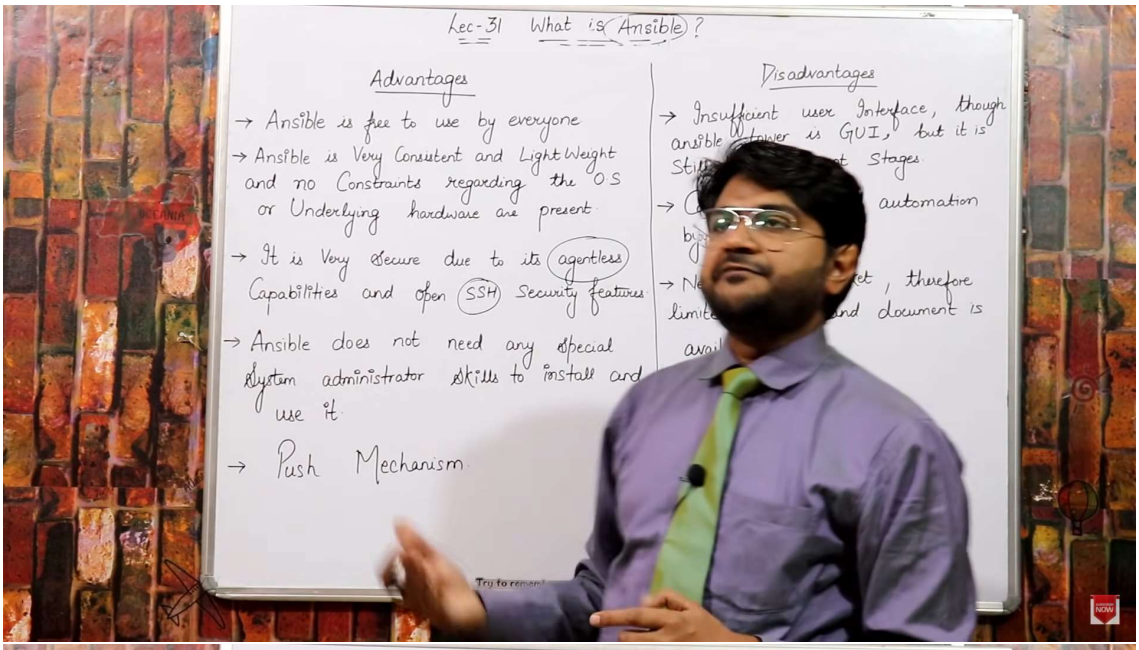
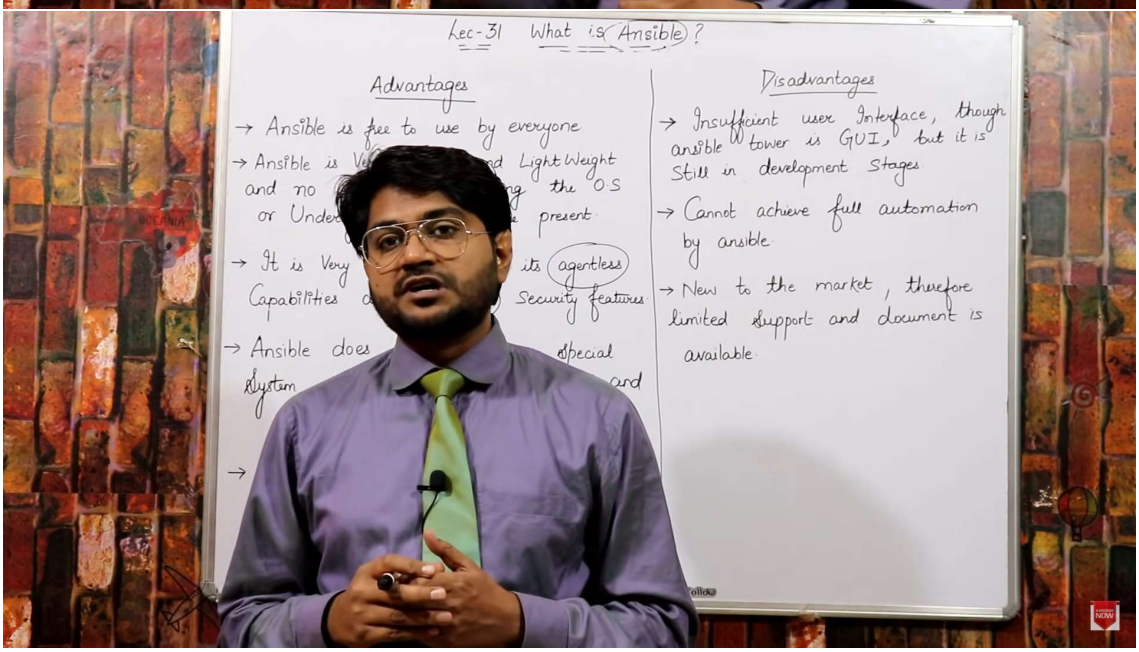
Ansible: Push configuration management tool, open source (infrastructure as code)

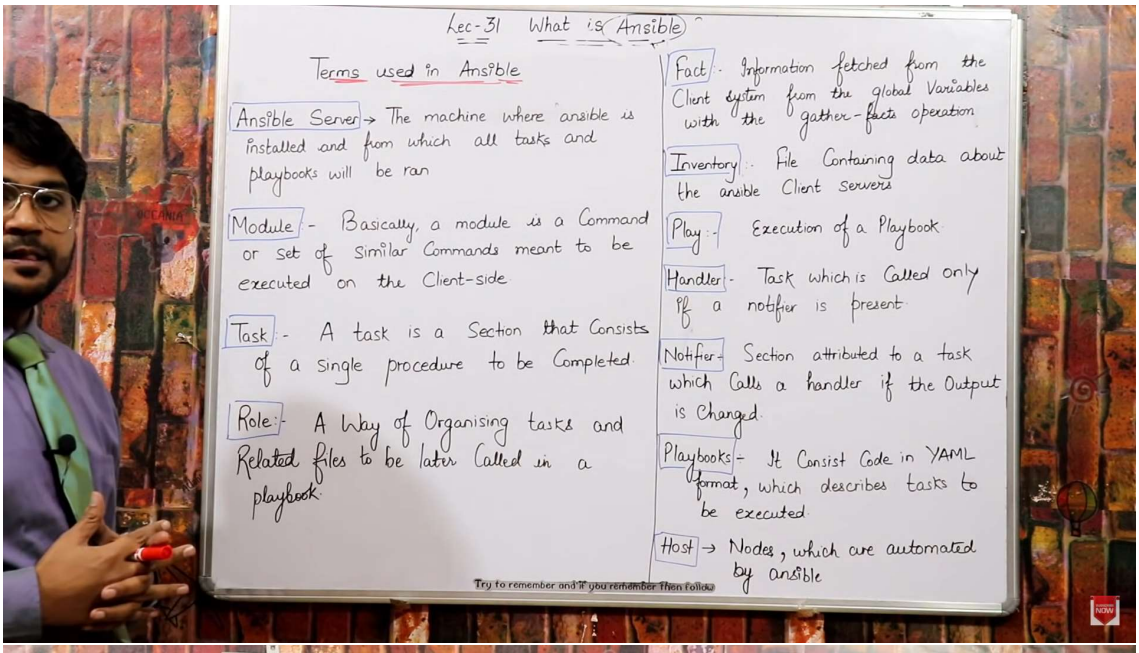
- Michael Dehaan developed ansible in Feb 2012. Redhat acquired it in 2015.
- Ansible is available for redhat, centos, oracle Linux, Debian and also on windows.
- Can use this tool whether the servers are on premises or on cloud.
- It turns your code into infrastructure i.e your computing environment has some of the same attribute as your application.

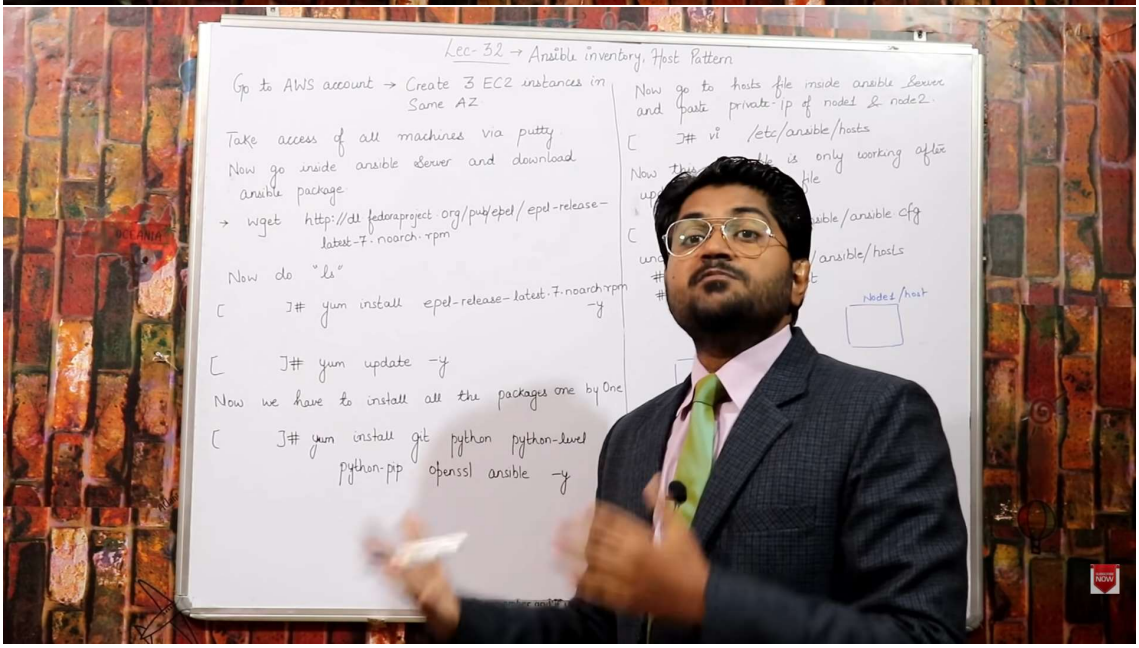
  Ansible is agent less
- Ansible server communicate with nodes using ssh.
- **Playbook**: where we write our code.
- Ansible uses **YAML** (yet another markup language) scripting language.





Ansible inventory, Host pattern (epel :extra packages for enterprise linux)



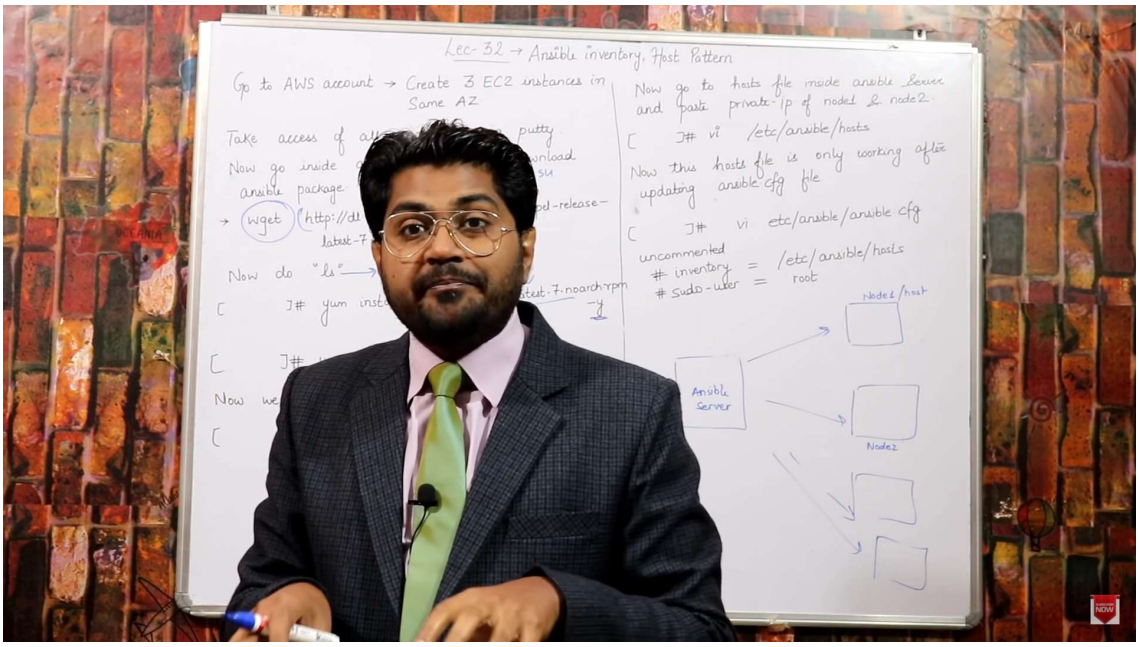


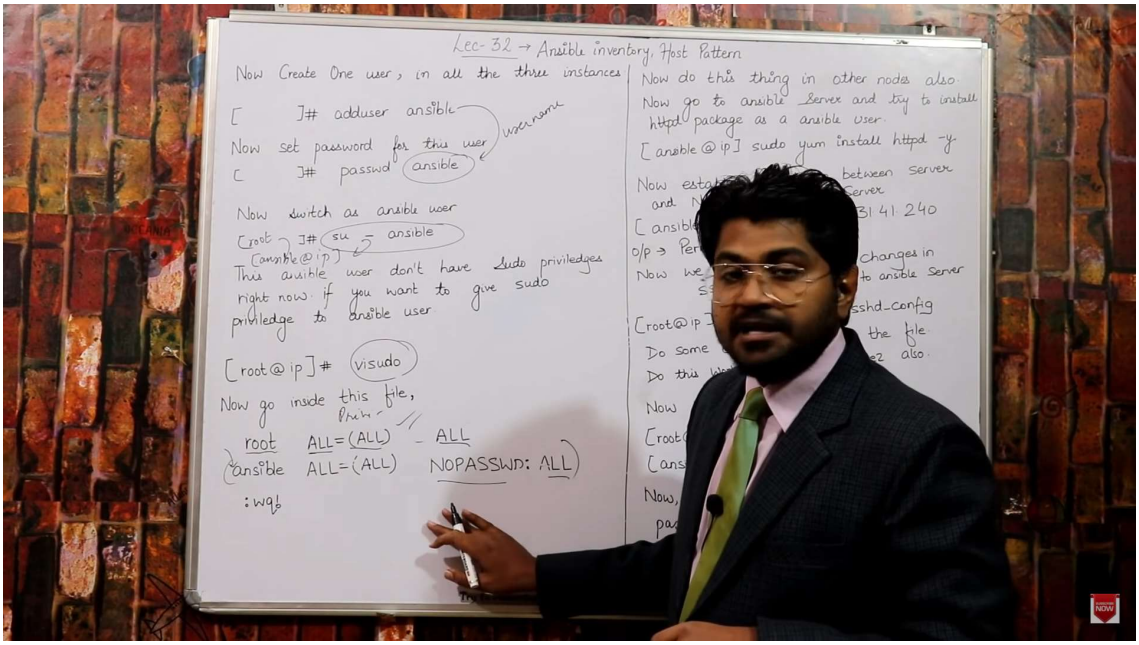
- **<u>Lab steps:</u>**
  Step1: create 3 machines in same availability zone.
  Step2: Make on machine as Ansible server by installing ansible server package.
  Cmd: **wget httpd://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm**
  Step3: ls
  Step4: yum install epel-release-latest-7.noarch.rpm -y
  Step5: yum update -y
  Step6: install the packages one by one from epel package
  Cmd: **yum install git python python-level python pip openssl ansible -y**

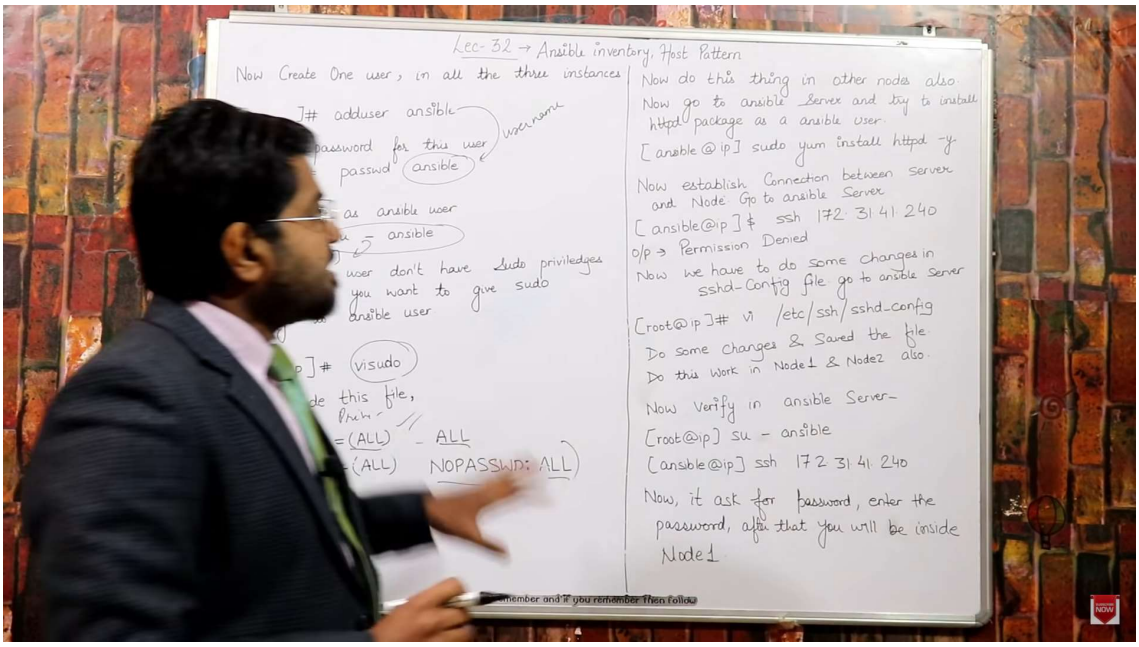
- **To check the nodes connected with ansible server.**

Ansible server has a default fine named as host where we update the private Ip of the the nodes so that server knows which nodes are connected.
 # vi /etc/ansible/hosts
Create a group and Update the private ips of 2 machines.

- **To activate the hosts file**
  Go to ansible configuration file **#vi /etc/ansible/ansible.cfg and uncomment the inventory line plus sudo-user =root**





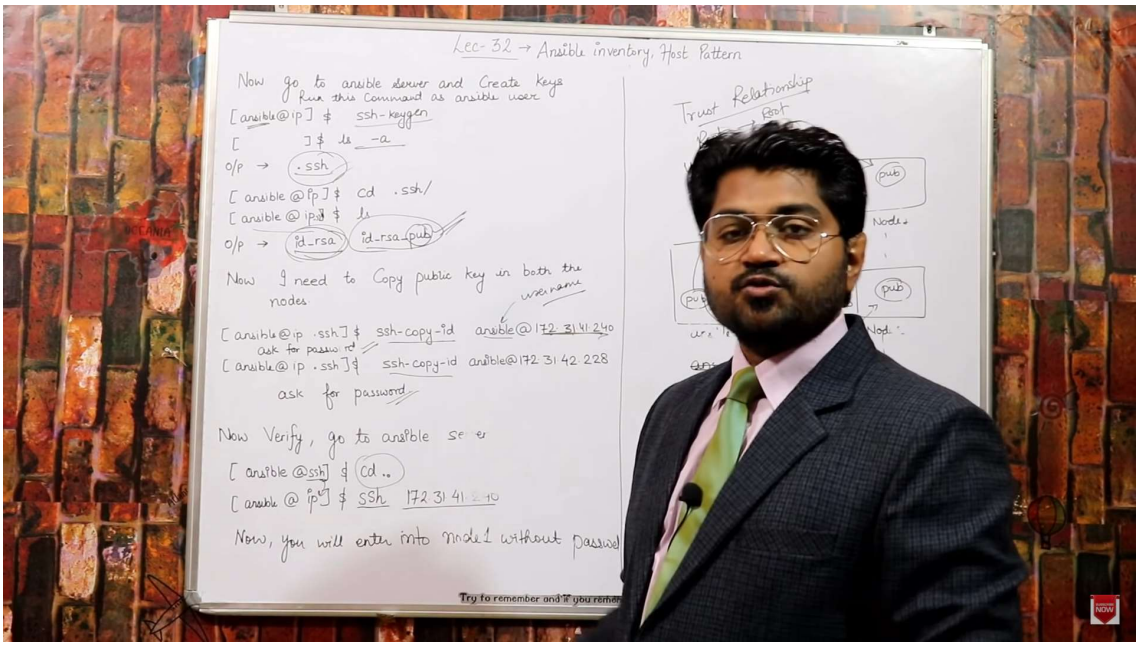


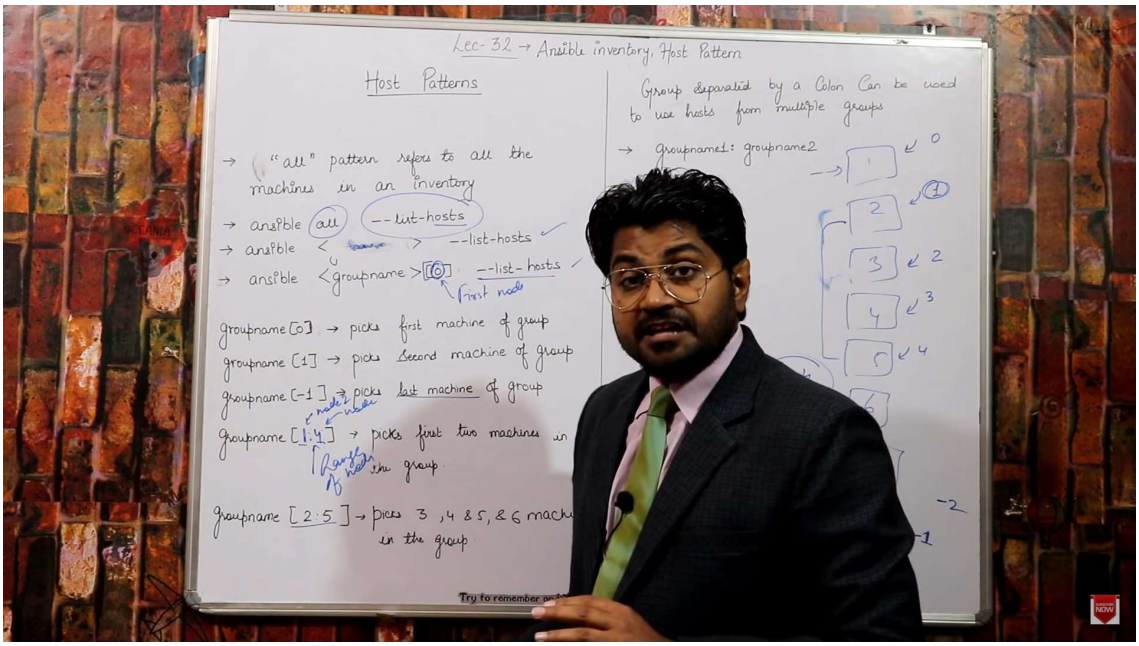
Go to vi /etc/ssh/sshd_confg
Uncomment PermitRootLogin yes and PasswordAuthentication Yes
Comment PasswordAuthentication no
After these steps restart the service so that these changes are implemented.

- First machine pr hum asnisble server package install krte hai jiske wajah se server pr 2 file create ho jaate h **1.ansible.cfg 2. Host**
- Host file me hum ek group banate hai jisme nodes ke private ip mention krte hai
- Uske baad hum ansible user create krte hai saare machines pr and then usko sudo priviliges prvide kre **visudo** me jaakr. Now ansible user can install the packages.
- Nodes ke beech connectivity krne k lea server ke hum sshd_config file me kuch changes krte hai.

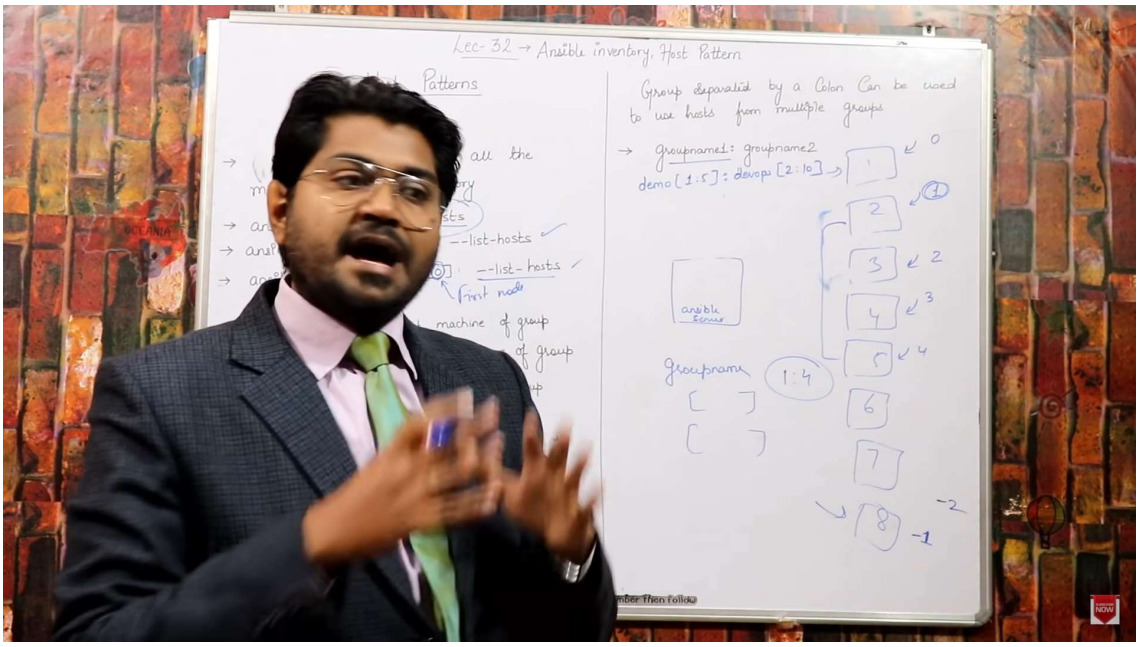
Password less login on Node (Trust relationship)

It is created on **equal level** for example **root with root and user with user**.
To create trust relationship b/w devices they should be on same level either they should login into root on all machines or into a user on all machines.
With creating trust relationship **login to ansible user**.





First node is 0, last node is -1
groupname[2:5] shows node 3,4,5,6 of group
demo[1:3]:devops[2:4] will show node 2,3,4 of group demo and node 3,4,5 of group devops

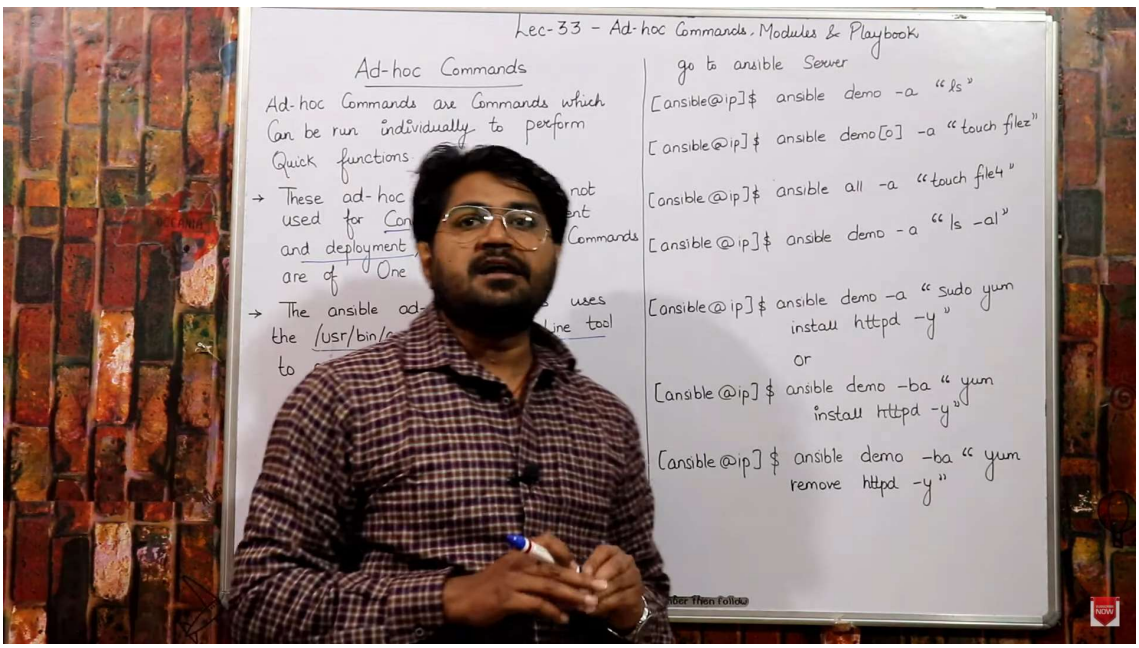


<u>3 ways to push codes from server to nodes:</u>

1. Adhoc commands (simple Linux command) **Disadvantage:** there is no idempotency.
2. Modules -written in YAML (to run one single command)
3. Playbooks- written in YAML (to run more than one module)

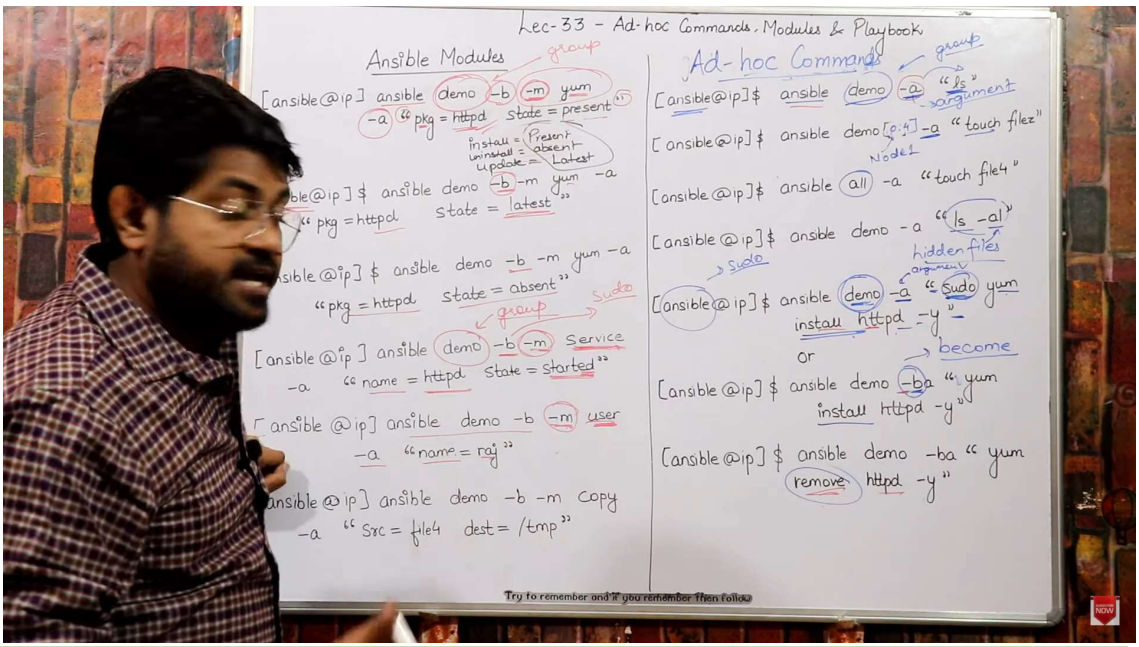
ADHOC COMMANDS

- These can be run individually to perform quick functions.
- These adhoc commands are not used for configuration management and deployment, because these commands are of one-time usage.
- The ansible adhoc command uses the /usr/bin/ansible cli to automate single task.

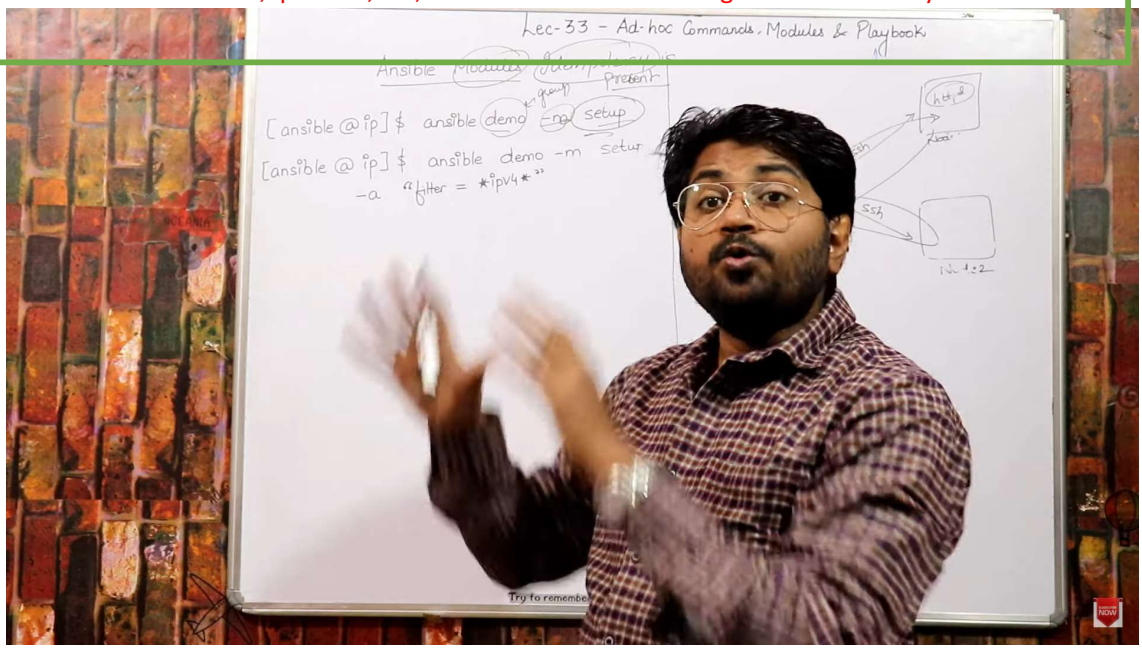


Ansible Modules

- Ansible ships with a number of modules called module library that can be executed directly on remote host via playbooks.
- Your libraray of module can reside on any machine and there are no servers, daemons or database required The defualt location for the inventory file is /etc/ansible/hosts
- To execute a module we need to mention **-m** in command.
- **-b** is used do that we don't have to write sudo to execute the command.

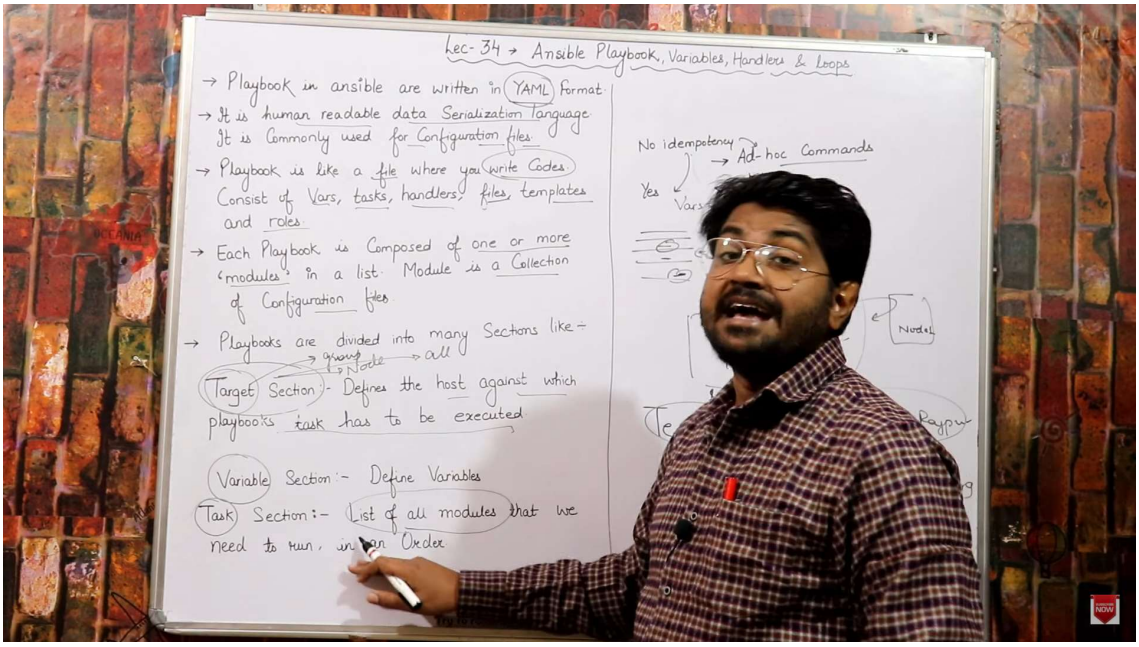


Incase of ansible when a playbook is run then a setup command is executed itself and checks with the node if the package is present on those nodes or not. This set up command helps to achive idempotency. Set up module gives the information about the current configuration of the nodes like location, ipddress, file, folders. It runs in the background continously.

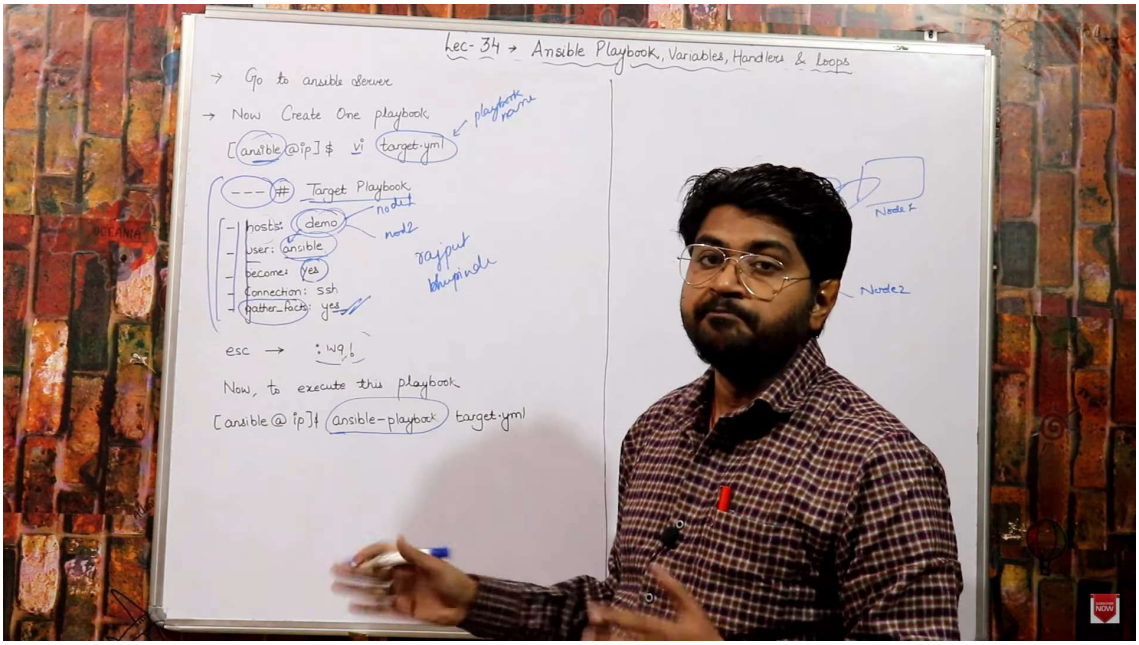


Ansible playbook, variables, Handlers and loops

Intendation: same level things should be written in same line
There should be space between key value pair. Ex: name: rajesh
Become means give sudo privileges.

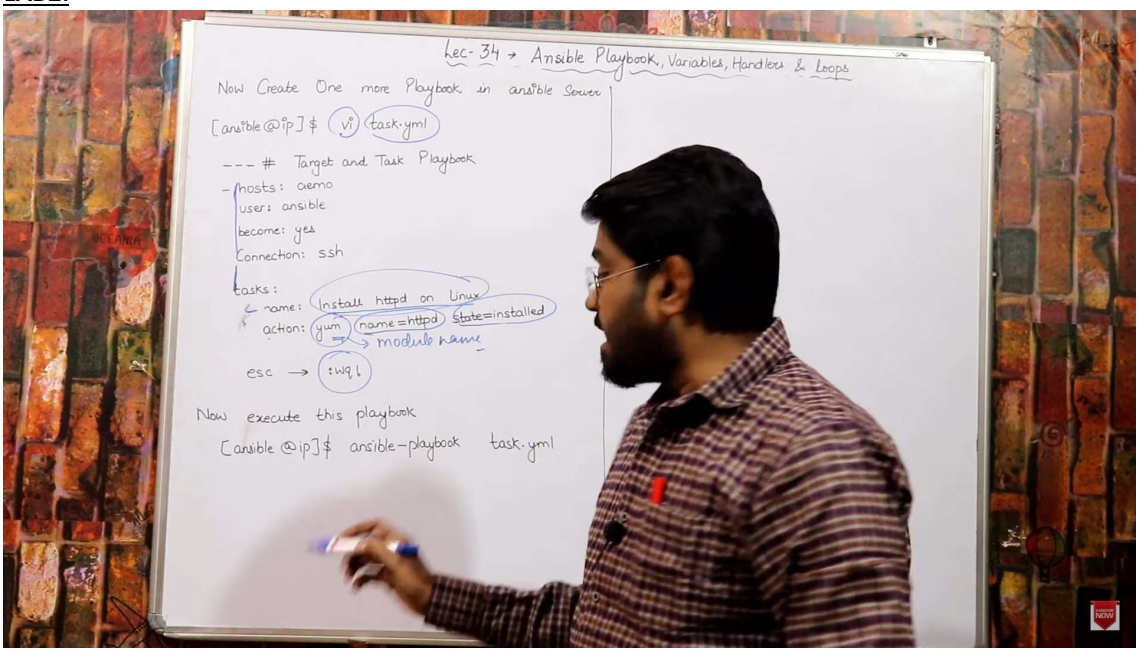


LAB1:



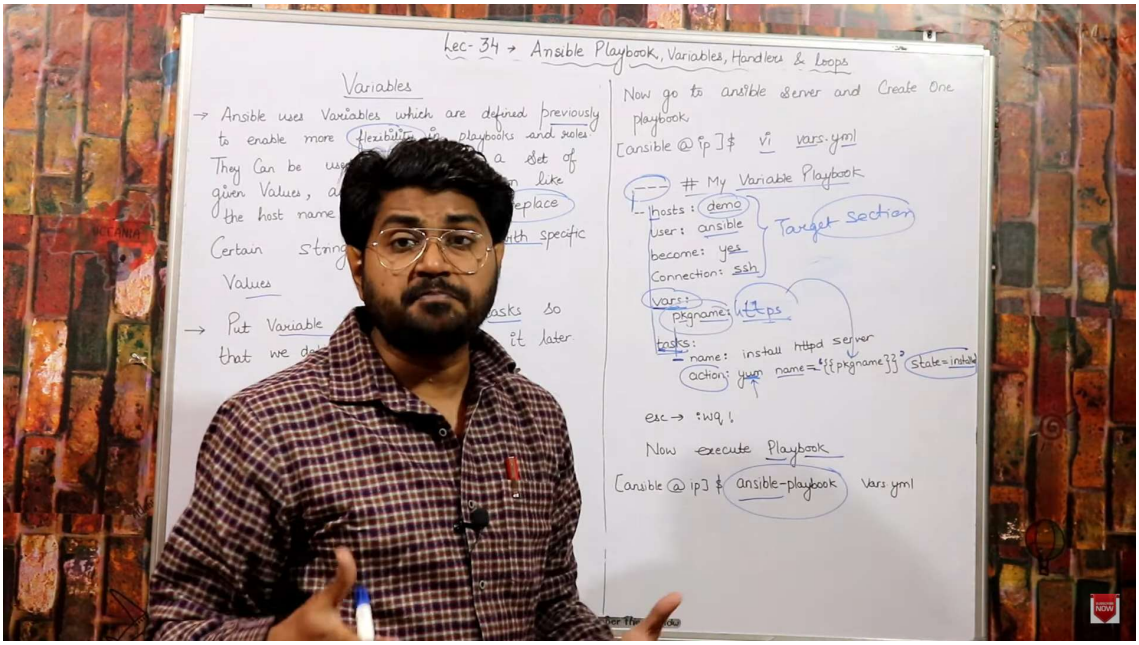
ansible-playbook playbookname /// **run a playbook** ///

LAB2:



LAB3 (varibales)

- Ansible uses variables which are predefined to enable more flexibility in playbooks and roles. They can be used to loop through a set of given values, access various information like hostname of a system and replace certain strings in a templates with specific values.
- Put varibales section above tasks so that we define it first and use it later.



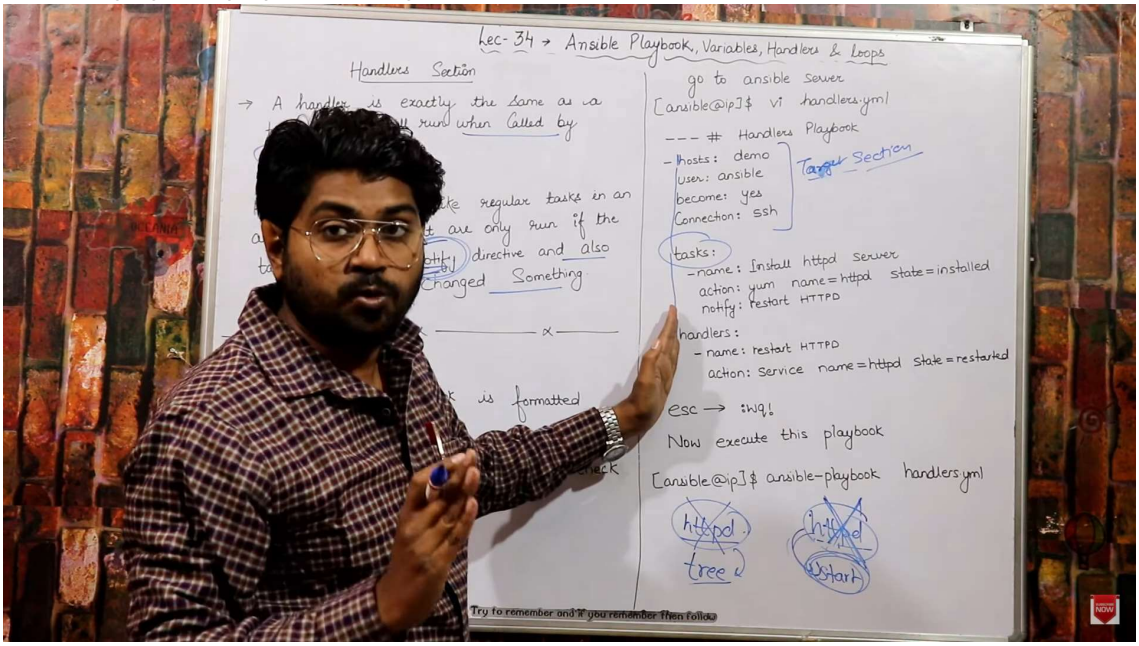
Handlers

- Handler is exactly same as the task but it will run when called by another task.
- Handlers are just like regular tasks n ansible playbook but are only run if the task contains a **notify directive** and also indicates that it changed something.

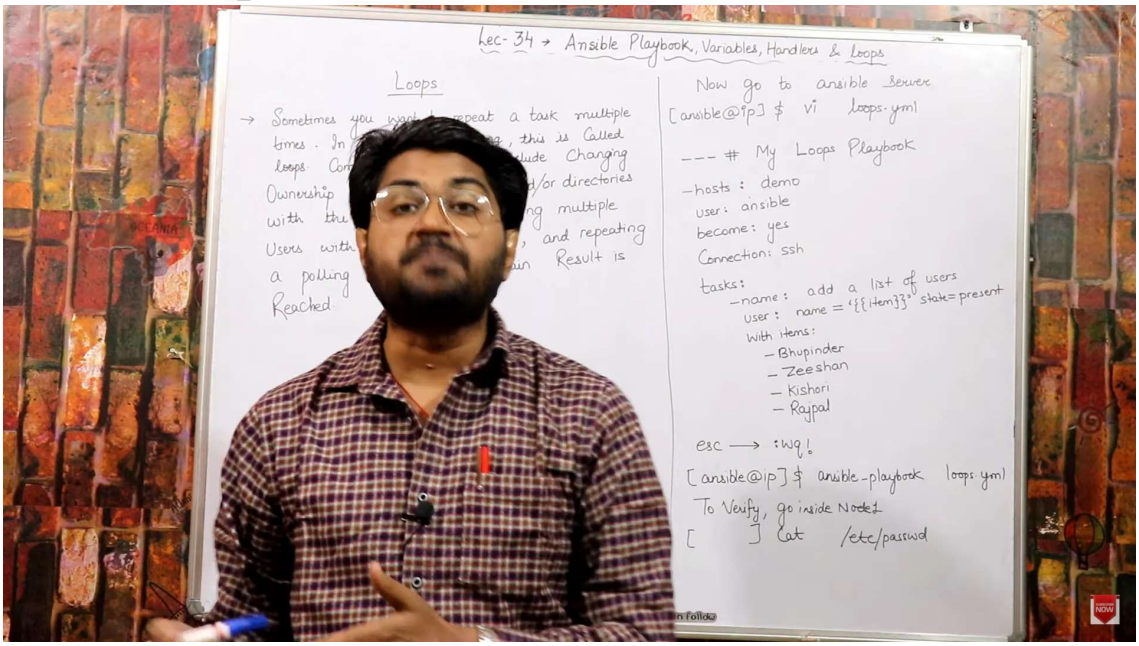
Lab: There should be same thing in task notify and in handler name section so that playbook can identify the dependency and first complete the task then handler.

DRYRUN: check whether the playbook is formatted correctly

 # ansible-playbook playbookname.yml --check



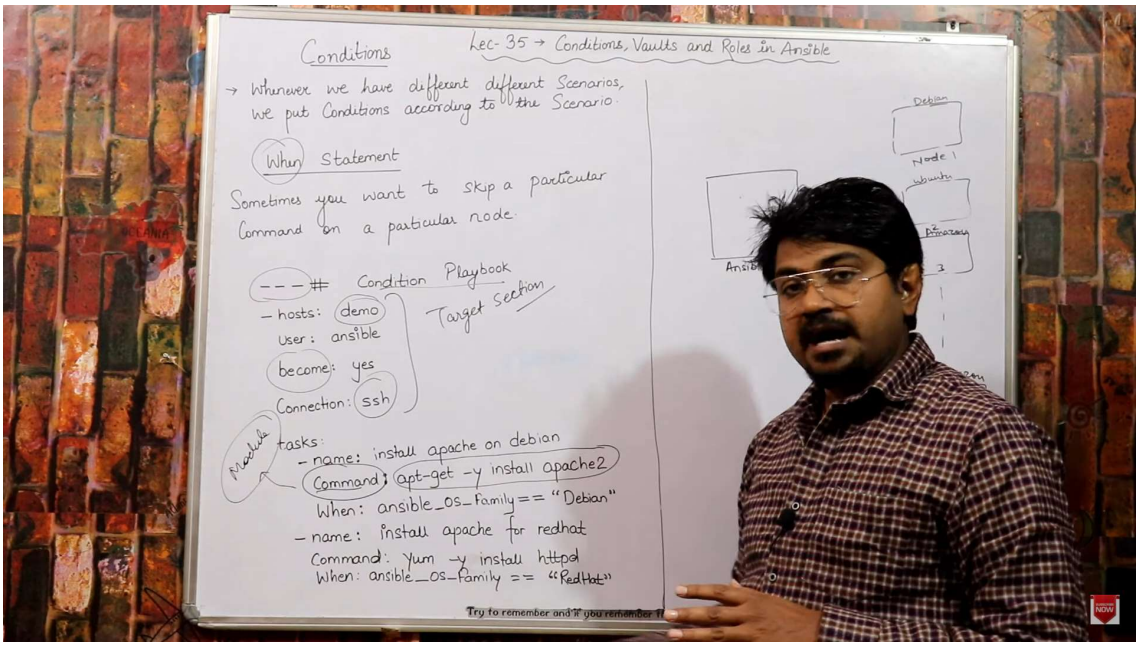
Loops: Sometimes you want to repeat a task multiple times, this is called loop. Common ansible loops includes changing ownership on several files or directories with file module and created multiple users with user module and repeating apolling step until certain result is reached **(it is with_items)**



Conditions

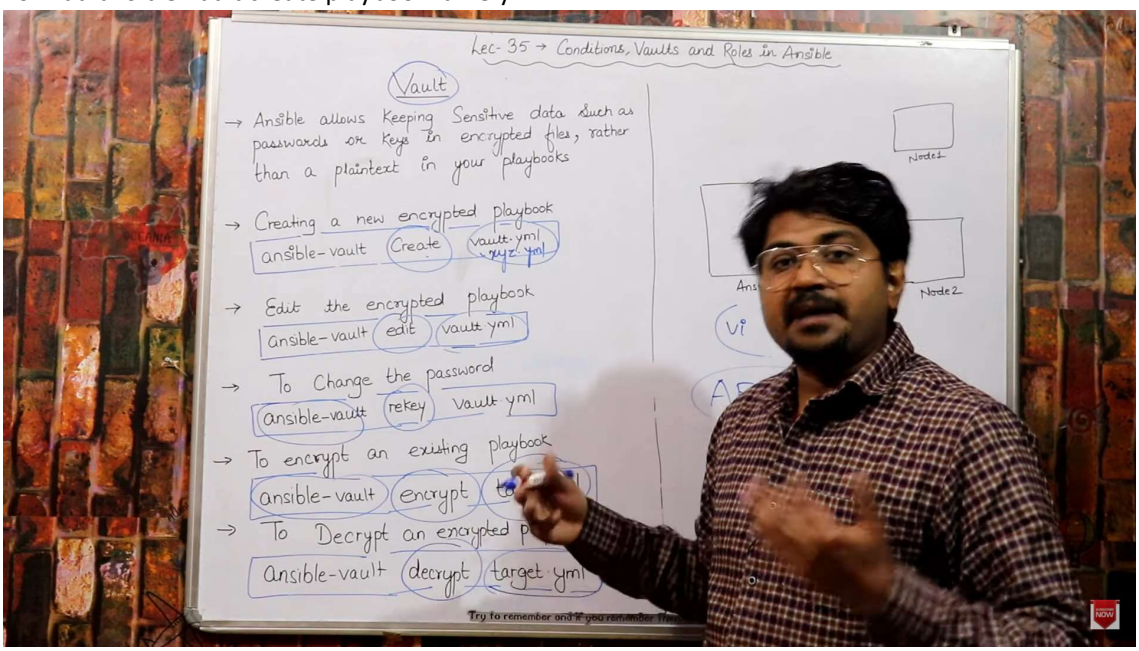
We can use Command moduel to run linux commands.

Note: please make sure of the name spelling.

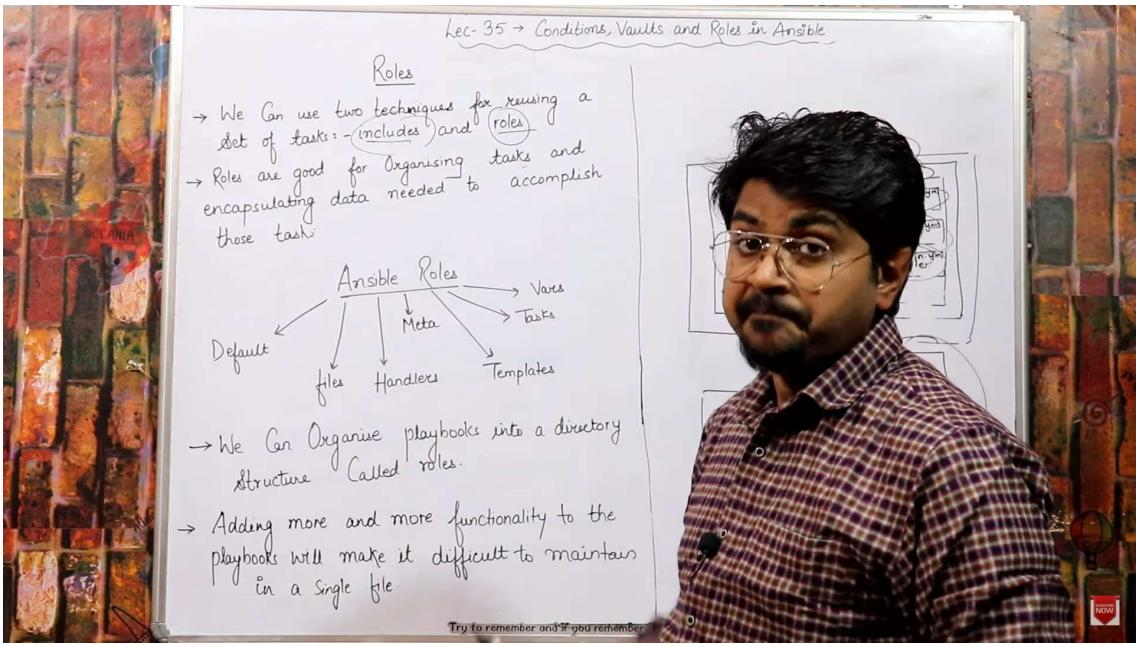


Vault:

We can keep our data in encrypted form rather than in plain text in playbook.
AES256 encryption is used.
Format: ansible-vault create playbookname.yml



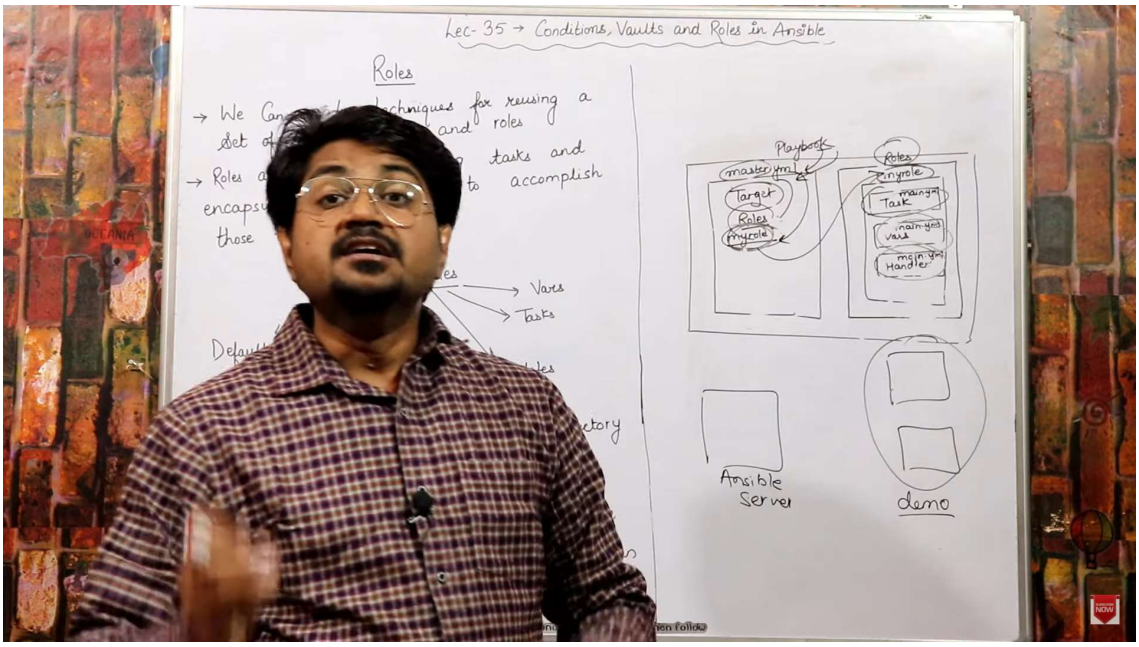
ROLES

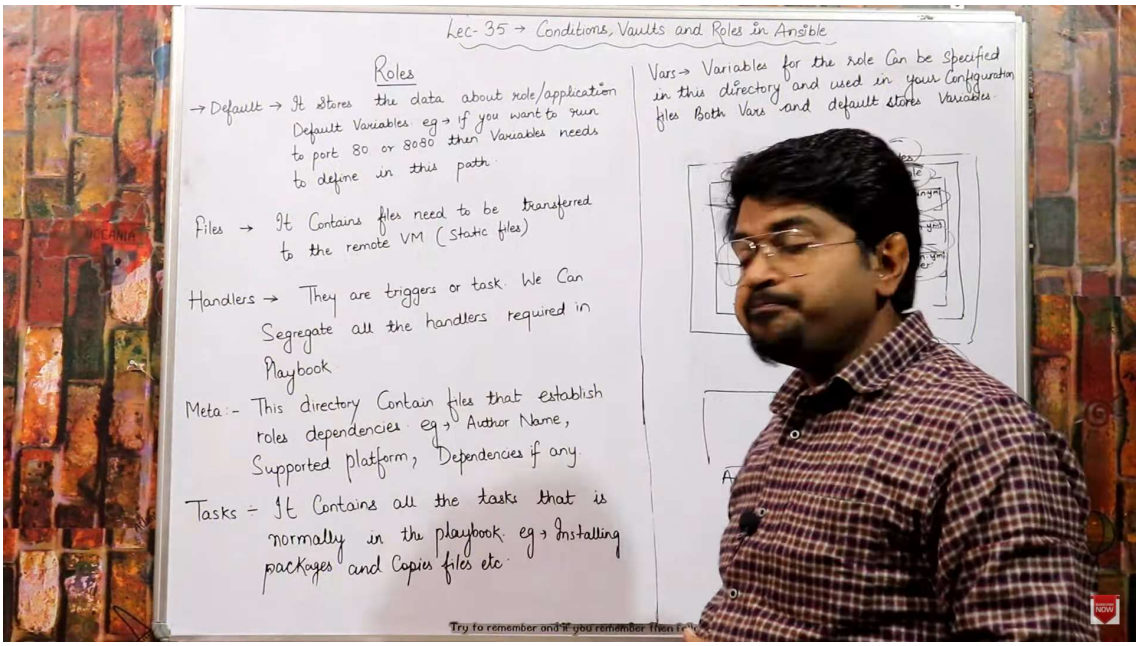


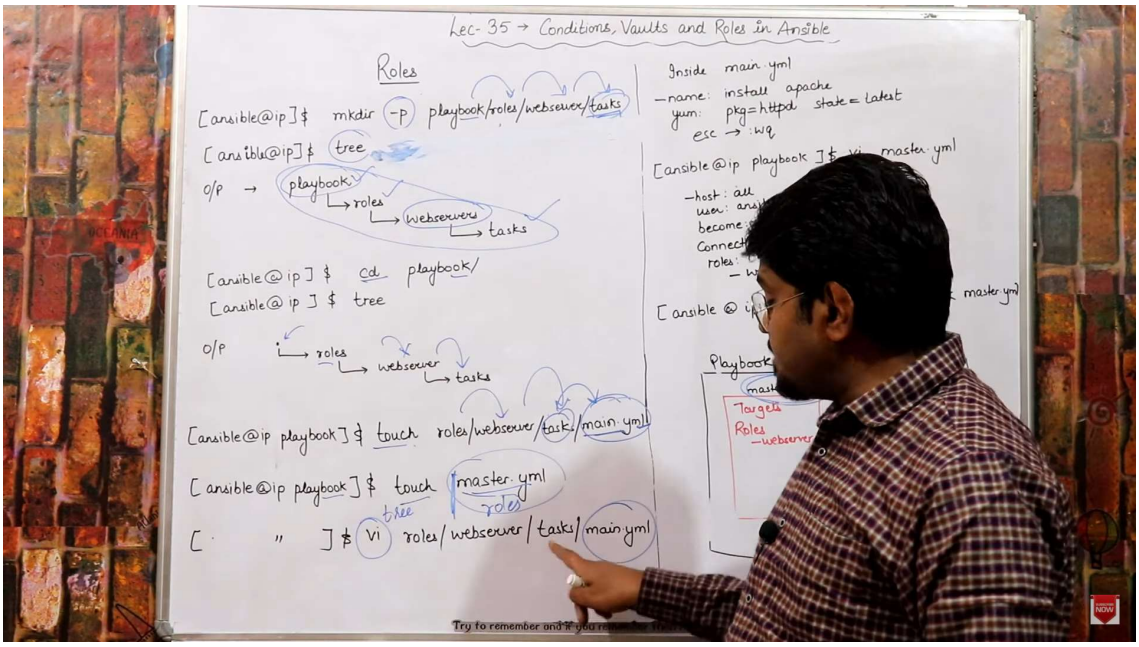
- Jab hum playbook banate hai to usme hum target, handler, task, variables and conditions define krte hai but in real life scenarios playbook me bhaut sare tasks, handlers, conditions hote hai to its very difficulty to identify ke kitne task hai kitne handlers hai kitne conditions hai so to avoid this we define roles.

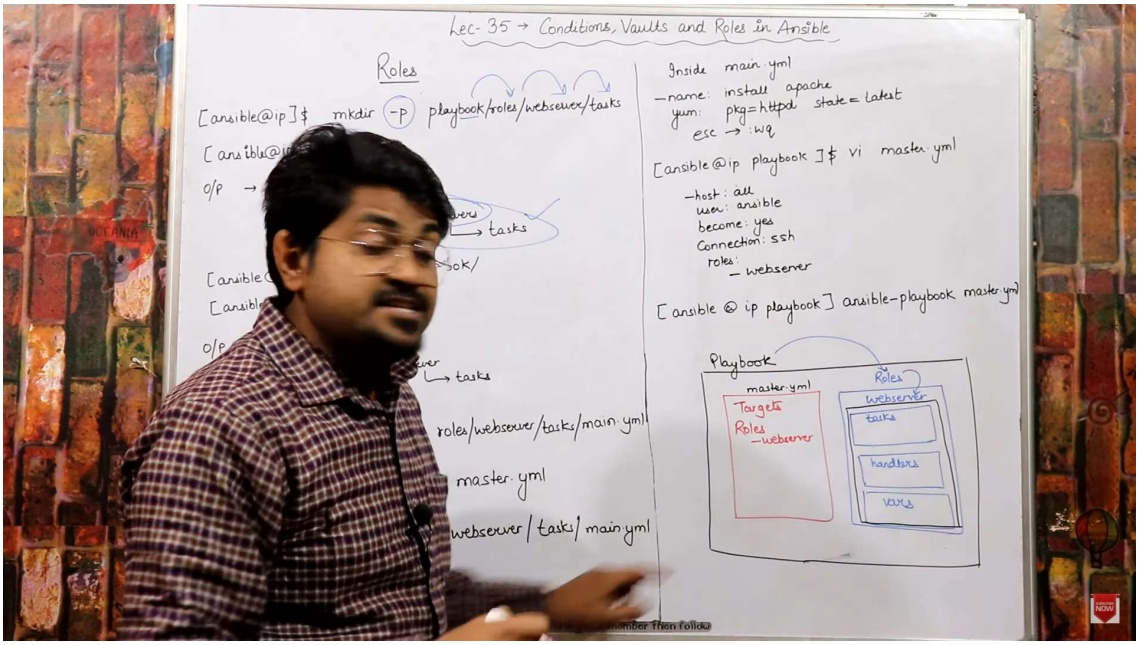
- We create a play book, we create a file in this play book named as **master.yml** and directory as **roles**.
- We then create another directory in Roles and can name it anything for example- myrole
- We then create different verticles in myrole like tasks, handlers, variables, conditions
- We create a file named as **main.yml** in all the verticles
- we define all the handlers in **handler main.yml**, all the tasks in **task main.yml**, all the conditions in **conditions main.yml** and all the varibales in **variables main.yml**
- With this it is easier for us jaise ke kitne tasks likhe hai playbook me so poore playbook pdne ke wajahe hum sedha task.yml me jaakr dekh skte hai.
- Jaise he master.yml run hoge to wo pehle >>> target(hmare nodes) ko call krega then >>> roles >>> ar roles apne andr myroles ko >>> ar myroles sare main.yml ko call krega jo hmne task, condition, varibale handlers me likhe honge.
- We define **targets** in master.yml

```
-host: all
 user: ansible
 become: yes
 connection: ssh
 roles:
        -   myrole
```


Lec- 35 → Conditions, Vaults and Roles in Ansible
Roles
→ We Can ... techniques for reusing a set of ... and roles
→ Roles ... tasks and encaps... to accomplish those
... Vars
... Tasks
... tory
Playbook
master.yml
Target
Roles
myrole
Roles
myrole
main.yml
Task
main.yml
vars
main.yml
Handler
Ansible Server
demo


Lec- 35 → Conditions, Vaults and Roles in Ansible
Roles
→ Default → It stores the data about role/application Default Variables eg → if you want to run to port 80 or 8080 then Variables needs to define in this path.
Files → It Contains files need to be transferred to the remote VM (Static files)
Handlers → They are triggers or task. We Can Segregate all the handlers required in Playbook
Meta - This directory Contain files that establish roles dependencies eg, Author Name, Supported platform, Dependencies if any
Tasks - It Contains all the tasks that is normally in the playbook eg, Installing packages and Copies files etc.
Vars → Variables for the role Can be specified in this directory and used in your Configuration files Both Vars and default stores Variables


Lec- 35 → Conditions, Vaults and Roles in Ansible
Roles
[ansible@ip]$ mkdir -p playbook/roles/webserver/tasks
[ansible@ip]$ tree
o/p → playbook
roles
webserver
tasks
[ansible@ip]$ cd playbook/
[ansible@ip]$ tree
o/p roles
webserver
tasks
[ansible@ip playbook]$ touch roles/webserver/tasks/main.yml
[ansible@ip playbook]$ touch master.yml
tree
roles
[ ]$ vi roles/webserver/tasks/main.yml
Inside main.yml
-name: install apache
yum: pkg=httpd state=latest
esc → :wq
[ansible@ip playbook]$ vi master.yml
-host: all
user: ansible
become: yes
Connection: ssh
roles:
- webserver
[ansible@ip playbook] ansible-playbook master.yml
Playbook
master.yml
Targets
Roles
-webserver
Roles
webserver
tasks
handlers
vars


Lec- 35 → Conditions, Vaults and Roles in Ansible
Roles
[ansible@ip]$ mkdir -p playbook/roles/webserver/tasks
o/p →
tasks
roles/webserver/tasks/main.yml
master.yml
webserver/tasks/main.yml
Inside main.yml
-name: install apache
yum: pkg=httpd state=latest
esc → :wq
[ansible@ip playbook]$ vi master.yml
-host: all
user: ansible
become: yes
Connection: ssh
roles:
- webserver
[ansible@ip playbook] ansible-playbook master.yml
Playbook
master.yml
Targets
Roles
-webserver
Roles
webserver
tasks
handlers
vars