Ansible PlayBook document.

1. What is Ansible Playbook.

2. Oraganizing tasks in Ansible Palybook.

3. Introduction to YAML.

4. Writing a play book.

Configure your Ansible account

ansible all -m ping (on master)

5. Working with some adhoc commands.

ansible all -s -m shell -a 'uptime'

ansible all -s -m shell -a 'date'

ansible all -s -m shell -a 'cat /etc/redhat-release'

more /etc/ansible/hosts

ansible all -s -m shell -a 'df -h'

ansible all -s -m shell -a 'mount'

If you want to mount all systems you can mount easily using below command

ansible all -s -m shell -a 'mount -a'

ansible all -s -m shell -a 'service sshd status'

ansible all -s -m shell -a 'service httpd status'

These are acting very fast than individual systems.

So everthing is working fine in our ansible setup now. Again we want to check test user to root user once on all nodes uing below command

sudo su -

again we are going to check few adhoc commands

ansible all -s -m shell -a 'yum installed|grep telnet'

ansible all -s -m shell -a 'yum installed|grep python'

ansible all -s -m shell -a 'yum installed|grep ftp'

ansible appserver -s -m shell -a 'yum installed|grep ftp'

ansible apacheweb -s -m shell -a 'yum installed|grep ftp'

ansible all -s -m shell -a 'yum installed|grep telnet' (on Master)

am going to comnnect system which ever in green colour and am going to remove telenet

ssh test@<green colour node ip>

sudo yum remove telnet --- going to remove telnet on this node now

logout from current node –-- (that means you are back to master now again)

ansible all -s -m shell -a 'yum installed|grep telnet' (on Master)

Now I can see the failed state on that acting node is without telenet since we have connected that node and removed telnet.

6. Ansible command Line . (writing a plybook)

we tested so many commands now in ansible

7. system facts.

ansible all -m ping

cat /etc/ansible/hosts

what is meant by system fact ?

I want to get system information of all the nodes presen here means in master.

We have already learnt called setup module right using we can get it

ansible all -m setup|more – instead of all you can mention system name like local

this is the fact we are going to discuss here.

Go to tmp folder on master

I can see nothing has been created here now.

ansible all -m setup - -tree /tmp/system\_facts.txt

this file is going to grab all the systems info into here

8. How do we write Playbook.

First playbook

we have to create folder called Playbooks

cd – going to home dir

mkdir Playbooks

cd Playbooks

now am going to install vi editor first on all nodes usind adhoc command later I will show with playbook how we can do this.

ansible all -s -m yum -a 'pkg=vim state=installed update\_cache=true'

ansible all -s -m shell -a 'yum list installed|grep vim' ---- to check all nodes for vim

Then first playbook

vim myfirstplaybook.yml

---

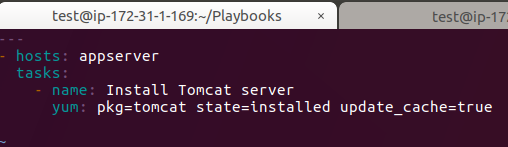
- hosts : apacheweb

tasks :

- name: Install the tomcat server

yum: pkg=tomcat state=installed update\_cache=true

:wq!



Explanation about Yml file syntax :

===========================

The file starts with:

---

This is a requirement for YAML to interpret the file as a proper document. YAML allows multiple "documents" to exist in one file, each separated by ---, but Ansible only wants one per file, so this should only be present at the top of the file.

YAML is very sensitive to white-space, and uses that to group different pieces of information together. You should use only spaces and not tabs and you must use consistent spacing for your file to be read correctly. Items at the same level of indentation are considered sibling elements.

Items that begin with a - are considered list items. Items that have the format of key: value operate as hashes or dictionaries. That's pretty much all there is to basic YAML.

YAML documents basically define a hierarchical tree structure with the containing elements further to the left.

On the second line, we have this:

---

- hosts: all

This is a list item in YAML as we learned above, but since it is at the left-most level, it is also an Ansible "play". Plays are basically groups of tasks that are performed on a certain set of hosts to allow them to fulfill the function you want to assign to them. Each play must specify a host or group of hosts, as we do here.

Next, we have a set of tasks:

---

- hosts: all

tasks:

- name: Installs tomcat web server

yum: pkg=tomcat state=installed update\_cache=true

At the top level, we have "tasks:" at the same level as "hosts:". This contains a list (because it starts with a "-") which contains key-value pairs.

The first one, "name", is more of a description than a name. You can call this whatever you would like.

The next key is "yum". This is a reference to an Ansible module, just like when we use the ansible command and type something like:

ansible -m yum -a 'whatever' all

This module allows us to specify a package and the state that it should be in, which is "installed" in our case. The update-cache=true part tells our remote machine to update its package cache (apt-get update) prior to installing the software.

The "notify" item contains a list with one item, which is called "start nginx". This is not an internal Ansible command, it is a reference to a handler, which can perform certain functions when it is called from within a task. We will define the "start nginx" handler below.

---

- hosts: all

tasks:

- name: Installs tomcat web server

yum: pkg=tomcat state=installed update\_cache=true

The "handlers" section exists at the same level as the "hosts" and "tasks". Handlers are just like tasks, but they only run when they have been told by a task that changes have occurred on the client system.

For instance, we have a handler here that starts the Nginx service after the package is installed. The handler is not called unless the "Installs nginx web server" task results in changes to the system, meaning that the package had to be installed and wasn't already there.

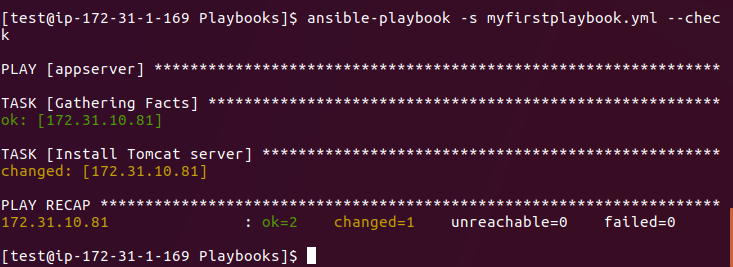
We can save this playbook into a file called something like "myfirstplaybbok.yml".

Before executing playbook we can actually check the syntax errors of any playbbok using below command.

ansible-playbook myfirstplaybook.yml - - check

for installations and all we need sudo access so

ansible-playbook -s myfirstplaybook.yml - - check



Here you can check syntax of the palybook file and if the servers are in yellow colour tomcat is not installed so this playbook file is going to install here. Before if you want to check on mentioned node.

Go to connected node on other window

and with test user

sudo yum list installed|grep -i tomat

here you can check tomcat is installed or not .

back to master node

ansible-playbook myfirstplaybook.yml

This command is going to install tomcat on all nodes

for installations and all we need sudo access so

ansible-playbook -s myfirstplaybook.yml

Till here – hosts I have mentioned apacheweb now if you want you just mention all as hosts.

* hosts: all

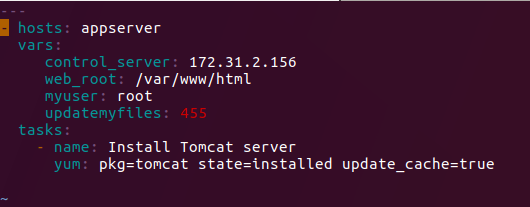
9. Writing a playbook using Variables :

Variables: Inclusion types

First am going to work with my firstplaybook program and before that am gong to copy my play book with a new name called variable with below command.

cp -pr myfirstplaybook.yml myfirstplaybook\_variable.yml

vim myfirstplaybook\_variable.yml

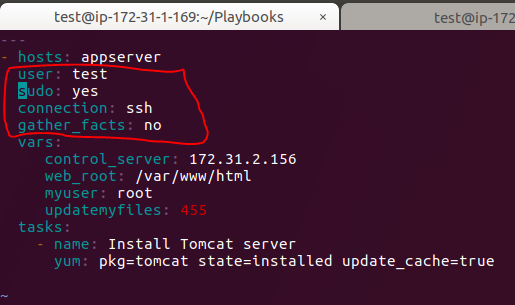


9. Writing a playbook with Target section :

If you want to define some targets for this playbaook you can menton in it. Targets are something like this playbook can be used any environment.

cp –pr myfirstplaybook\_variable.yml myfirstplaybook\_target.yml

vim myfirstplaybook\_target.yml

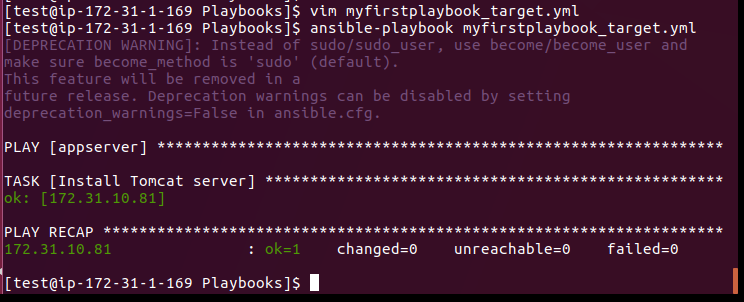


So here we are going to define targets as user only test user can run and sudo is yes and connection should be ssh and gather\_fatcs is no.

So when ever you want to run this playbook in any environment these targets should be meet.

So just try to execute this command in master node

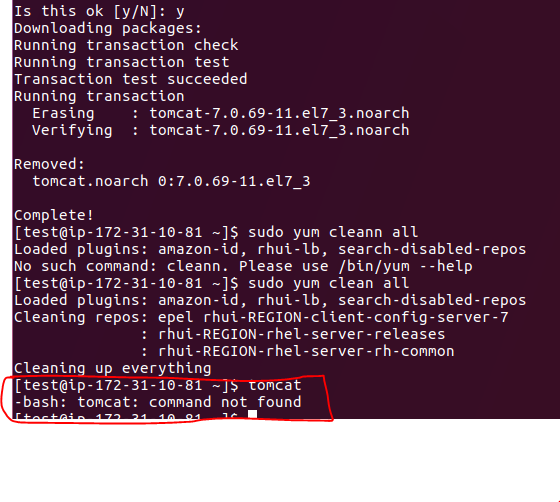
ansible-playbook myfirstplaybook\_target.yml (master node)



If it is installed already details in green colour. If not gong to install and details in red colour. So now am going to remove tomcat in appserver node using below commands

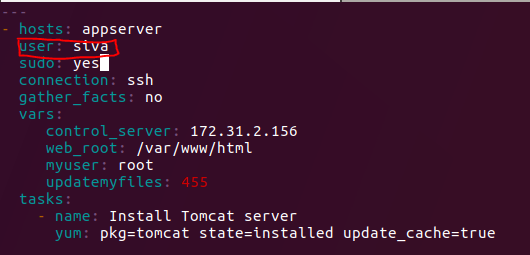
sudo yum remove tomcat ---- on appserver node (in our example node3)

sudo yum clean all



Coming to Master node again want to run with some other user like “siva” not with test user for that we have to edit yml file.

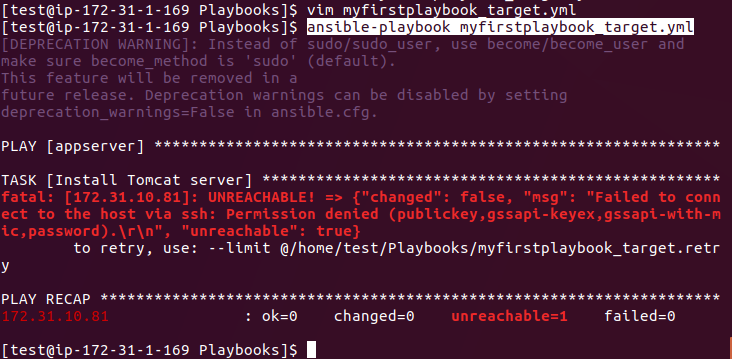
vim myfirstplaybook\_target.yml



Now I want to run using below command

ansible-playbook myfirstplaybook\_target.yml

It will give an below error.

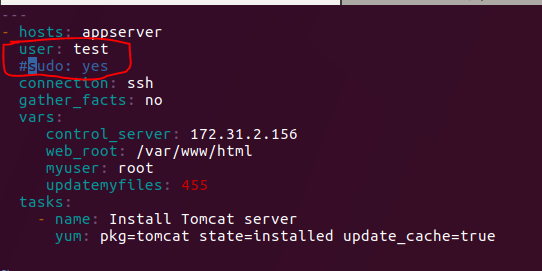


Why because we have mentioned target as siva user but you are going to run with user name called test. So it s giving an error.

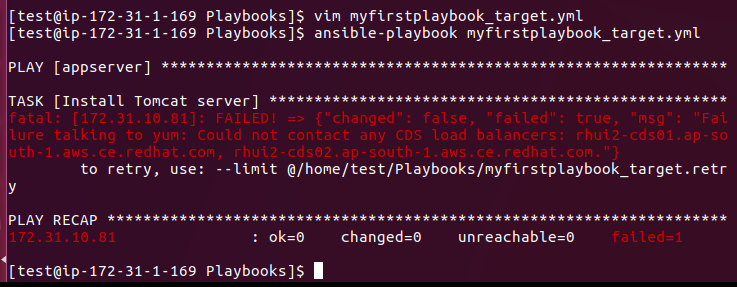
This time go and check on appserver node tomcat cant be installed using **tomcat** command

Now I want to run with test user but with out sudo accesss like below

vim myfirstplaybook\_target.yml

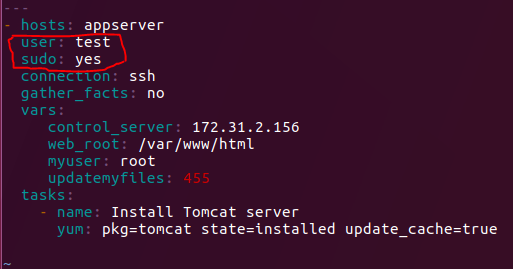
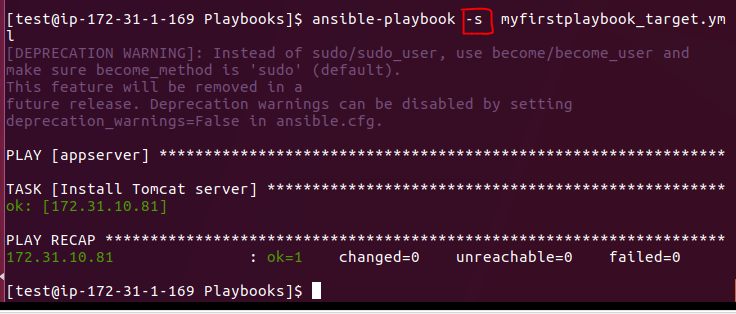


ansible-playbook myfirstplaybook\_target.yml



Again we are going to get same error why because we are going to run test but without sudo access. For this we can do 2 ways.

1. we have to update yum file with sudo: yes
2. ansible-playbook –s myfirstplaybook\_target.yml

If you want to check go to appserver node and check with **tomcat** command.

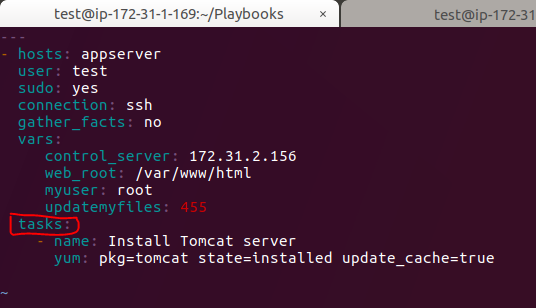
**So this is what all about target section under hosts line.**

10. Writing a Play book with Task section :

So now again am going to create new playbook for this section as well same like previous one by copiyng existing playbook.

cp -pr myfirstplaybook\_target.yml myfirstplaybook\_task.yml

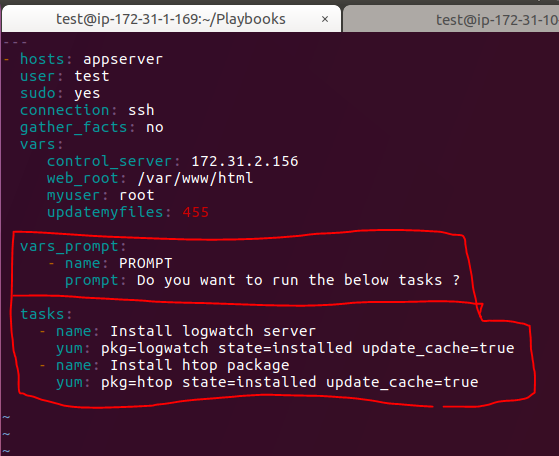
**vim** myfirstplaybook\_task.yml



We had already written task section from the beginning of our first playbook. And in tasks section we can declare number tasks.

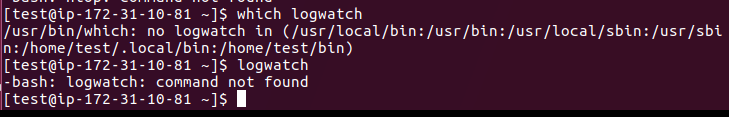
Now one more variable we have to discuss that is **vars\_promt.** This variable is going to prompt the user for the inputs. This prompt we have to write like below

**vim** myfirstplaybook\_task.yml

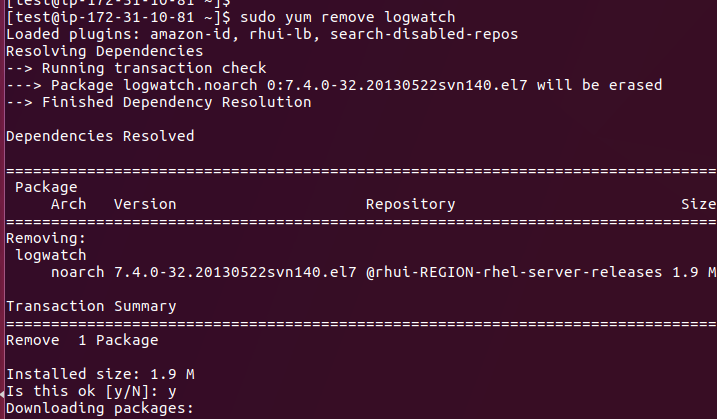


First check on appserver node wether packages are installed or not using command

which logwatch

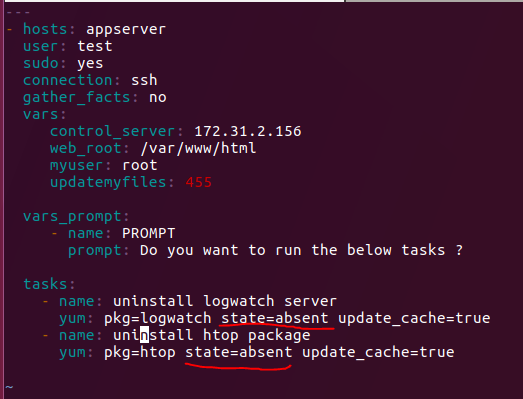


You remove logwatch using **sudo yum remove logwatch** on indivdual nodes.



This can be uninstalled on this particullar node.

And till now we have done installation part using playbooks now we can uninstallations with playbooks. Using keyword **absent**



That’s it for task section.

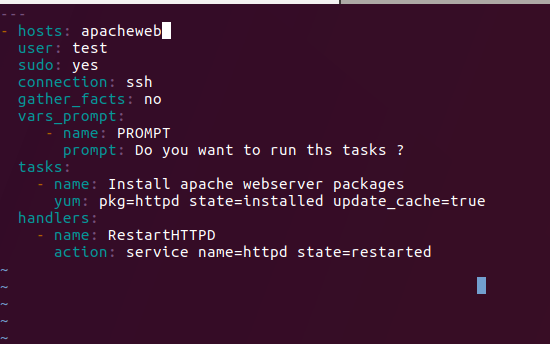
**11. Writing a Playbook with Handler section :**

Now am going to copy existing playbook for handler section as well

cp -pr myfirstplaybook\_task.yml myfirstplaybook\_handler.yml

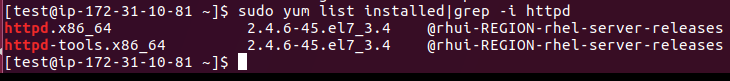
vim myfirstplaybook\_handler.yml

I just removed all variables and iam going to install httpd server packages and finally it is going to restart the services on apacheweb group servers.



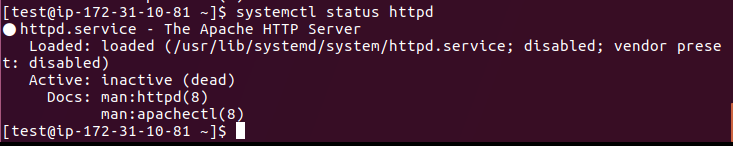
You just go to individual nodes and check with below command

sudo yum list installed | grep –i httpd

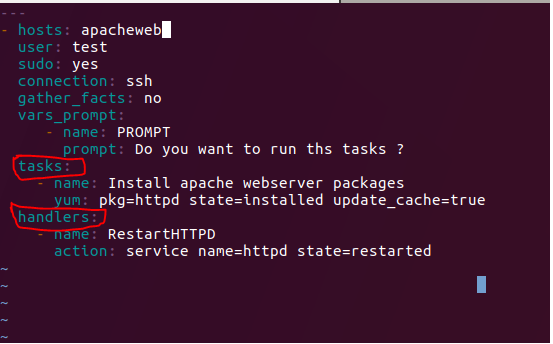


So httpd is installed now and now we can check servics restarted or not using below command

systemctl status httpd



Now services are inactive not started. If you just check .yml file now

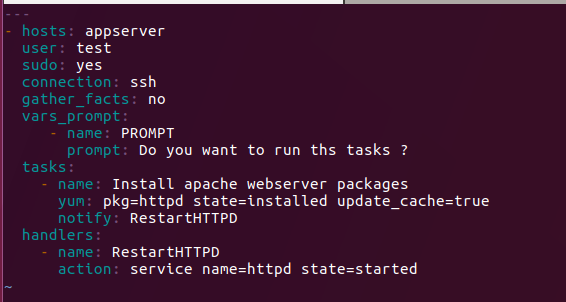


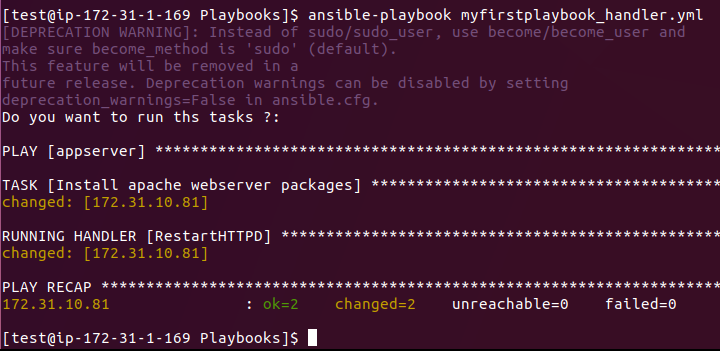
These 2 tasks and Handlers are not inter linked that’s why it s not working as expected.

So after task one more module we have to add that s called notifiy. Notify value is same like handler name. see the below screen shot.

So once the task is completed it will nofity the handler to do the next task.

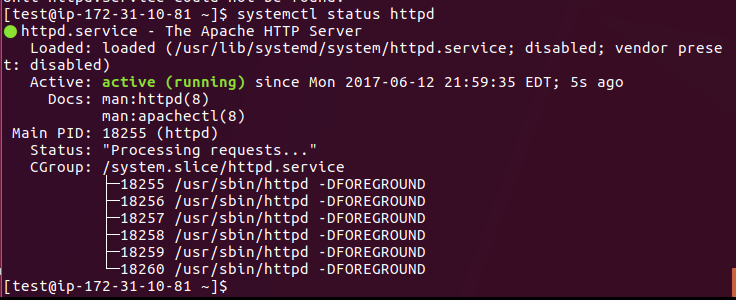
ansible-playbook myfirstplaybook\_handler.yml



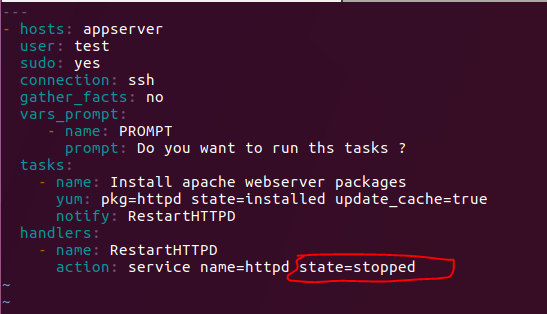


Goto apacheweb nodes and check the services started or not using below command:

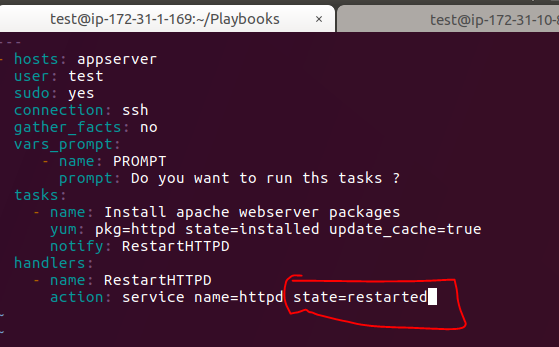
**Systemctl status httpd**



If I don’t want services now so we can stop the services now usng “stopped”



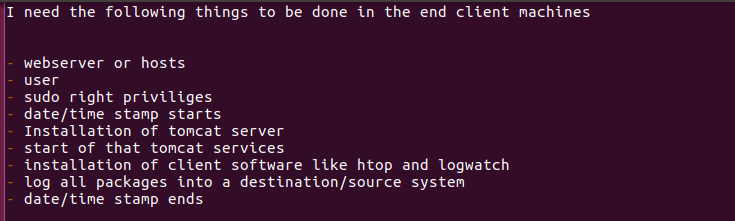
If you want to restart:



**12. Design and outlining your playbook :**

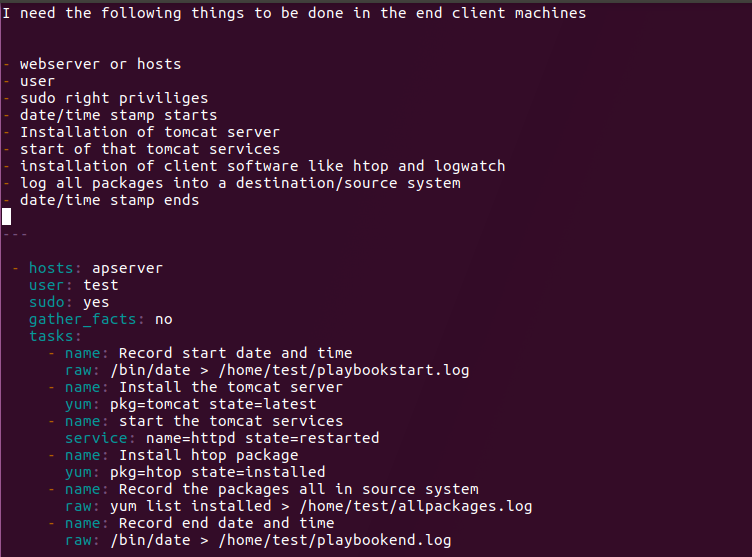
Now we are just going to create our outline of our requirement first in the form test file then we are going to convert text file to playbook file.

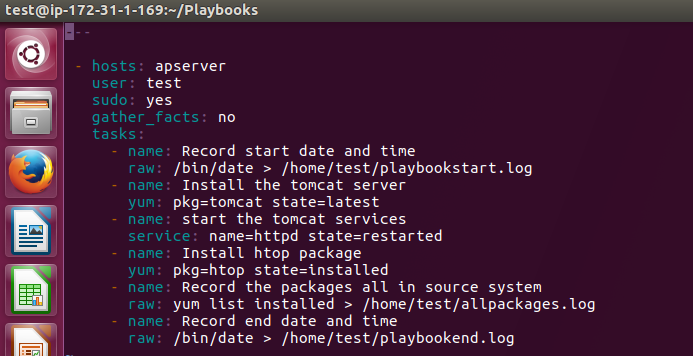
vim otline.txt



cp –pr outlne.txt myfirstplaybook\_outline.yml

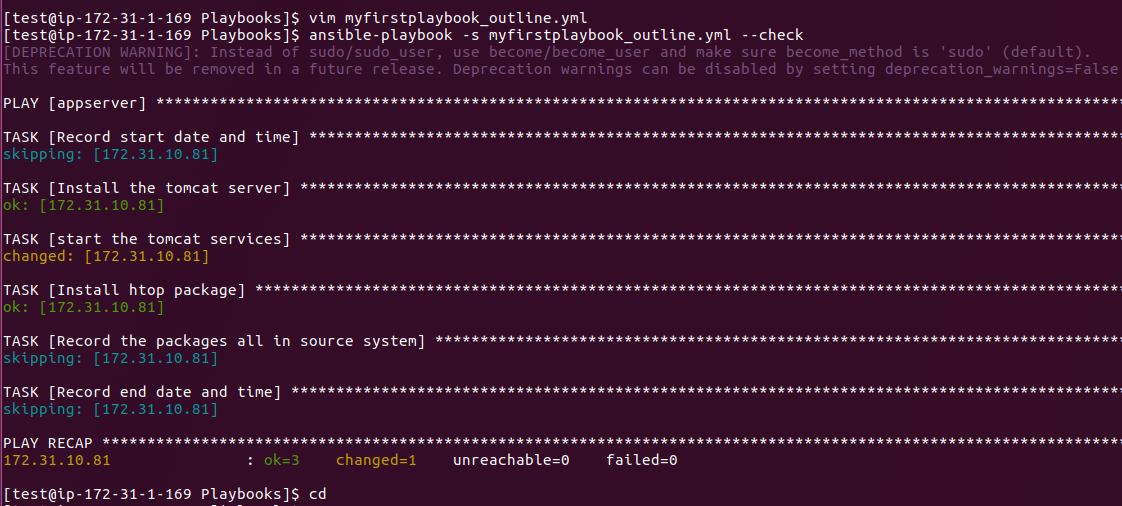
vim myfirstplaybook\_outline.yml





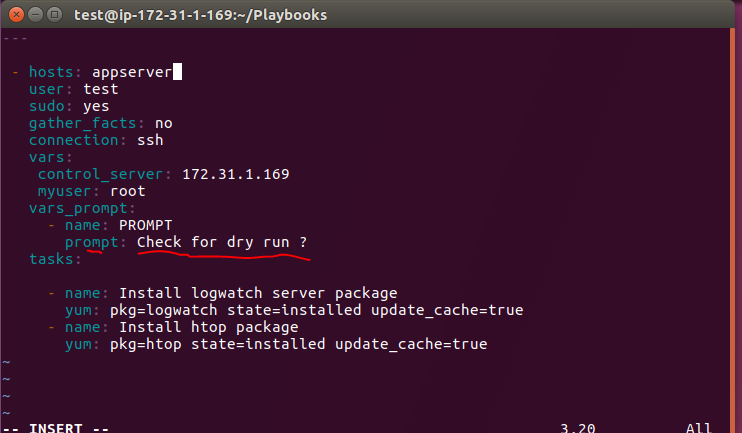
Then we have to run the yml file using below command

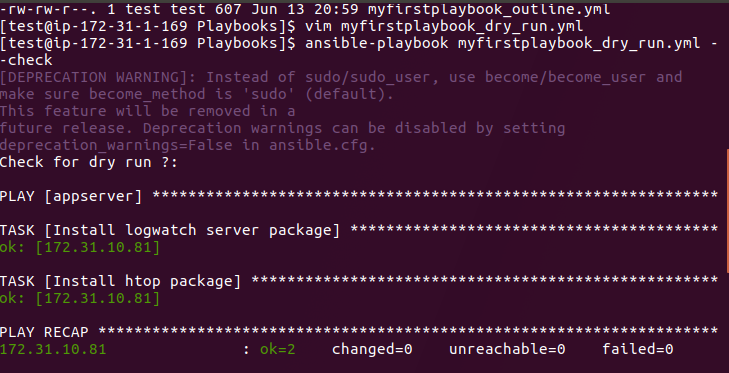
Ansible-playbook myfirstplaybook\_outlook.yml –check



13. Writing a playbook with a Dry Run concept

cp –pr myfirstplaybook\_tasks myfirstplaybook\_Dry\_Run.yml





Here we are getting always some error when you are executing ansible commands saying Deprecation warnings. If you want to remove do the follwing steps

Sudo su –

Cd /etc/ansible

Vi ansible.cfg

Remove # sysmbol infront Deprecation Warnings False lable.

So Dry run means just it will check wether I can do or not. If you –check that is a dry run if you are not using –check for ansible command that is a real run.

ansible-playbook myfirstplaybook\_dry\_run.yml –check --🡪 Dry Run

ansible-playbook myfirstplaybook\_dry\_run. --🡪 Real Run.

Ansible is so fast to install or remove packages than ndividual systems

State=installed --- means it is going to install package

State=absent --- means it is going to uninstal package.

14. Writing a playbook with Asynchronous Polling :

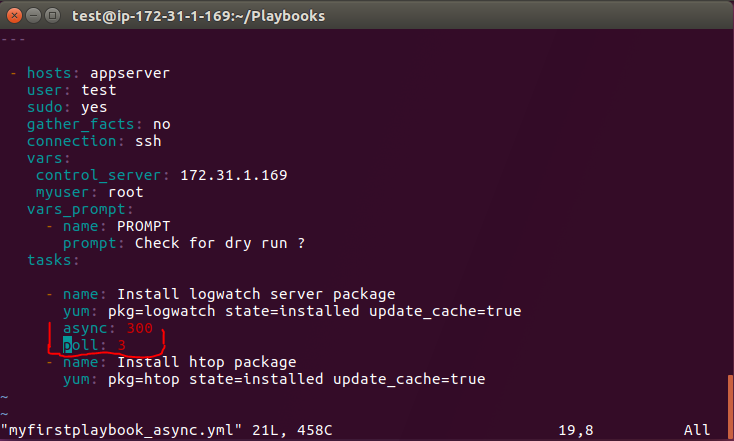
By default tasks in playbooks block, meaning the connections stay open until the task is done on each node. This may not always desirable, or you may be running operations that take longer than the ssh timeout.

The easiest way to do this is to kick them off all at once and then poll untill they are done.

To launch a task asynchronously, specify its maximum runtime and how frequently you would like to poll for status. The deafult poll value is 10 second if you do not specify a value for a poll.

Cp –pr myfirstplaybook\_dry\_run.yml myfirstplaybook\_async.yml

Vim myfirstplaybook\_async.yml



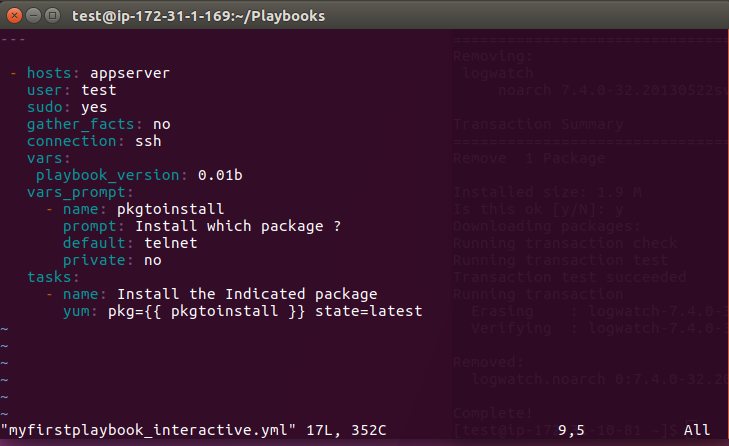
Ansible-playbook myfirstplaybook\_async.yml –vv

15. Writing a playbook with Interactive mode

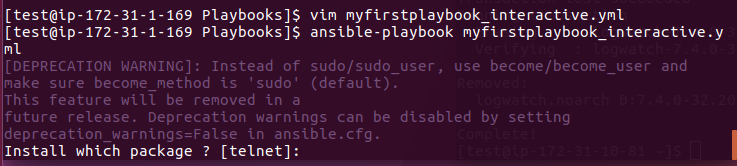
Now that we have seen a playbook, let’s step back and develop a more complex one and explaining it a section at a time, Now how to use the variables we have defined.

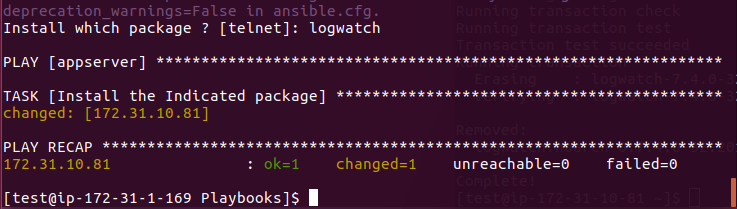
cp –pr myfirstplaybook\_async.yml myfirstplaybook\_interactive.yml

vim myfirstplaybook\_interactive.yml



If you want to uninstall given package just mention state as absent.



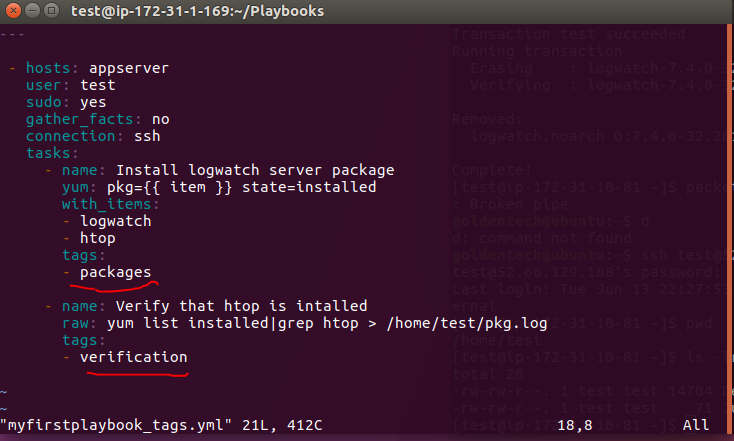


This is how we are going to interactive with our ansible playbooks.

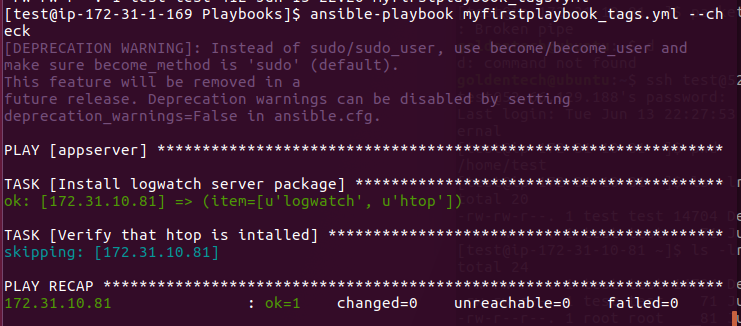
15. Writing a playbook with Tags:

As we did during our modules section, we will demonstrate playbook functions, one at a tim, using examples to illustrate how they work in Ansible.

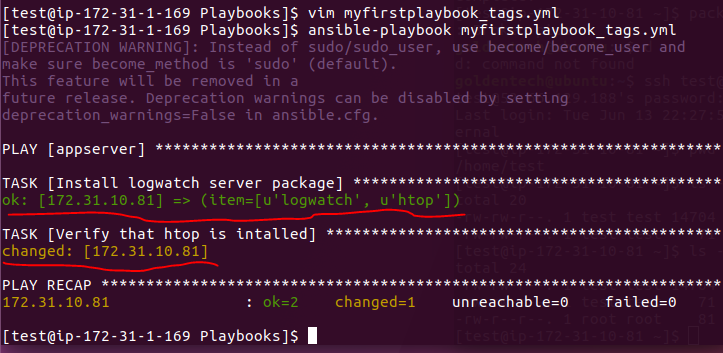
cp -pr myfirstplaybook\_tasks.yml myfirstplaybook\_tags.yml



ansible-playbook myfirstplaybook\_tags.yml - -check

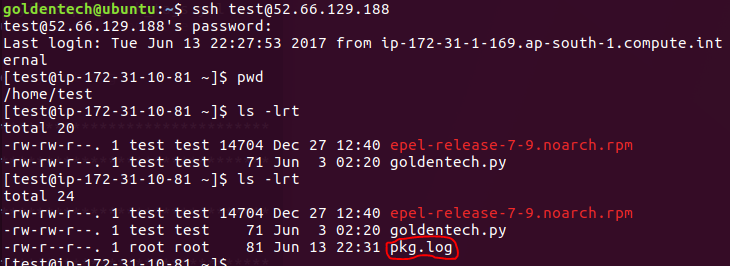


ansible-playbook myfirstplaybook\_tags.yml



Now I can on destination nodes logwatch and htop are installed and second step is verified also.

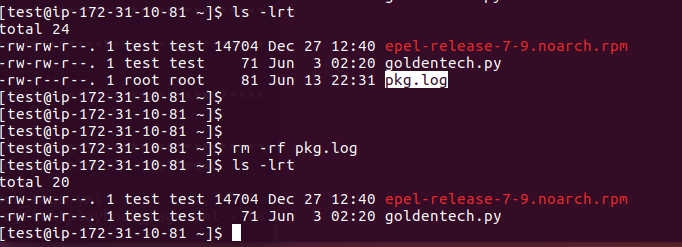
Now Iam going to verify on destination node



So that means my 2 tags are working fine in my play book one is packages and second one is verification.

Now I want to skip one tag and I want to run only one tag in my playbook. Just run the below command.

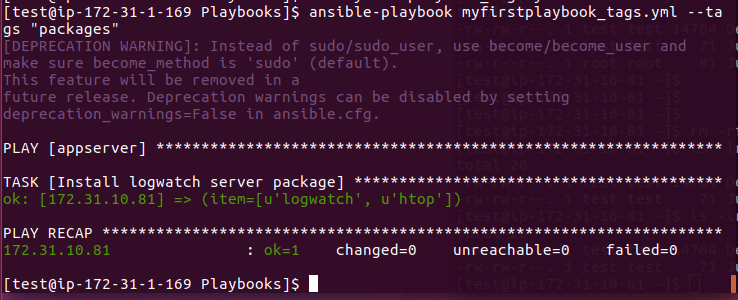
Now I am going to remove pkg.log file from destination node like below



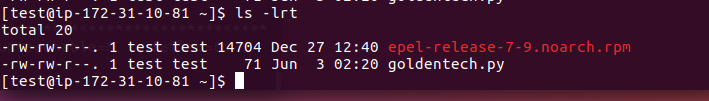
Now am going to run only job not 2 jobs

ansible-playbook myfirstplaybook\_tags.yml - -tags “packages”

ansible-playbook myfirstplaybook\_tags.yml --skip-tags “verification”



No Verfication tag was ran and no file got created on destination nodes.



16. Writing a Playbook with RunOnce module :

cp -pr myfirstplaybook\_tags.yml myfirstplaybook\_runonce.yml

vim myfirstplaybook\_runonce.yml

This taskonly run once on one host only if you mention in any task.

