring tools.
3. Need to install core engine and then plugins to it(around 2000+ plugins).
4. Nagios, HP site scope, ICINGA, Dynamo, INSTANA, neurilik(application performance monitoring tools) are the most popular monitoring tool.
5. Install Nagios binary.
6. Start the Nagios service
7. Login by hitting IP/Nagios. User name and password are nagiosadmin/nagios
8. Commands.config-> which need to be monitored, C drive of a machine etc.
9. Contacts.cfg-> to whom mail need to send, grouping the people on various levels
10. Timeperiods.cfg-> in which time period alerts should send…. Mon to Fri etc
11. Templates.cfg-> template is having list of rules of all the above, we can reuse in different machines. We can write our own templates as well.
12. Install Nagios pulgins.

**19/09/2018:**

1. Remove the Nagios and other dependent stuff if already installed.
2. Nagios prerequisites should be installed first.
3. Go to Nagios site for source. <https://www.nagios.org/downloads/Nagios-core/>
4. Now download Nagios plugins.
5. Extract both
6. Add Nagios user as we shouldn’t give root user privileges, change mod for this
7. Add apache user, change mod.
8. Cat /etc/passwd
9. Compile the Nagios binary with Nagios group name.
10. Build the Nagios source code using make all command
11. Install Nagios with make install command
12. As per the instructions given, the Nagios front ending s/w’s should also be installed.
13. Copy local libraries to Nagios installed location.
14. Provide the executable permissions recursively to Nagios files.
15. Start Nagios service
16. Start apache service
17. Create http password and copy to Nagios file. -> for changing the password
18. Build and install the Nagios plugins as well.
19. To check Nagios config file at the time of system start run the commands.
20. Configure email addresses.
21. Go to objects and update contact.cfg file
22. This user should be a member of one of the groups.
23. Enable warning state monitoring.
24. Check the logs and mail box for alerts.
25. Edit localhost.cfg file
26. Check the command for plugin with which it configured.
27. Check that plugin in plugins folder.
28. Use find command to check the command -#find . -name check\_http
29. Use help command to check the documentation.
30. The warning might be because no index.html in home directory as it is showing 403 forbidden as warning description.
31. Create the file and check.
32. Check for the log location in main config file and go to that location and check logs.
33. Chk whether an email has been sent or not in the logs.
34. Check and debug for all warnings and errors to know the reasons and try to fix them???
35. Give permission to other users on the folder on which access denied error.
36. We need a messaging s/w SMTP to send the email. Eg., postmail, emailx. Configure this SMTP with Nagios.
37. Go to hosts file, define a host and add this host.
38. Configure the host with IP address of the machine to which you want to monitor.
39. Check logs why mail has not sent.
40. Add our machine in services to be monitored.
41. If we want to configure customized monitoring, eg., monitor tomcat7 add that plugin and configure.
42. Google -> Nagios plugin exchange -> see the reviews and installation steps-> then install plugin

**20/09/18:**

1. How many live connections are there should be displayed on Nagios.??? Use netstat command, write a new command in command.cnf file and add it in service by calling with commands.
2. On which port Nagios will run? **5666**
3. Active check->if Nagios check directly and Nagios is responsible for monitoring it is call active check. If some other s/w is checking and providing the readings to Nagios and Nagios is just displaying on dashboard, it is called passive check.
4. What is diff b/n NRPE and SNMP??? NRPE is available for all operating systems by default but SNMP is not.
5. What is Flap Detection???-> if CPU usage is increased more than threshold and by the time Nagios check happens it reduces and again increases, Nagios can’t send alert. To check this situation Nagios used Flap detection mechanism. Nagios checks last 21 times whether there are too many hikes.
6. Distributed Monitoring(**Nagios Fusion**) -> distributed monitoring system. Will keep more number of Nagios slaves in various locations to monitor machines at that place and give the report to Nagios Master and it will display the dashboard and send the alerts.
7. We can set up a small cron job in another machine to monitor the Nagios machine.
8. Why DevOps -> bringing dev and operations together. Cross skilling of dev team and ops teams. DevOps providing the configuration tools to set a new machine or reset a damages machine easily in a less time.

**Chef:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Config tool** | **Developed on** | **Agent based/not** | **Popularity** | **Mechanism** |
| **Salt stack** | **Python** | **Agentless** | **Deprecated** | **Push** |
| **Puppet** | **Ruby** | **Agent Based** | **Deprecated** | **Pull** |
| **Chef** | **Ruby, Erlang** | **AB** | **Stable** | **Pull** |
| **Ansible** | **Python, Yaml** | **AL** | **Emerging** | **Push** |

**What is automation:**

**Chef-Framework:**

1. Login to hosted chef server
2. Create organization
3. Download the kit
4. Setup EC2
5. Upload the kit
6. Install chef DK using wget and rpm commands.
7. Set up local chef repo

Manage.chef.io -> official chef site.

Account details:

laxman\_devops

laxman123

1. Create chef account -- https://www.chef.io/
2. Download the kit
3. Upload this into our linux machine using winscp. Use SCP protocol
4. Login into linux machine
5. Download chef DK for linux version (#cat /etc/\*release\*)
6. Go to home directory and wget the chef DK link.
7. You will see chefdk and chef-starter in home directory
8. Install using the command #rpm -Uvh chefdk-3.3.30-1.e17.x86\_64.rpm
9. RPM doesn’t need internet to install b/c all the dependencies are bundled in it.
10. Ls -l
11. #unzip chef-starter.zip
12. #cd chef-repo
13. #ls -la
14. #Cd ./chef
15. #Ls -l
16. #cd cookbooks/
17. #cd..
18. Check whether we have connectivity with chef server. #knife ssl check

* Set up node/EC2
* Register this node with chef server(Bootstrap)

1. Install chef-client
2. Establish SSH to connect node from work station
3. Execute knife bootstrap <IP> <UID> <\*.pem>

* Now this machine is the member of our organization.
* Chef-client connectivity

1. Establish SSH connectivity from work station to client using authorization keys.
2. Run a command to set up required things in client.
3. Go to client machine root directory.
4. Goto chef client
5. Write a new cook book in work station.
6. From chef client upload this into server
7. Pull this into our chef client.

**04/10/2018:**

1. Zero mode/local mode/testing mode -> is used to install in local chef server for command testing purpose.
2. If you just wanted to see how it is working but not to install, use **noop** mode.
3. Create proxy pass inside chef recipe (default.rb):

Template ‘/etc/httpd/conf.d/proxy.conf’ do

Source ‘proxy.conf.erb’

End

1. Run the chef command to create templates folder.
2. Give proxypass and proxypassReverse here.

**Learn most imp resources and actions from chef help document.**

We can get ready made cookbooks in supermarket.chef.io/

**Write a cookbook which install apache, tomcat and the proxy lines should be inserted in existing httpd.conf. Also, instead of hard coding the ip address here, collect it from server.**

**Clue -> docs.chef.io/ohai.html -> how to extract IP address.**

**Create file, insert two lines. If I change the 2nd line, it should be appending at the end and if we edit the 1st line, it should replace it.**

**05/10/2018:**

Attributes: To know the status of node before/after/during the installation of chef client.

Ohai : which runs on node before chef client installation and is useful for getting status of nodes.

It collects pulse of the system as a node object and store in chef server.

**https://docs.chef.io/attributes.html**

**Exercise:**

**Complete the document for MySQL, Apache, PHP set up…..**

**Create attribute -> default attribute with name lamp-stack**

**Recipe -> install apache and get the attributes from above step.**

**08/10/2018:**

**Puppet:**

1. $yum install puppet3-server in server
2. $yum install puppet3 in client
3. $vi /etc/puppet/puppet.conf -> In client add the server host name as shown under ssl-dir. (host name can be identified using $hostname -f)

server = <>

1. Start puppet master. #service puppetmaster start (or)
2. $/etc/init.d/puppetmaster start -> in server
3. $service puppet start -> start client in client machine
4. $puppet cert list –all -> in server to check which client are waiting for sign (+) to connect with server
5. $puppet cert sign <DNS of the client which is waiting for sign>
6. $puppet cert list - -all -> you should see “+” for the client now.
7. $puppet agent - -test - - noop -> just for test. Don’t install but test it and provide the errors if any.
8. $puppet agent - -test -> from client to check any catalogue to be executed. Nothing as of now.
9. $cd /etc/puppet/manifests/
10. $vi site.pp -> in server

package { ‘httpd’:

Ensure => present,

}

service { ‘httpd’ :

ensure => running,

enable => true,

require => Package[“httpd”],

}

1. #puppet agent -t -> from client to execute the above
2. #cd /var/lib/puppet/ssl -> in server
3. #ll
4. #ls certificate\_requests
5. You can see which clients are in queue.
6. If you start client before server, you will face issues(client will be in hang state), then you can check this path.
7. You can also clean the hang status client.
8. #cd /var/lib/puppet/ -> under client go here and remove this ssl folder completely.
9. In server don’t do this, because there will be so many clients already connected.
10. Google -> puppet forge -> apache

**09/10/2018:**

1. 3 ways to create.
2. Write all the details in site.pp and use it.
3. Write different modules for each requirement and point them in site.pp.
4. Download readily available modules and refer them in site.pp.

In site.pp, we can import different pp files writing below line.

**import “lamp.pp”**

Second way:

1. Go to modules folder.
2. Create a folder with same name as class.(create lamp folder under it create manifests folder)
3. Create manifests folder under it.
4. Inside this folder go to manifests and create init.pp
5. Write the resources here. But include the same code under **class <class name same as folder naem> {}.**
6. Call this in site.pp

**include <class name>**

Third way:

1. Open puppet forge website.
2. Search for required module.
3. Go to /etc/puppet/modules
4. #puppet module install <module name>
5. Check the module names of downloaded ones.
6. Edit site.pp with these details.

**Include <module name>**

1. Create the php file

Puppet cert clean –all

**10/10/2018:**

**Ansible:**

1. Developed on python, most of the machines having python so no need to add extra dependency of python while installation of ansible.
2. Push mechanism, so no need of agent installation.
3. How to do ssh from one linux machine to other from root user?

In the target machine edit below line in vi /etc/ssh/sshd\_config file.

# vi /etc/ssh/sshd\_config

#PermitRootLogin yes -> this has to be uncommented.

#service sshd restart

1. Create the ssh connection from controller to client from root to root.
2. In server install ansible

#pip install ansible

#mkdir -p /etc/ansible

#vi /etc/ansible/hosts

[servers]

<IP addresses of client – u will get by using #ifconfig comand>

.

.

1. We can run any linux command in clients from the controller

#ansible -m command -a ‘df -h’ Servers

1. Write smaple play book in yml file with commands.

Name -> is like comment.

Action -> syntax

1. Check yaml syntax in [www.yamllint.com](http://www.yamllint.com)
2. Can also do from linux browser.

#elinks <http://www.yamllint.com>

1. To check httpd status in client from controller….

#ansible Servers -m **service** -a “name=httpd state=started”

1. Another two plugins generally use from controller

#ansible Servers -m **command** -a “/sbin/reboot -t now”

#ansible Servers -m **ping**

1. Ansible -I invent
2. Plugin to copy files from controllers to clients.

#ansible Servers -m copy -a “src=/etc/ansible/apache.yml dest=/tmp”

1. How to apply only for single client. Just replace Servers with that IP address.

#ansible <IP Address> copy -a “src=/etc/ansible/apache.yml dest=/tmp”

1. To remove any service from client use “state=absent”

#ansible Servers -m yum -a “name=tomcat7 state=absent”

**11/10/2018:**

1. Set up a web server and deploy index.html – Requirement.
2. Ansible# vi index.html 🡪 create index.html inside ansible folder.

welcome

1. Ansible# vi http.yml 🡪 create http.yml inside ansible folder.

---

- host: Servers

become: yes

tasks:

- name: Ensure Apache installed

yum: name=httpd state=present

- name: Create a directory

file: path=/var/www/html state=directory

- name: Deploy index

copy: src=index.html dest=/var/www/html/ mode=0644

- name: apache is running

service: name=httpd enable=yes state=started

- name: Flush all temporary rules

service: name=iptables state=restarted

- name: allow port 80/http access from anywhere

iptables:

action: insert

chain: INPUT

protocol: tcp

destination\_port: 80

state: present

source: 0.0.0.0/0

jump: ACCEPT

:wq!

1. Check the syntax of yml and correct in case any syntax mistake.(remove any passwords if present before checking)
2. Ansible# ansible-playbook http.yml 🡪 push the play book to clients.
3. Getting issues with IPTABLES -> remove it.

* Create one more play book.

1. Ansible# vi db.yml

---

- hosts: Servers

become: yes

vars:

dbname: devops

dbuser: mint

dbpass: 12344

tasks:

- name: mysql install

yum: name=mysql-server state=present

-name: mysql running

yum: name=mysqld state=started

-name: ensure mysql-python is also installed

yum: name=MySQL-python state=present

- name: Create DB

mysql\_db: name={{dbname}} state=present

- name: Create username

mysql\_user: name={{dbuser}} password={{dbpass}} priv='\*.\*:ALL' state present

:wq!

1. Ansible# ansible-playbook db.yml  push the play book to clients.
2. We can dynamically provide passwords by using prompt module.

vars\_prompt:

- name: dbpass

prompt: "enter password for DB"

tasks:

- debug: msg="DB password is {{dbpass}}"

1. You can use notify module as below.

- name: Copy the website config file

copy: src=httpd.conf dest=/etc/httpd/conf/httpd.conf

notify:

- Apache Restart

Bottom of the yml file --

handlers:

- name: Apache Restart

service: name=httpd state=present

1. Installation based on OS and their sub-versions:

tasks:

- name: Install Apache on Centos

apt: name=apache2 state=present

when: ansible\_os\_family == "Ubuntu"

- name: Install Apache on Centos

yum: name=httpd state=present

when: ansible\_os\_family == "RedHat"

* become: yes is used to become sudo(root) user.
* become\_user:<> to become some other user.

**15/10/2018:**

**Google-> wordpress ansible role.**

1. Install git and initialize using the below commands.

#cd /etc/ansible/

#yum install -y git

#git init

#git clone <sample url>

#mkdir roles -> unser ansible folder

#wordpresee…. init apache

Ansible vault is used to save passwords and encrypted data.

#ansible-vault create vault

**16/10/2018:**

**Berks in chef:**

1. Berks is useful to upload and download the dependencies of our cookbooks.
2. We can write the dependencies in meta.rb file.
3. We can upload our cookbook along with the dependency using #berks upload command.
4. In real time we use berks instead of knife.

**Roles creation:**

1. Rule is used to group few cookbooks together. Role is a group of cookbooks.
2. Instead of saving individual cookbooks in the run list, we can save role directly.
3. In chef.io -> policy -> Create.

**Managing users in chef.io:**

1. Chef.io -> Administration -> Add/Remove user, Group users, restrict user to run only few cook books.
2. We can also add users in groups. Add/remove read/wirte/execute permissions for users/groups.

**Data Bags:**

1. Chef.io -> Policy -> Data Bags : used for passwords.
2. Chef.io -> policy -> Clients -> delete clients.

**Devops Chef anotomy:**

1. Google -> Devops chef anotomy -> https://docs.chef.io/chef\_client\_overview.html
2. In work station, under etc->chef hosts -> knife.rb contains the client details.
3. In client, the server details are present in client.rb???

**22/10/2018:**

**Build Tools:**

* **s/w** contains multiple files -> **configure**(pom.xml) -> **make/ant/maven** are the **build tools** which will read the configuration files and prepare -> **package**.

**Make:**

* It is platform dependent.

**Ant:**

* Platform independent.

**Maven:**

* It always produces one artifact.
* Jar/war/ear is the artifact.

1. Set up new machine
2. Install java8 using wget under opt folder.
3. Unzip it using #tar -xzvf <>
4. Set up java soft link path to java 8 using a command.
5. Set up JAVA\_HOME using export command.
6. Google -> download maven for linux. Wget it for download and extract it.
7. Create softlink for this maven.
8. Set up M2\_HOME path using export command.

**GIT – GIT HUB:**

* GIT clone will copy files from remote repository to both local repository and local machine as well.
* GIT fetch will update the files only in local repository but not local hard disk.
* GIT works over distributed environment.
* GIT is a client version and GIT-HUB is a server side version.

#yum install git

#mkdir practice

#git init -> to create a local repository inside practice folder

#touch october18.txt

#git status -> to check the status of our created files

#git branch -> to check all the branches of that trunk

#git add . -> to add all our files to staging area

#git status

#touch october18\_1.txt

#git rm - -cached october18.txt -> remove the file from staging

#git status

#git commit . -> to commit all files in local to local repository.

#git status

#git ls-files -> to check local repository status

#git config - -

#git commit - - -> to edit the last commit.

#rm october18.txt

#git ls-files

#git status

#git checkout

#git branch temp -> to create a branch with name temp

#git checkout temp -> to move to temp branch to do all operations on temp branch.

#git branch -> to know in which branch you are.

**GIT\_HUB:**

* Two types of repositories, public and private.
* Creation of account is free and public repo is also free.
* Anyone can download from public repo.
* Google -> Github.com -> create account -> login -> create new repository
* User name: vlkumarm

#practice# git remote add origin <repo link>

# git push -u origin master

#git pull -u origin master

* It pulls data from origin to local repo and also to hard disk.

# git push -u origin master

* We can create our organization and invite members for it to handle by all team members.
* Vimdiff is a linux tool similar to Beyond compare.
* Create two branches, create same file in two branches and do changes differently and commit from those branches. Try to merge one to other to remove one.
* You will get conflicts and then you need to manually merge it.

**25/10/2018:**

**Rebasing vs Merge:**

**GIT Stash:**

**Merge Conflicts:**

**GIT Cherry Pick -> pulling and pushing files from one branch to other.**

**JENKINS:**

* JENKINS developed on java.
* Google -> install Jenkins for <OS> -> take the link -> wget <url>
* https://wiki.jenkins.io/display/JENKINS/Installing+Jenkins+on+Red+Hat+distributions
* Rpm key for Jenkins
* Install Jenkins
* Jenkins need java 8, so download it
* Google -> java 8 for <OS>
* #cd /opt
* Wget - - no-cookies……
* Set up paths and all for java
* Jenkins work on port number 8080, so check whether this port is available or not.
* #netstat -plan | grep 8080
* #service jenkins start
* Go to browser and check with <IP Address>:8080 -> it asks you for key.
* #publinc ip:8080 -> it gives the location where the key is present.
* # cat /var/lib/Jenkins/secrets/initialAdminPassword
* Provide this key and install the suggested plugins.
* In case if you forgot password you can go to above mentioned path.
* Once the plugins installation completed, create first admin user with sample details.
* Left side -> **manage Jenkins** -> **Configure** **Global Security ->** allow users to sign up.
* **People ->** contains people details
* **Mange Jenkins -> global tool configuration -> give path for jdk also check automatic installation for maven and its version and save.**

Jenkins credentials:

Uname: Laxman

Pwd: laxman123

**Manage Jenkins:**

* Go to mange Jenkins, you can see Jenkins home directory. You can always change this. Just click on question mark, it will say where you can change it.
* **# of executions ->** how many builds can be done in parallel
* **Extended email service -**> once deployment is done, it sends a mail. here we need to provide SMTP name.
* **Configure Global Security** -> explore it
* **Global Tool Configuration** -> Add JDK here with oracle account, instead of we downloaded it.
* or else you can provide JAVA\_HOME which already install. You can add more than one JDK.
* if you want to install some tools like git, we can provide “yum install git” commands. You can add various softwares, you can simply click add and provide the version. That will automatically install.
* **You can provide proxy settings also.**

**Create a new item:**

* **Provide the name and select free style project.**
* **Source Code Management -> provide the GIT/SVN and provide the url.(eg., google->game of life git hub)**
* **Build Triggers ->** set up this
* **Buid ->**  select maven and provide clean install as target

**Need to build in one Jenkins machine and copy to other linux machine with credentials and SSH connection.**

**Take back up before copying.**

**How to run the deploy commands after copying into target machine?**

**29/10/2018:**

**Docker:**

* Physical machines -> VM ware(on one physical heavy server 10’s of VMs can be prepared) -> Containers(Docker)
* Docker will sit in b/n our application and OS kernel.
* You no need to consider about the OS.
* Install docker on linux machine
* #docker images -> all the software images will be listed(java, apache, linux……)
* #docker version
* Docker Swarm is useful in moving containers to Docker

**30/10/2018:**

* Docker gives us the identical environments for development, test and production developments so that, you don’t end up with environment issues.
* Container is a machine but image is a s/w. you can login into container but not into image.

#yum install docker

#service docker start

#docker version

#docker images -> you will see what are all the images already there

# docker pull hello-world -> pulls the “hello-world” image.

# docker images

#docker run hello-world

# docker ps -> will provide the containers which are running

# docker ps -a -> will provide the containers which are running along with stopped ones.

# docker rm b84a3e2f6966 -> remove this container but not image.

#docker run hello-world

# docker ps -> will provide the containers which are running

# docker ps -a -> will provide the containers which are running along with stopped ones.

#docker rmi hello-world -> will remove complete image as well.

#docker run ubui

#ctrl+p+q -> you will go to your actual machine

#docker -i -t

#docker attach < docker\_id > -> you will go inside the docker. You can work inside that container.

#ifconfig -> will give the container’s ip

#exit -> you will come out of container.

#docker ps -> will give short id of all containers

#docker inspect <docker\_id> -> complete details of docker container along with its long id. it will also gives us the location where it is stored.

#docker logs <docker\_id>

#docker run ubuntu:14.04 echo ”run completed”

#docker run ubuntu:14.04 ps ax

#docker run -i -t ubuntu:14.04 -> now you are in ubuntu machine

# apt-get update -> ubuntu repository will get update.

# cat /etc/\*release\* -> to know the linux release version.

#exit -> you will come out of that container machine. All your work will be gone. So we should not exit from here, instead we can switch to EC2 machine.

#ctrl+p+q -> you will switch to EC2 machine and the container will be running.

#docker attach <container\_id> -> again u will connect to same container.

#docker run -d centos:7 ping 127.0.0.1 -c 20 -> it downloads the centos and run it and ping the given ip for 20 times and then stop.

#docker logs <container\_id>

# docker run -d -P tomcat:7 -> Pulls tomcat7 install it, run it and assign the EC2 machine port to this tomcat7 image. So that from out side world we can access this tomcat with EC2 machine IP with generated port number.(you can see the port number using # docker ps)

Eg., IP:port

* Google->Docket hub -> create login -> check for images
* By default the IP address of all docker containers will start with **172.17.**

User name: vlkumarmygapula

laxman123

**31/10/2018:**

**Creation of Images in Docker:**

* Every image is again a group of images.
* Docker commit vs Dockerfile
* All the images are read only with a writable container at the top.
* If you do any changes on a container those will be applicable on the top writable container and you need to explicitly commit it so that a new image will be created with base image and your changes.
* You can also user Dockerfile to do this in a script.
* # docker commit <container\_id> <user\_name>/<container\_name>:<version>

**Dockerfile:**

* Create a new directory and go inside it.
* # vi Dockerfile
* Write **FROM, RUN, CMD** commands inside this.
* # docker build -t laxman:1.0 .
* **ENTRYPOINT ->** we can provide the argument at run time for a command which will run at the time of starting the container.
* **EXPOSE** -> used to configure on which ports container should listen.

**# docker exec -it bb36bdd0e70b /bin/bash -> exec will be used to start an app of a container when there is on OS on container.**

**01/11/2018:**

**Uploading the created images:**

# docker run -it centos:7

# yum install -y url

# docker images

# docker ps -a

# docker commit <container\_id> centos/curl:1.0

# docker tag <image\_id> <userNameOfDockerHub>/imageName

#docker tag <> raghuopsdev/curl:1.0

# docker images

# docker login

# docker push raghuopsdev/curl:1.0

# docker push <userNameOfDockerHub>/<image\_name>

* To remove the images which are having multiple other referencecs…

# docker images | grep f216cfb59484 | awk '{print $1 ":" $2}' | xargs docker rmi

Browser -> docker hub -> create repository for dev and production separately -> upload to that repository.

**Volumes:**

* The data inside on container can be mounted on another, so that it pulls the data. Mounting is actually providing the connectivity.
* Create a folder and file in an EC2 machine and mount this with a container. Create one more file inside the container and exit and check EC2 machine.
* Create a container with option “-v” with source and destination with /bin/bash option.
* You will get into that container and the destination folder will be created.
* Now create a file inside that folder. It will be created in source machine volume as well.
* In case if you delete the container and create a new one also, it will be created with those files. You can create an image with commit option.
* If you do this in external volume by mounting to the machine, you don’t have fear even the EC2 or VM fails to start. You can use that volume to create a new container.

# docker run -it -v /ec2-user/volumeTest:/volumeDestination tomcat:2.4 /bin/bash

**Container networking basics:**

**Automapping ports:**

# docker run -d -P nginx:1.7

# docker run -d -p 8080:80 nginx:1.7

**Linking Containers:**

* Linking is a mechanism to establish communication b/n two containers securely.
* Recipient container will be having access to the data from source container.
* Connection will be created using container names.
* Create source container with a name
* Now create recipient container using - - link option by giving source container name.

# docker run -d - - name sourcecontainer postgres

# docker run -d - - name recipientcontainer - - link sourcecontainer:db nginx

docker inspect -f "{{ .HostConfig.Links }}" web

* How to copy data from source to recipient after providing connection???

**02/11/2018:**

**Using Docker in CI:**

* Go to home directory ec2-user, create folder for new project.
* Git clone <url>

|  |  |
| --- | --- |
| **Traditional** | **Docker** |
| GIT clone <> | Mvn image from ref |
| Mvn install, JDK, set up env, set mvn path | Git clone <> |
| Mvn clean install | Write a docker file to instruct to use mvn image to build the code. This file also contains build and set up of tomcat, env vars. |
|  |  |
|  |  |

**05/11/2018:**

1. Build maven images including one OS(ubuntu). Build the image. This image contains only OS and maven.
2. Install GIT using yum install.
3. Get sample java code.
4. Git clone <game of life link>
5. Go to docRoot where pom.xml is there
6. Build ther gameof life artifacts using MVN image
7. Docker run(refer the document)
8. Build tomcat image with gameoflife.war file
9. Dockerfile – Through which you will build Tomcat image

Will discuss on non-java project in docker???

**Orchestration tools:**

1. Docker Swarm
2. Amazon ECS(Elastic Container Service)
3. Kubernetes -> google -> containers

**13/11/2018:**

* # docker logs -f <container\_name>

**Kubernetes:**

* Useful for orchestration