**Configuration Management(CM)**

Configuration management (CM) refers to the process of systematically handling changes to a system in a way that it maintains integrity over time.

**CM helps to implement**

* Policies
* Procedures
* Techniques
* Tools

**Advantages of CM**

1. Increase Uptime
2. Improve Performance
3. Ensure Compliance(exact requirement)
4. Prevent Errors
5. Reduce Cost

Tools: Chef, puppet, Ansible, Salt

**Ansible**

* Ansible is an automation engine, that automates software provisioning, Configuration management and application deployment
* Manages infrastructure whether it is on-premises or in the cloud.
* It turns your infrastructure as code i.e you are going to manage or build your application from code. your computing environment has some of the same attributes as your application.

1. Your infrastructure is versionable(you can store it in central repo and manage, edit as version wise)
2. Your infrastructure is repeatable (you can use the same configuration file again & again)
3. Your infrastructure is testable

* Here we only needs to tell what the desired configuration should be(what to do), not how to achieve it.
* It is also called as Infrastructure as code(IAC)

**Why Ansible**

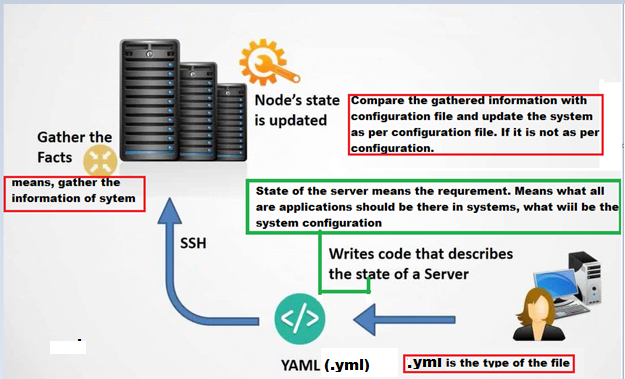
It is an open source tool.

* Other tools are huge overhead of Infrastructure setup
* Other tools are complicated setup
* Other tools are working in Pull mechanism(Server-Client module)- It means there should be a server for all configuration and clients will pull the configuration from server.
* Lots of learning required to manage

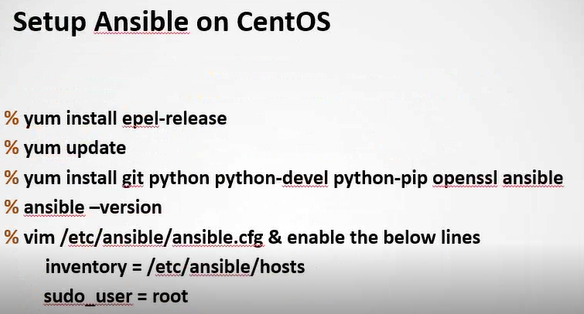
**Advantages of Ansible**

* Agentless : means no need to install Ansible in each and every system
* Relies on SSH
* Uses python
* Push mechanism: no need to do anything in all machines. Only we have to install Ansible in one system and push the configuration to all other systems.

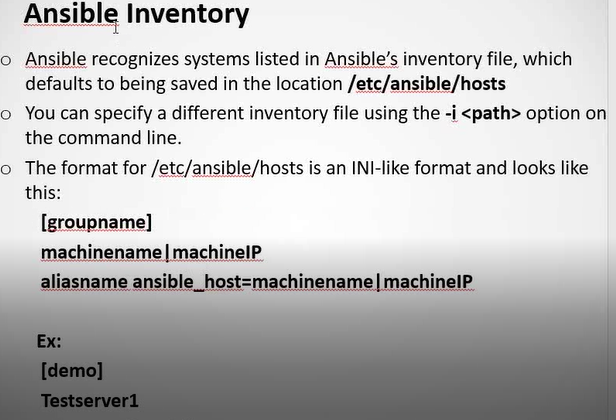
**How Ansible Works**

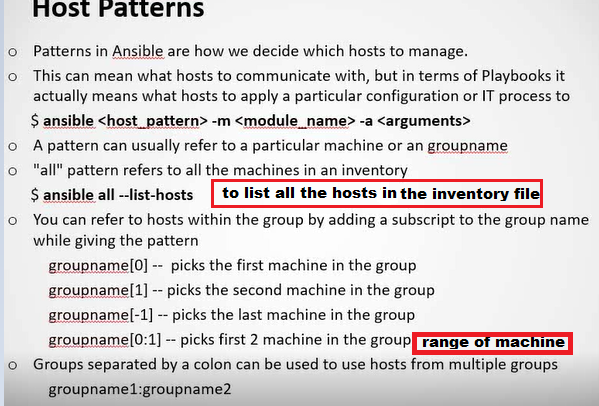




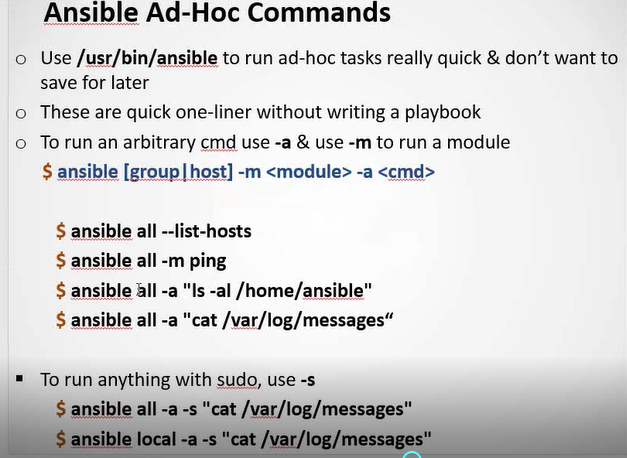




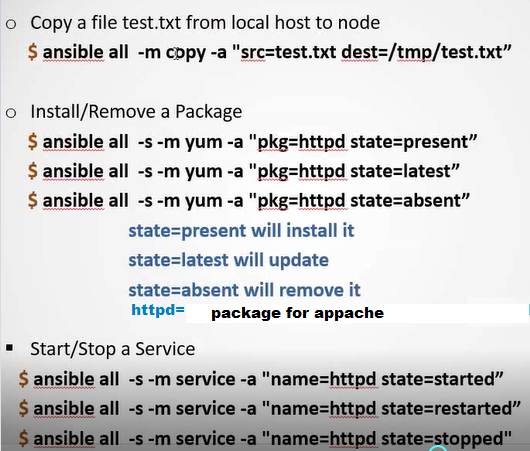




* Ansible <group name> --list-hosts : to list the all host in the particular group



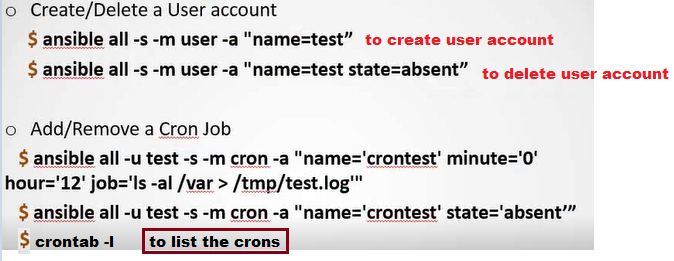
* <module>module basically used for executing operating system command(exa: ping)



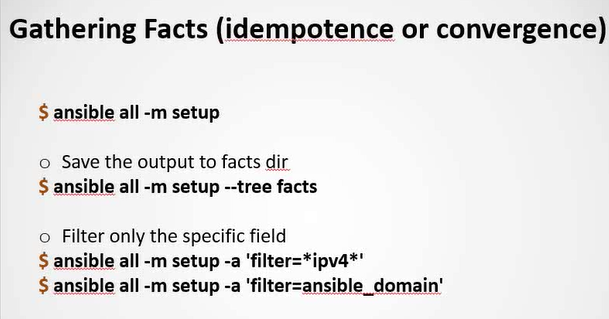
* Here yum & service are the module.

Note:

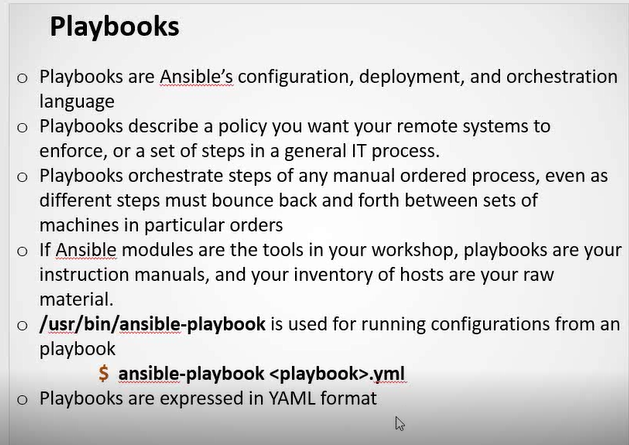
* If Ansible had done any changes(modified) in the destination machine by using the Ansible executable, the result will be in yellow color.
* If Ansible not done any changes in destination file, but command run successfully, the result will be in green color.
* If Ansible failed to execute , then result will be in red color

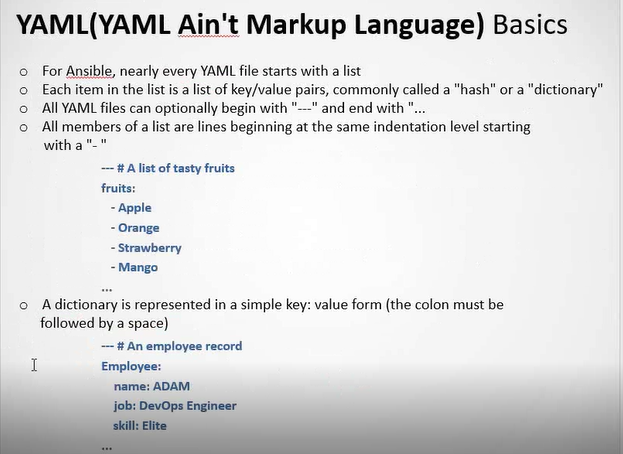


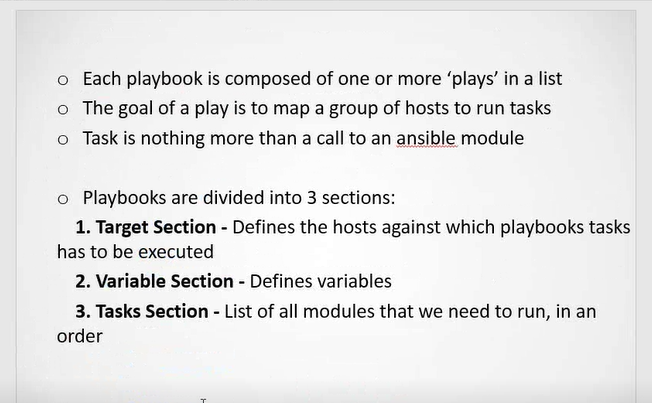
* Here job= the task you wants to do

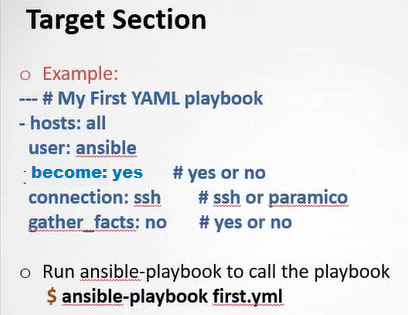


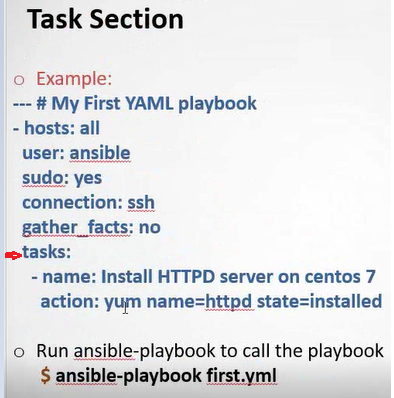
Gathering the facts(machine information) is called as idempotence or convergence.

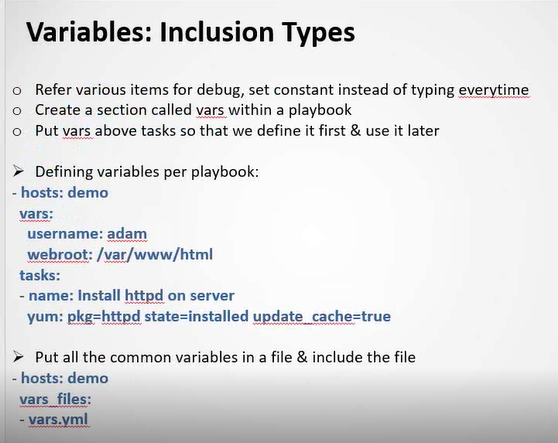


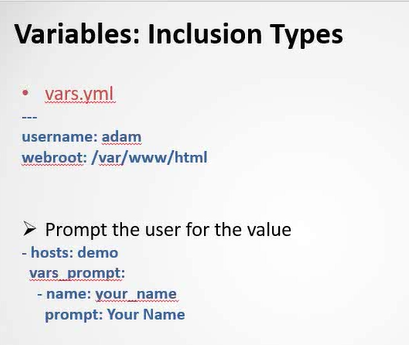


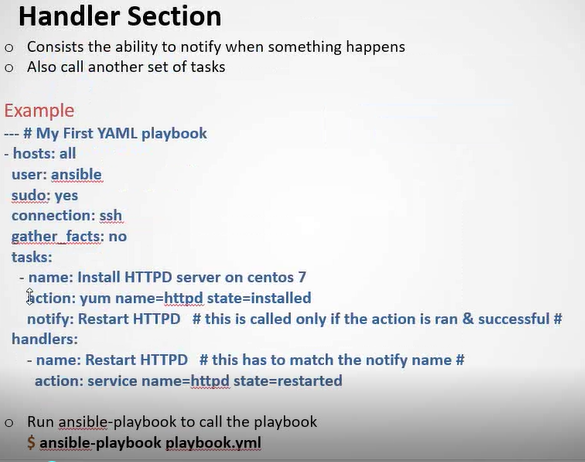


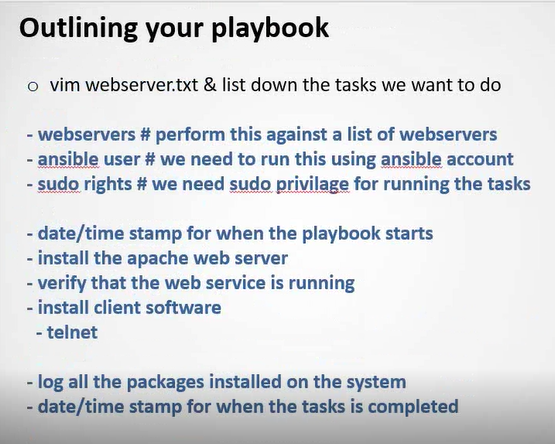


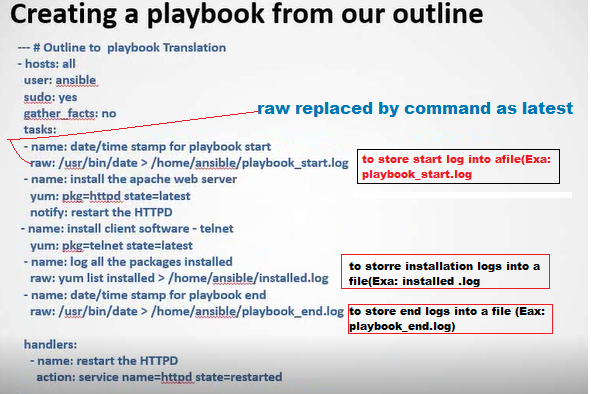




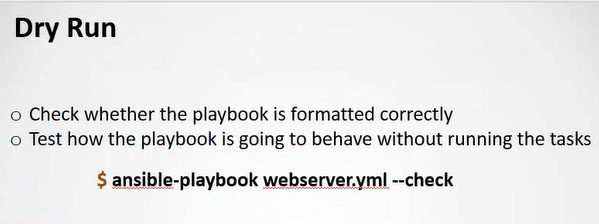




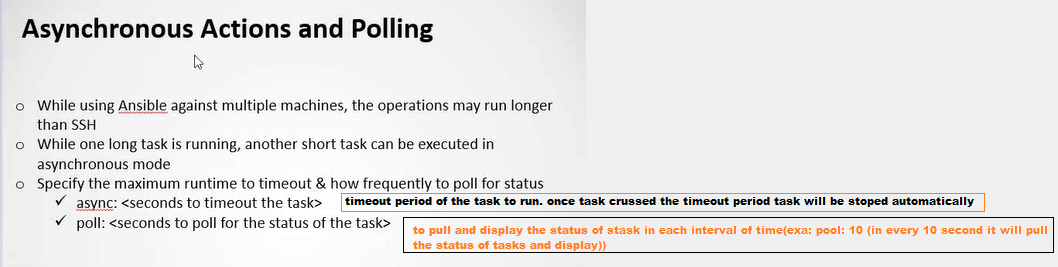




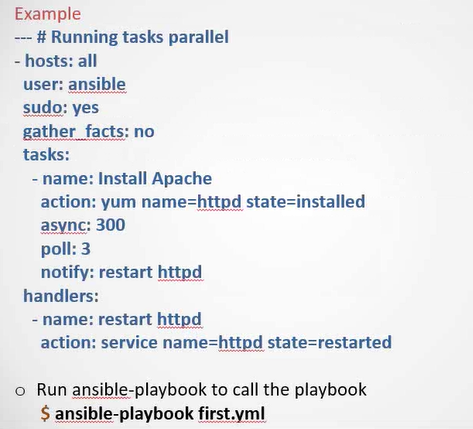
* raw: used to call operating system command

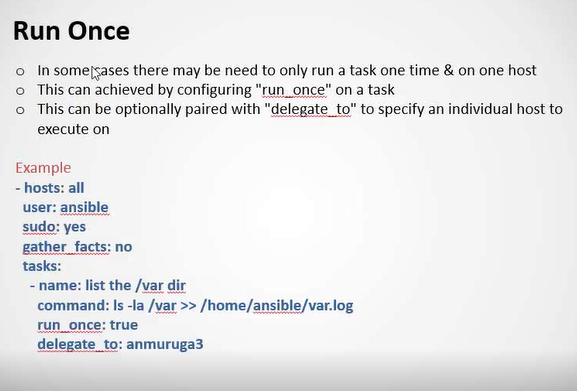


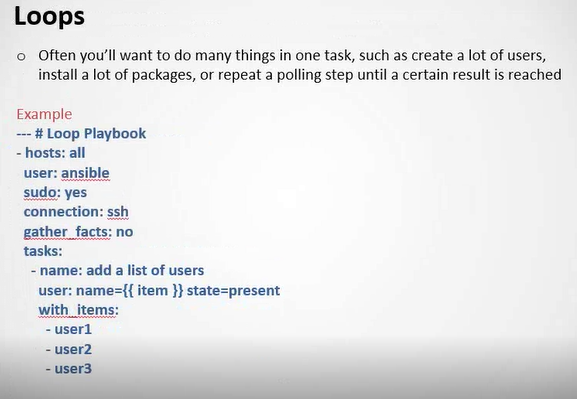
* here **webserver.yml** is the playbook name and **--check** is the command to check or test the playbook.
* It is not going to execute the task.
* Dry Run basically used to test the playbook. Like is it working fine or it has some syntax error or if it will run what is going to do. It’s one type of pre test



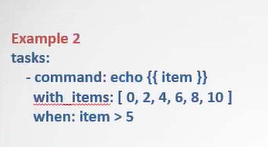
* By using **async** we can run multiple task parallel

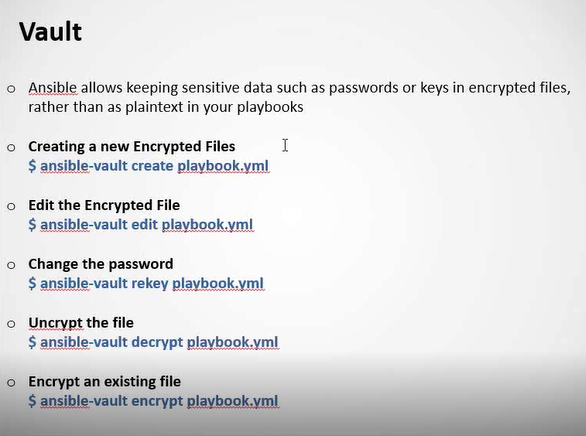






**Combine Loop & condition**



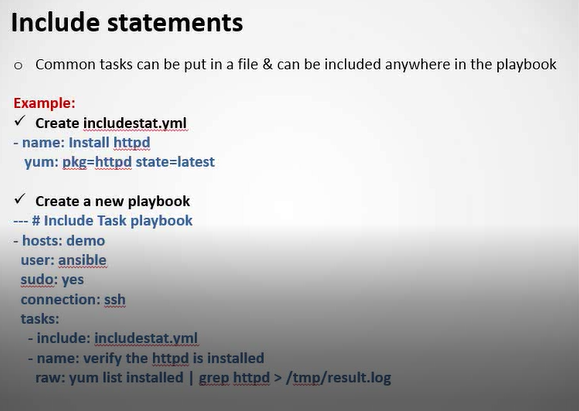


* **View encrypted file**

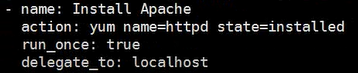
**$ ansible-vault view playbook.yml**

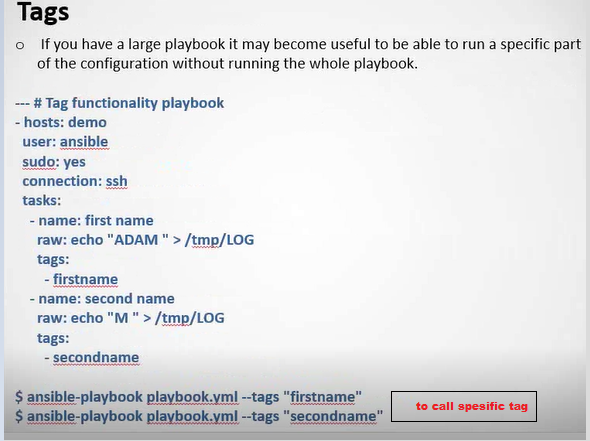
* vault allow to encrypt the files.

Note: We can declare(include) a **.yml** file inside in another **.yml** file.

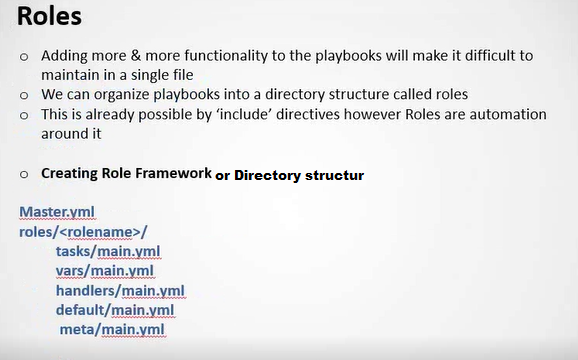


**Exa of includestat.yml**

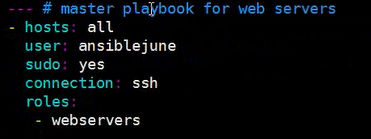


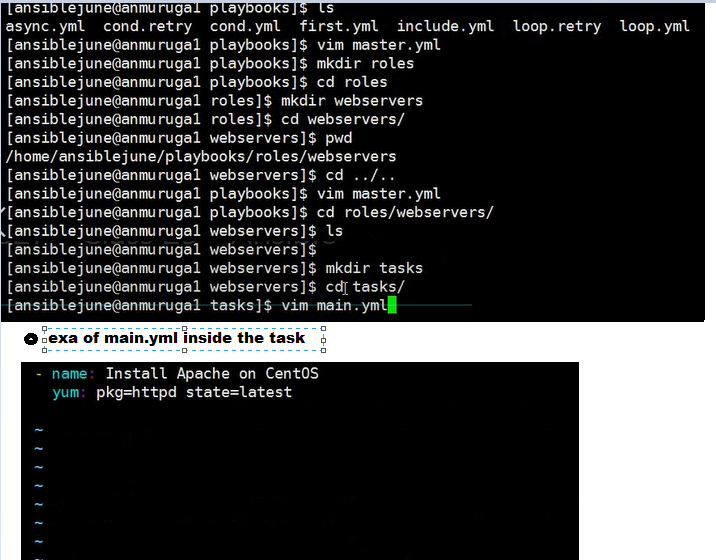


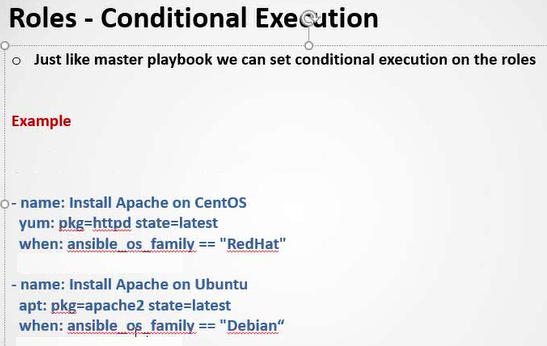


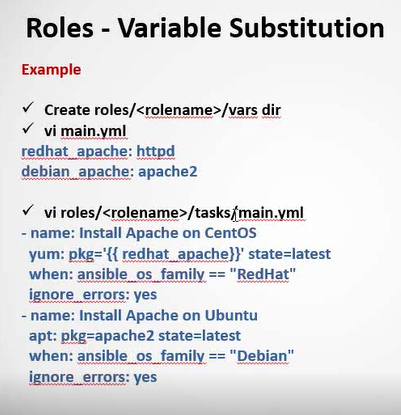


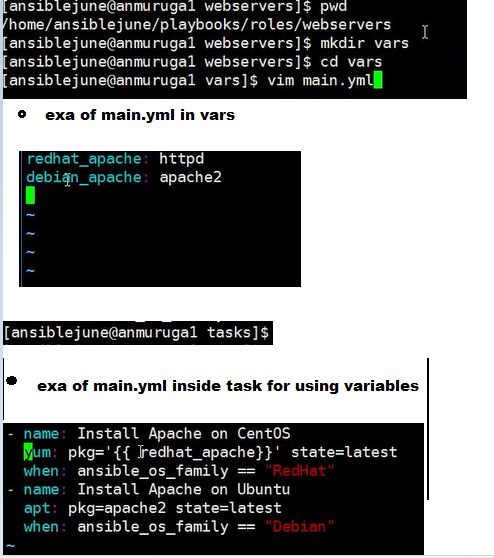
Exa of Master yml file

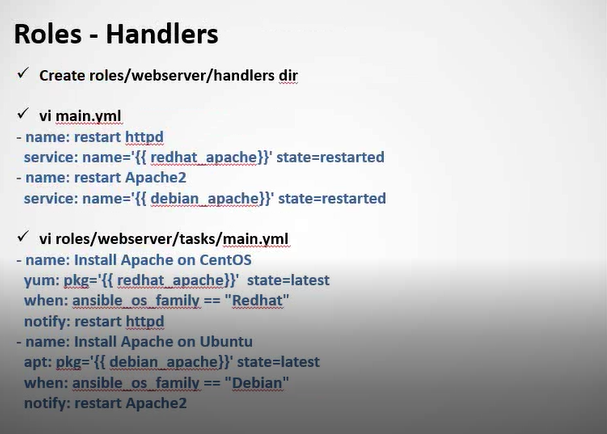


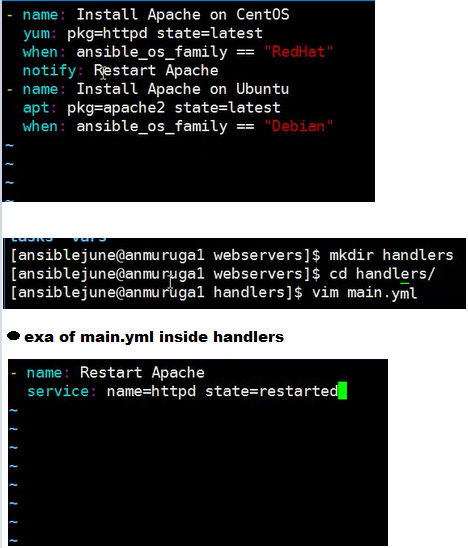


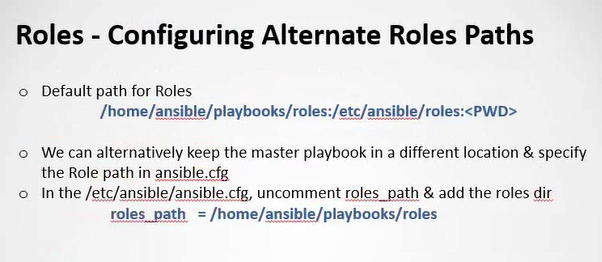


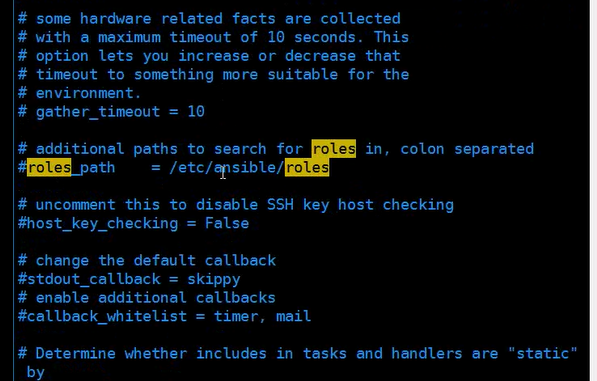


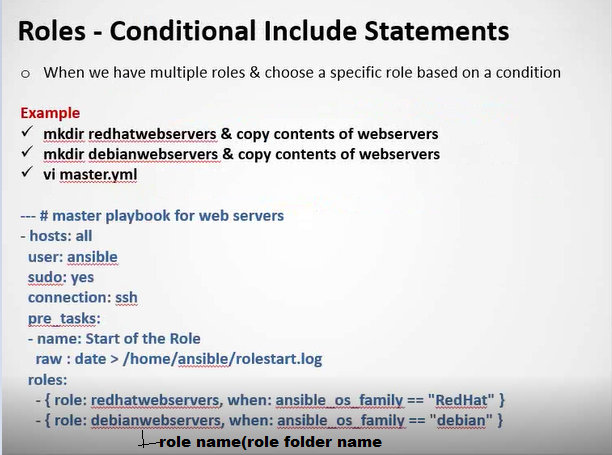


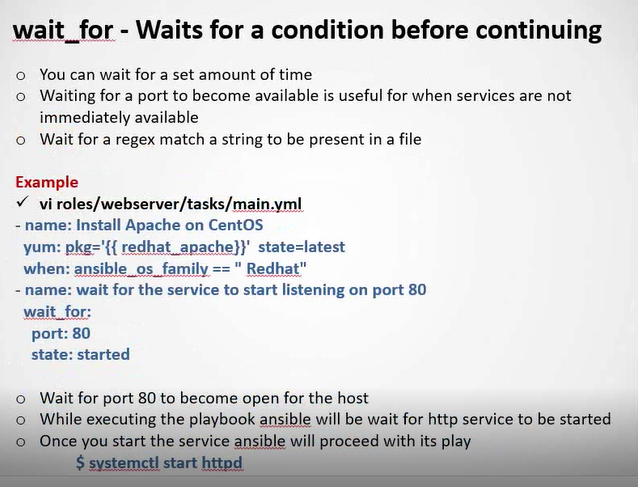


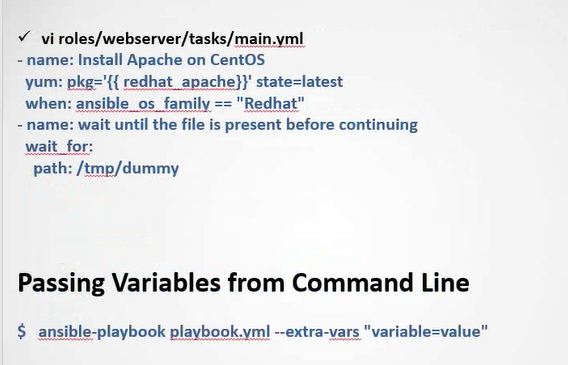










re

* **--extra-vars** used for pass runtime variable while calling playbook