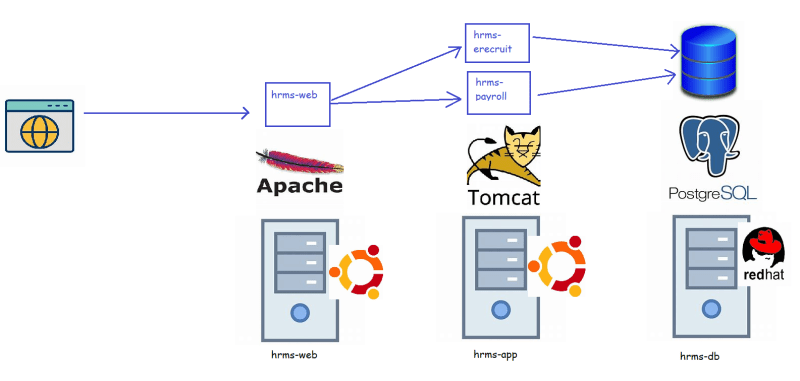
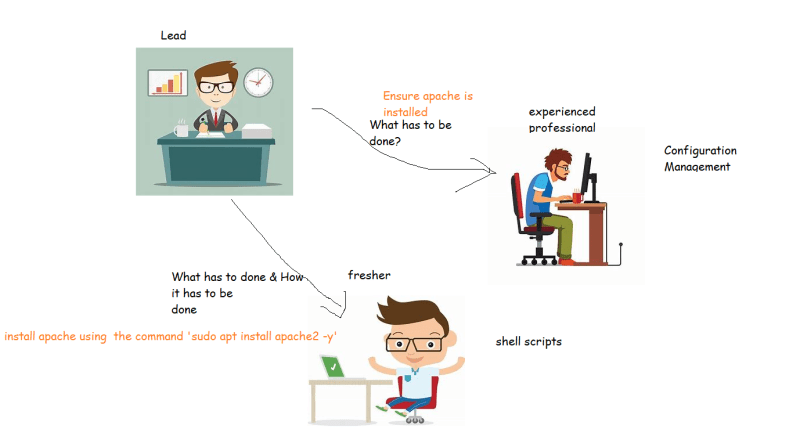
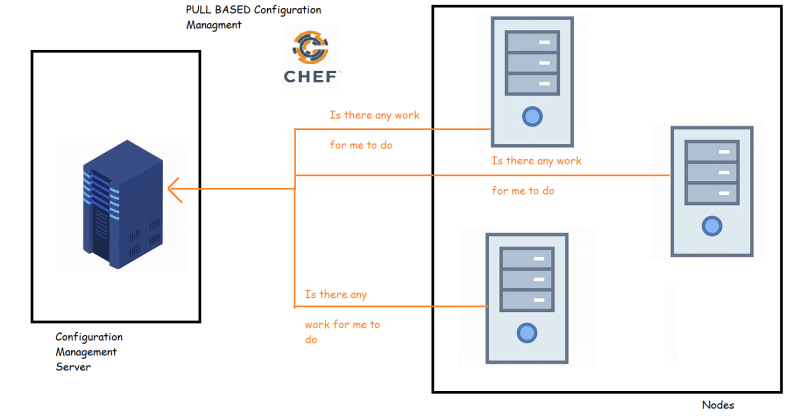
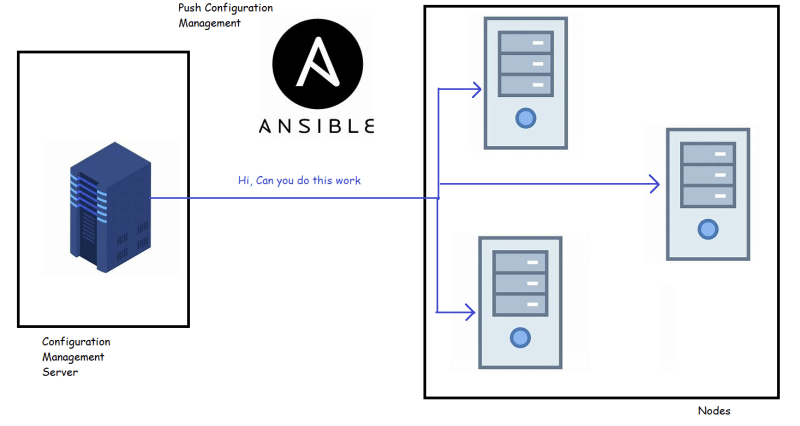
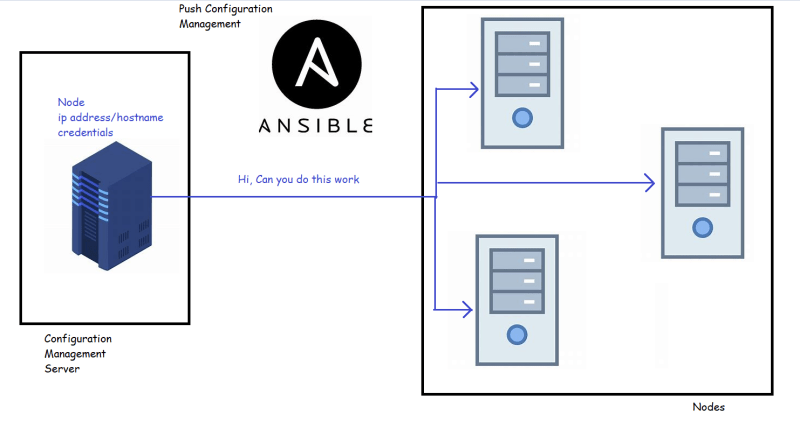
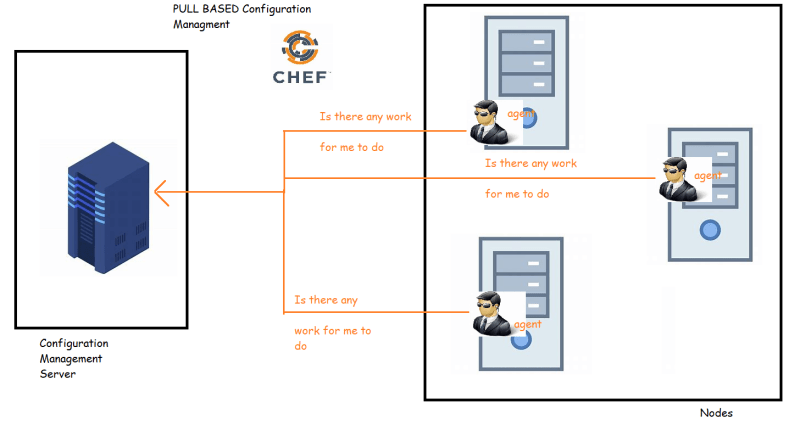
, 0000OCTOBER 5, 2021

DevOps Classroom Series – 05/Oct/2021

Story of an Enterprise

* This is a story of a fictitious IT Enterprise LearningThoughts which is developing an HRMS application
* The architecture of the application is as follows 
* Dev Team is working on building new features across all the server components
* The basic idea is to deploy the application into server’severy day at 11:00 PM and start running automated tests. This will give confidence to testing team to work on new version.
* Every weekend we need to create an environment and deploy the work of the complete week run the whole tests and make this version available for demo & customer usage (beta)
* So, this organization is looking out for a solution to handle deployments (daily, weekly) effectively
* Lets explore different options
  + Manual Deployments: This is not a sensible option is it time taking and there is chance of human error
  + Deployment using Shell/Bash scripts:
    - Better than manual
    - Shell scripts are not readable &main table
    - Writing a script which when executed n times gives the same result is challenging.
    - Scripting is a procedural approach (We need to tell what & how it has to be done)
  + Configuration Management:
    - We use the declarative approach (We specify what has to be done/what we want)
    - When we use CM the script is readable.
    - When we run CM n times we always get the same result (idempotence) 

Configuration Management

* Basic Models of Configuration Management (PULL vs PUSH)  
* Push based CM needs node information such as Hostnames/ip addresses and credentials to login 
* Pull based CM needs node to install agents which are responsible for communication 

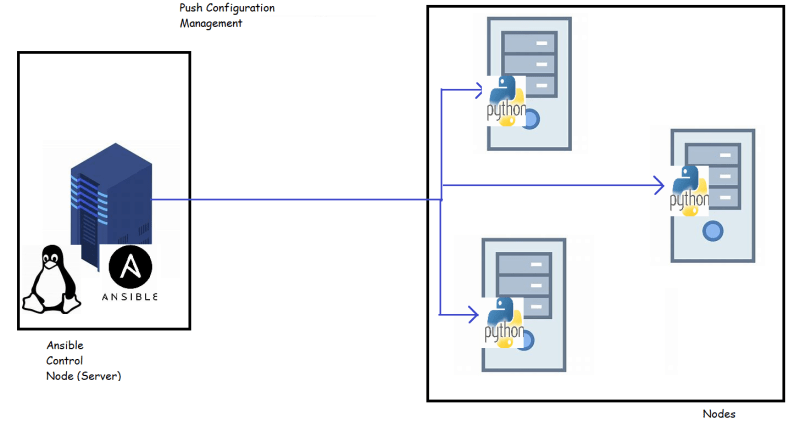
Exercise

* Watch 1-7 videos of DevOps Essentials [Refer Here](https://www.youtube.com/watch?v=dcWAf6FOOc0&list=PLuVH8Jaq3mLud3sVDvJ-gJ__0zd15wGDd)
* Linux class room videos [Refer Here](https://www.youtube.com/watch?v=1oMBgXsMtCk&list=PLuVH8Jaq3mLtx82QrxoUre38AqJQwQ3bs)
* System Setup for DevOps on your system (Windows): [Refer Here](https://www.youtube.com/watch?v=mRILfUNbsIo&list=PLuVH8Jaq3mLud3sVDvJ-gJ__0zd15wGDd&index=14)
* For Windows 10/11 users setup Windows Terminal [Refer Here](https://www.youtube.com/watch?v=qLVn2EvPsYc&list=PLuVH8Jaq3mLud3sVDvJ-gJ__0zd15wGDd&index=11)

OCTOBER 6, 2021

# DevOps Classroom Series – 06/Oct/2021

## How Ansible Works

* Configuration Management server in the case of the Ansible is called as Ansible Control Node, which can be any Linux, UNIX or mac instance.
* On all the Nodes Python should be installed 
* Ansible uses python to do configuration management on the nodes, As an ansible user we would not need to know python to do Configuration Management
* Now let’s consider that we would want to install apache on the web1, web2 and web3 nodes
  + Ansible should be configured
  + For doing deployment/installation we need to write a playbook which is a yaml.
  + Now we would ask ansible to execute playbook, Now Ansible will do the following
    - Generate a Python script that installs the apache
    - Copy the script to web1, web2 and web3
    - Execute the script on web1, web2 and web3
    - Wait for the script to complete and show the results
  + For copying and executing the python script created from playbook, Ansible needs to know the server details and credentials, we create a file called as inventory with server details where configuration management has to be executed.

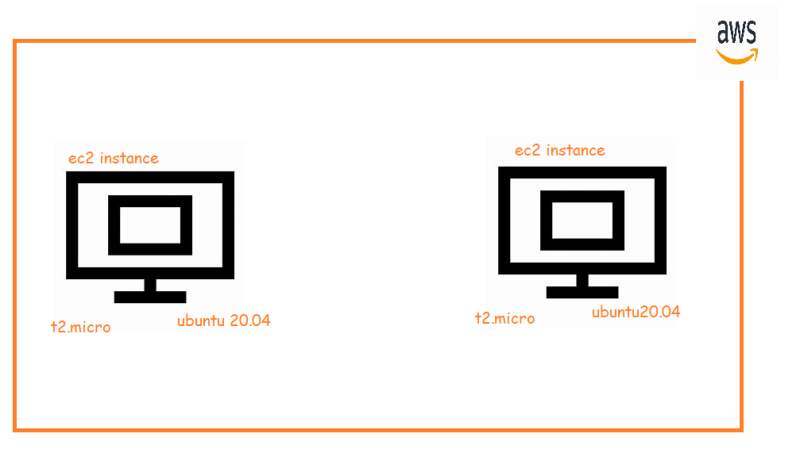
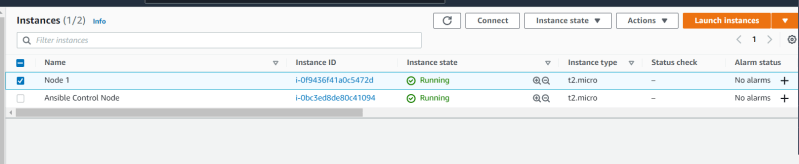
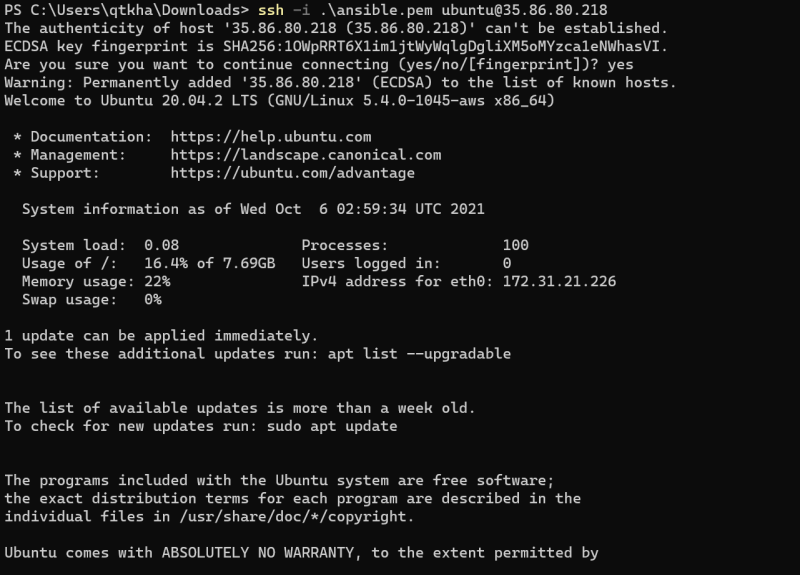
## What’s so great about Ansible

* Simple: Ansible was designed to have a simple setup process & minimum learning curve
* Easy to Read Syntax
* Easy to Audit
* Nothing to Install on Remote Hosts
* Easy to Share
* Powerful
  + Batteries
  + Paralled Execution
  + Master less

## What Doi Need to Know to work with Ansible

* The following are the topics which we need to know about linux systems
  + Connect to a remote machine using SSH
  + Interact with command line shell
  + Install packages
  + Using the sudo command
  + Check and set file permissions
  + Start and stop Services
  + Set Environment variables

## Ansible Lab Environment

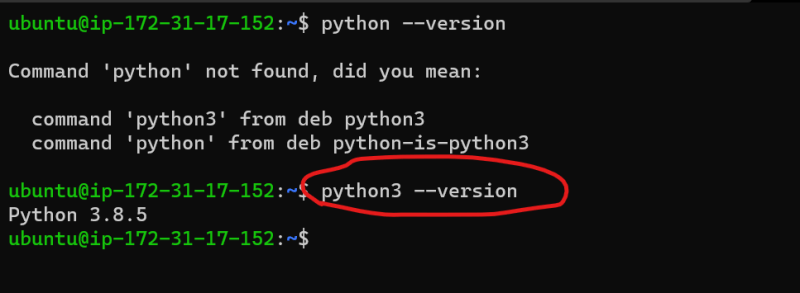
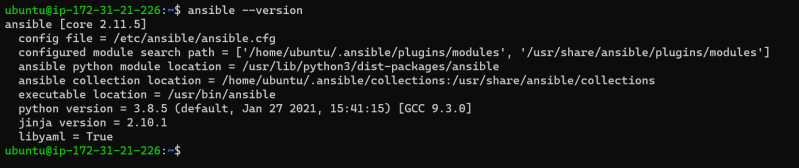
* We would create two ec2 instances in AWS with ubuntu 20.04  
* Note:
  + Creating AWS EC2 [Refer Here](https://www.youtube.com/watch?v=me2s3mTNwGo&list=PLuVH8Jaq3mLszrC7lv68a0VcrDripW-HK&index=2)
  + Creating Azure VMs [Refer Here](https://www.youtube.com/watch?v=P9X-4Z-NeGg&list=PLuVH8Jaq3mLuqXuGs6aeqxhuvCYSzB1kT&index=2)
* Now let’s try to install ansible on one node which would be ansible control node
  + Log in to the ansible control node 
  + Install on Ubuntu [Refer Here](https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html#installing-ansible-on-ubuntu)

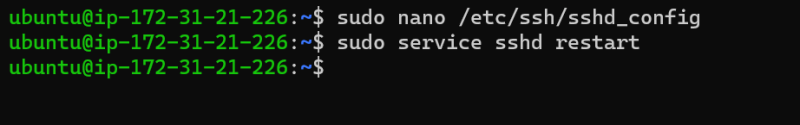
sudo apt update

sudo apt install software-properties-common

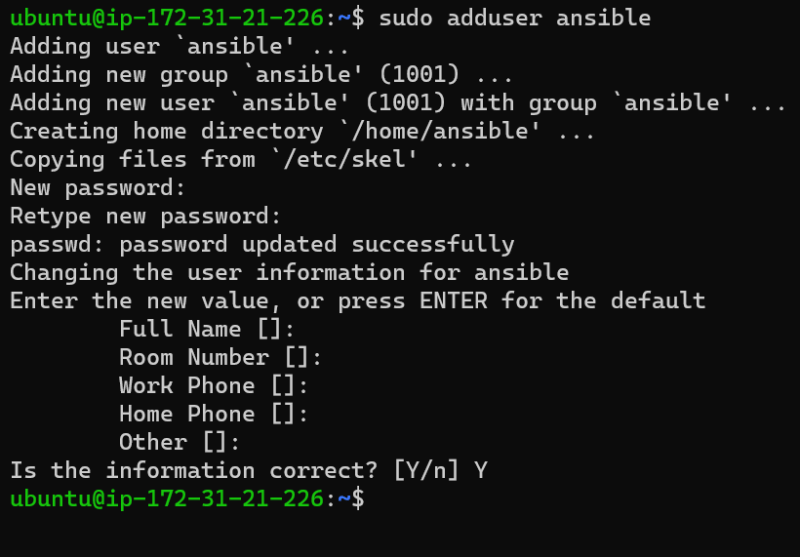
sudo add-apt-repository --yes --update ppa:ansible/ansible

sudo apt install ansible -y



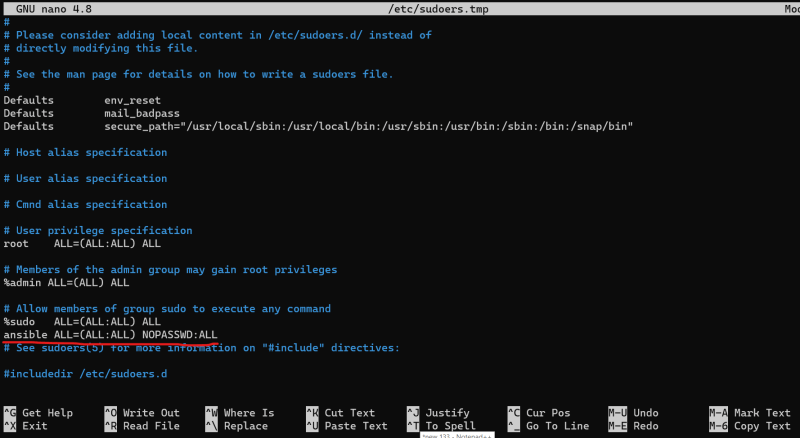
* + Now let’s try to install python on the node 1. Python is already installed
  + In AWS instances, the password is disabled by default. So lets enable password based authentication on both ansible control node and node1
    - In the /etc/ssh/sshd\_config lets set the value of PasswordAuthentication
    - yes 
  + Then we would create an ansible user which simulates it admin on both the machines.
    - Log in to nodes and execute

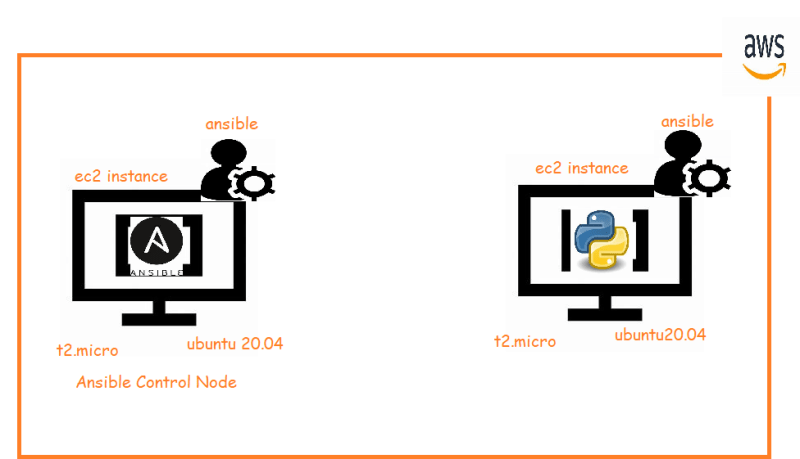
sudoadduser ansible



* + Now we need to give ansible user sudo privileges without prompting for password.

sudovisudo

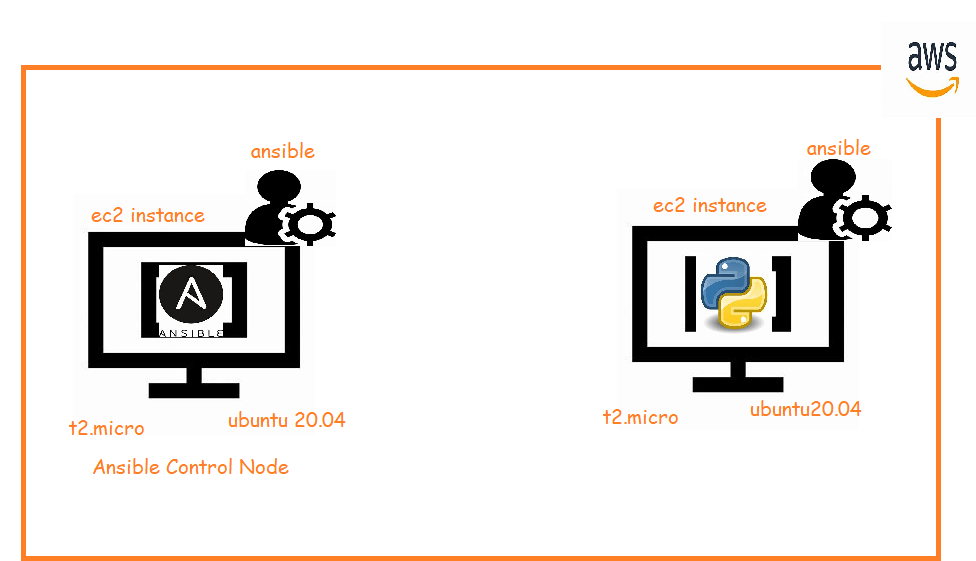


* + So now we have a sudo user ansible, so we need to configure in Ansible control node so that when ansible connects to node1, it connects as ansible user 

OCTOBER 7, 2021

# DevOps Classroom Series – 07/Oct/2021

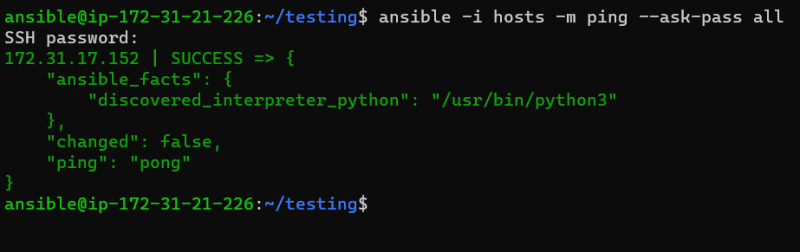
## Ansible Configuration

* The configuration done so far is represented as shown below 
* Let’s create an inventory file with the Node1 details. Create a file called as hosts with the following content

172.31.17.152

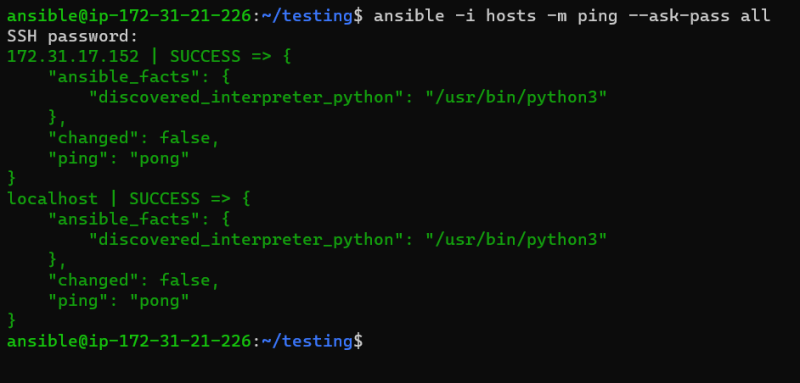
* To check the connectivity for ansible control node with Node which internally checks the connectivity and ssh login ansible has a special command

ansible -i<inventory file path> -m ping --ask-pass all

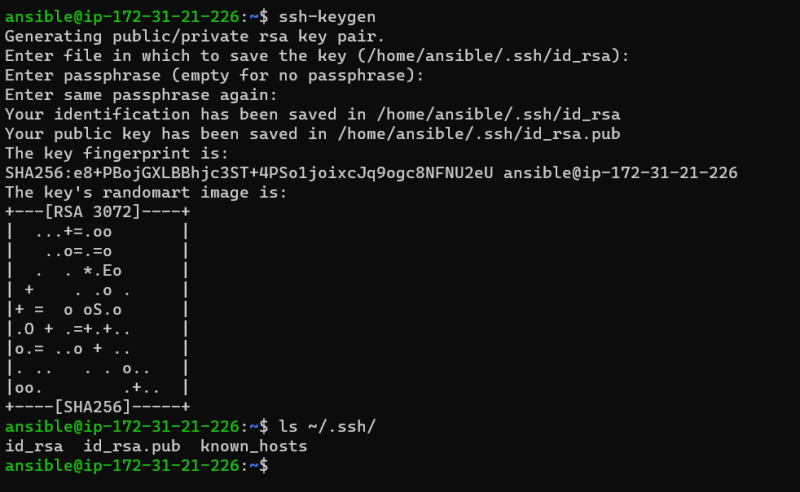
* So lets execute this command on control node 
* Now lets add one more entry into the inventory

172.31.17.152

localhost

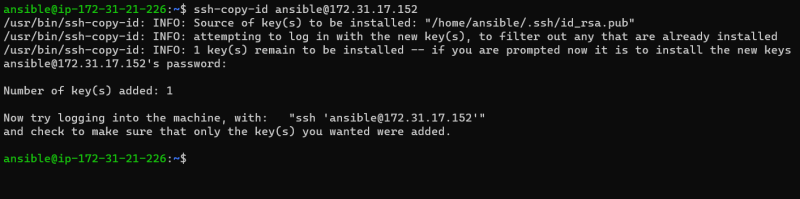


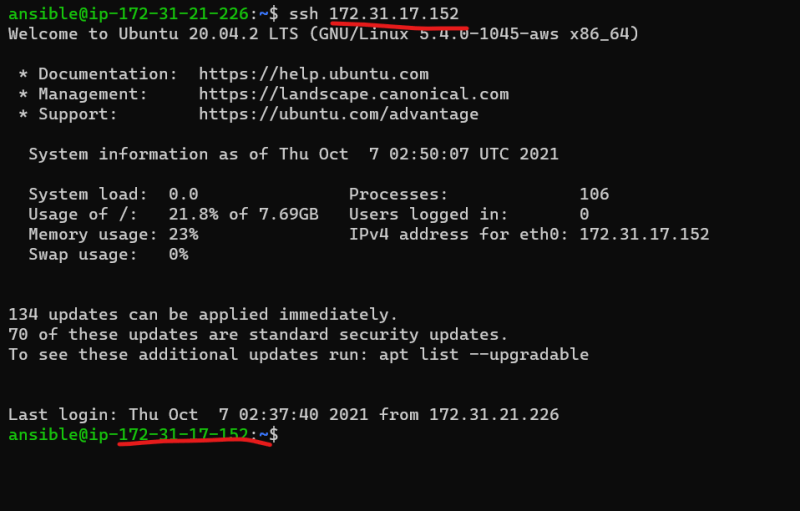
* Now entering password every time might not be good option for automating deployments, so let’s try to use key based authentication
* To Create a key based authentication b/w ansible control node and nodes
  + Create a key pair in Ansible control node
* ssh-keygen



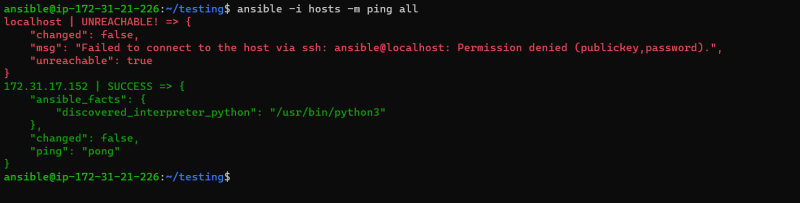
* + Copy the public key to the node(s) from ansible control node

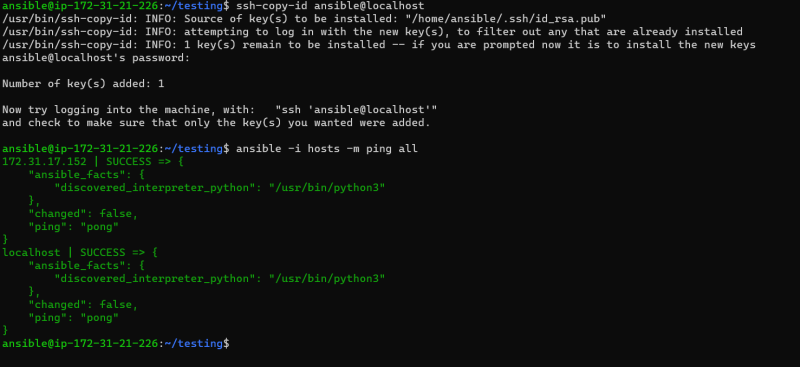
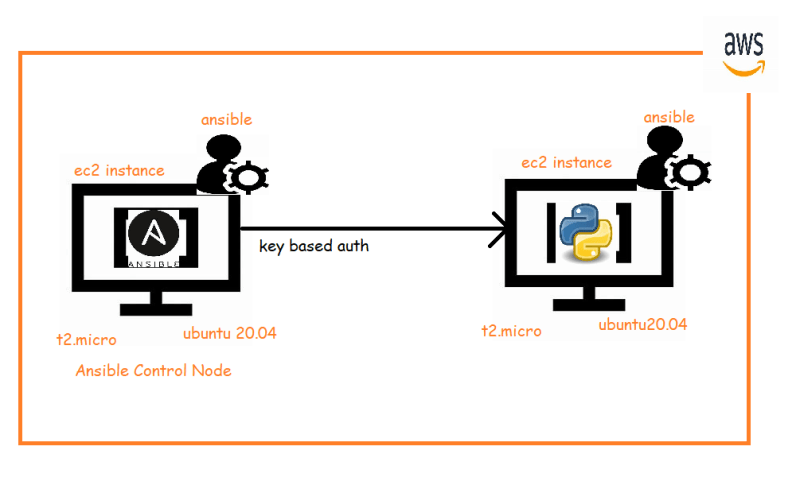
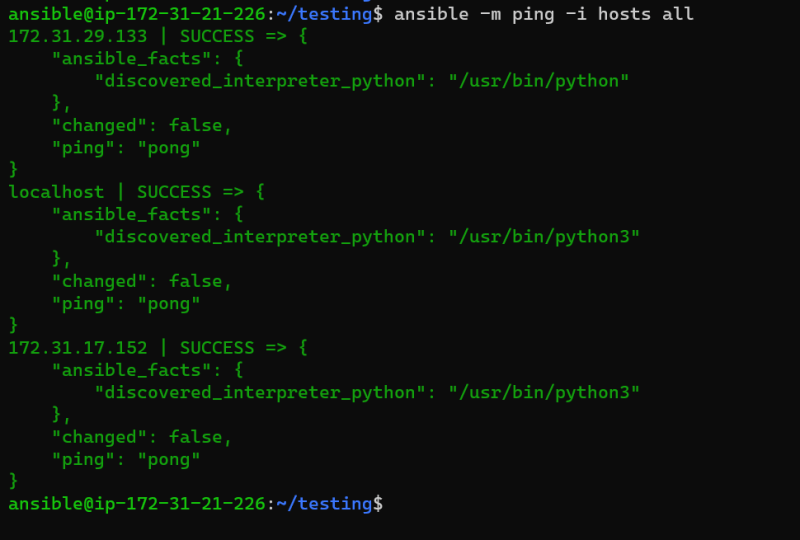
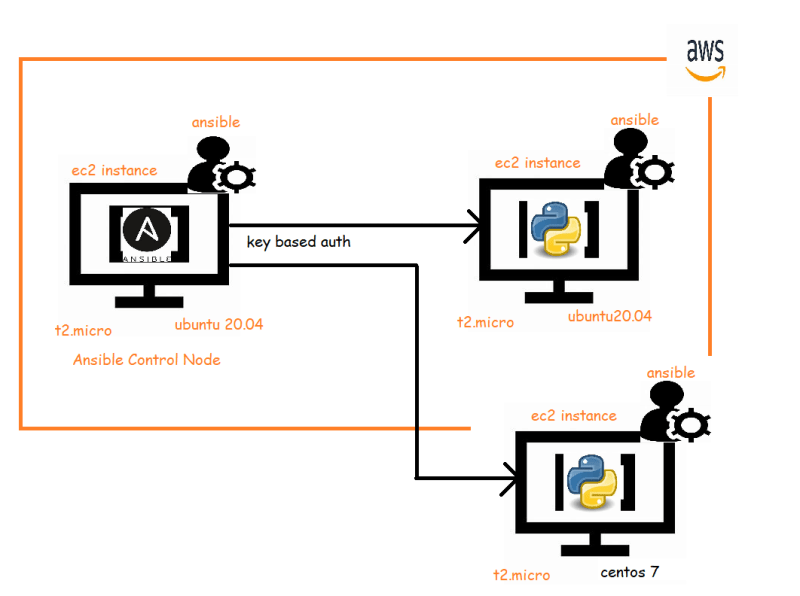
ssh-copy-id<username>@<ipaddress>



* + Now let’s try to login from ansible control node to node just by using ip address 
* Now lets try to ansible connectivity check without ask password

ansible -i hosts -m ping all

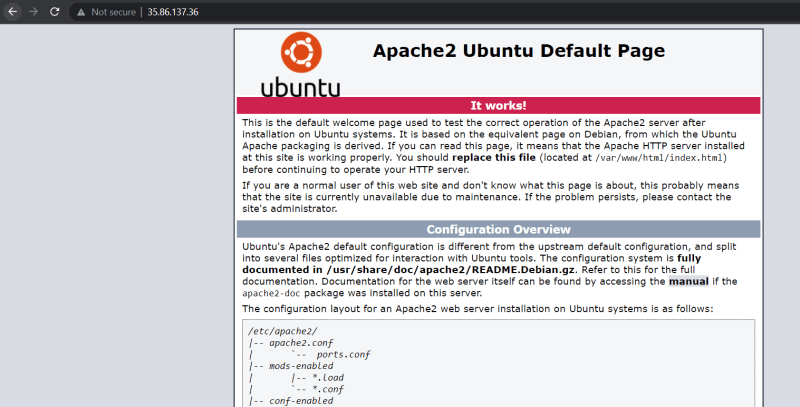
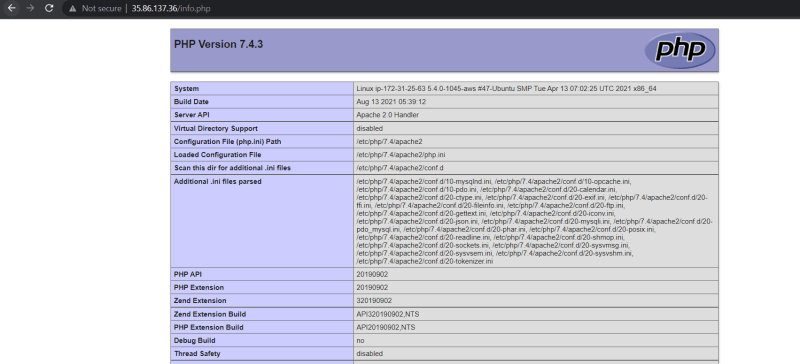


* Now lets try to copy keys to localhost 
* Setup done so far 
* Now let’s add the new centos 7 node and configure it to the ansible control node
  + for steps view recording 
* Configuration overview 
* Exercise:
  + YAML & JSON Tutorial [Refer Here](https://www.youtube.com/watch?v=ggOmHlnhPaM&list=PLuVH8Jaq3mLud3sVDvJ-gJ__0zd15wGDd&index=15)

OCTOBER 8, 2021

# DevOps Classroom Series – 08/Oct/2021

## Approach

* We use ansible in DevOps to automate deploying Applications, So we need to know the manual way of deploying before we automate the deployment.
* Make a note of all the steps involved in deploying an application and try to automate each step using Configuration Management.
* Example 1: Installing Apache and PHP on the Ubuntu server
  + Manual Steps: (Ubuntu 20.04)
    - all steps [Refer Here](https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu-20-04)
  + sudo apt update
  + sudo apt install apache2 -y
  + sudo apt install php libapache2-mod-php php-mysql -y
  + # Create a file in /var/www/html/info.php "<?phpphpinfo(); ?>"
  + #echo "<?phpphpinfo(); ?>" | sudo tee /var/www/html/info.php
  + sudo service apache2 restart
    - Navigate to http://<publicip&gt; and http://<publicip>/info.php  

## YAML

* We would be using YAML to write Ansible Playbooks
* In YAML we define data as collection of name-value pairs
* [Refer Here](https://docs.ansible.com/ansible/latest/reference_appendices/YAMLSyntax.html)

## Playbook

* Playbook represents collection of plays. Each play is collection of tasks. Each task represents a step to be automated (desired state)

- name: <name of your play>

hosts: <where do you want to execute>

become: <For installation do we need to become sudo user>

tasks:

- name: <name of the task>

<module>: <desired state>

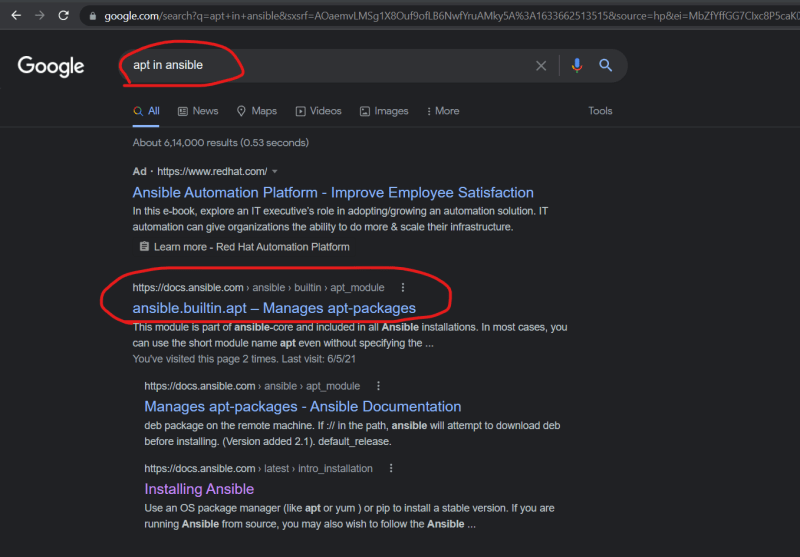
- name: <name of the task>

<module>: <desired state>

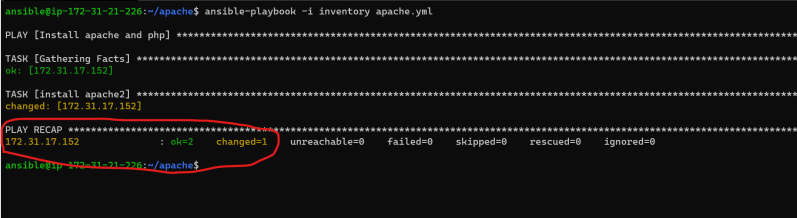
..

- name: <name of the task>

<module>: <desired state>

* Module in Ansible does the actual work (Smallest unit of work), using modules we can describe desired state
* [Refer Here](https://docs.ansible.com/ansible/2.8/modules/list_of_all_modules.html) for the list of modules
* Lets start writing ansible playbook for installing apache and php
* First step involves updating the ubuntu packages using apt. To find the module in apache which helps in automating apt 
* Once we find the module to express our desired we need to use parameters
* So in our case the module is apt [Refer Here](https://docs.ansible.com/ansible/latest/collections/ansible/builtin/apt_module.html) and lets look at parameters
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/beaf471afb42191b1a86e5e29726c48e1bf1c370) for the changes done in the class
* To run the playbook the command is

ansible-playbook -i<path to invetory><path to playbook.yaml>



* When we execute again ansible has dontany thing as apache is already installed 

### Share this:

OCTOBER 9, 2021

# DevOps Classroom Series – 09/Oct/2021

## Ansible Contd

* Manual Steps:

sudo apt update

sudo apt install apache2 -y

sudo apt install php libapache2-mod-php php-mysql -y

# Create a file in /var/www/html/info.php "<?phpphpinfo(); ?>"

echo "<?phpphpinfo(); ?>" | sudo tee /var/www/html/info.php

sudo service apache2 restart

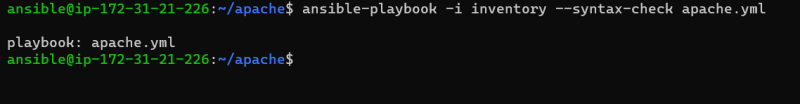
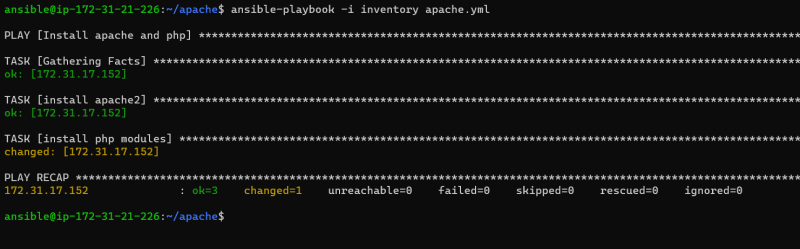
* We have written the play book for the following steps

sudo apt update

sudo apt install apache2 -y

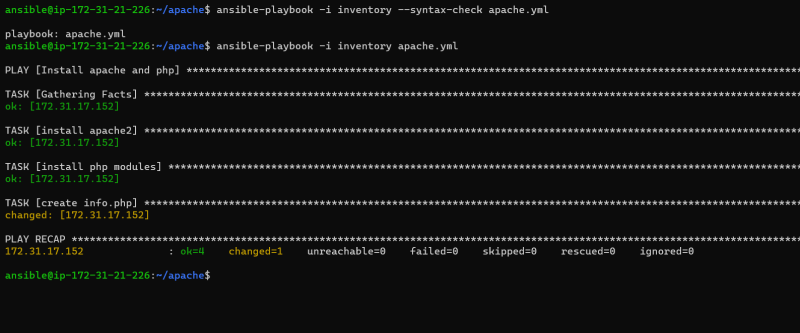
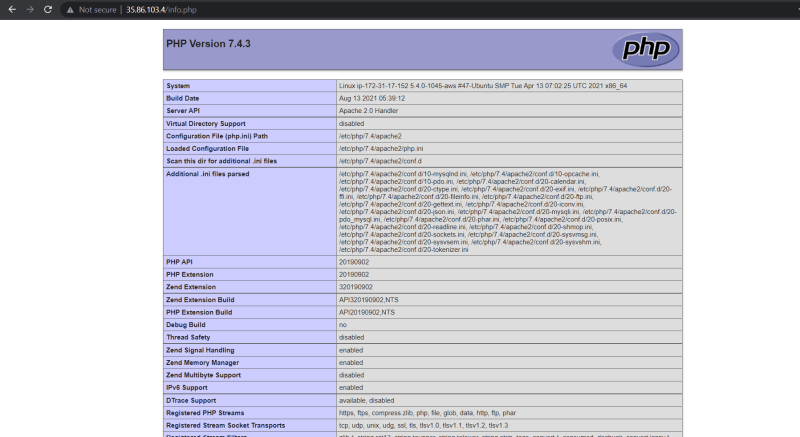
* Now lets try to use the ansible module for this manual step

sudo apt install php libapache2-mod-php php-mysql -y

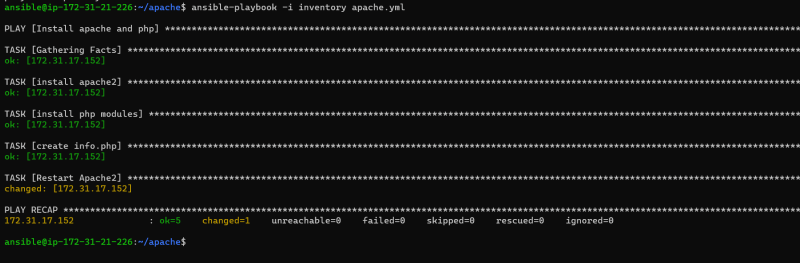
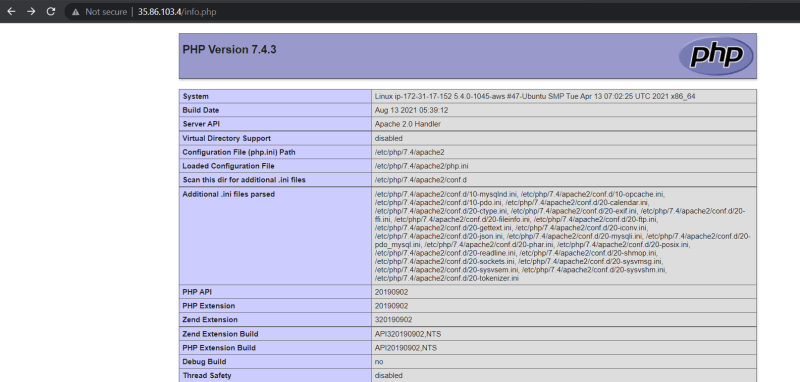
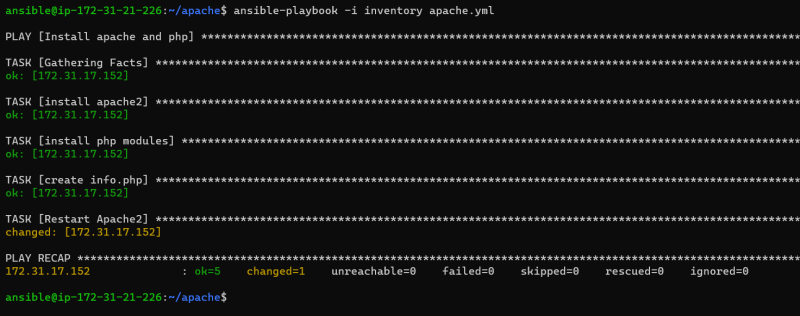
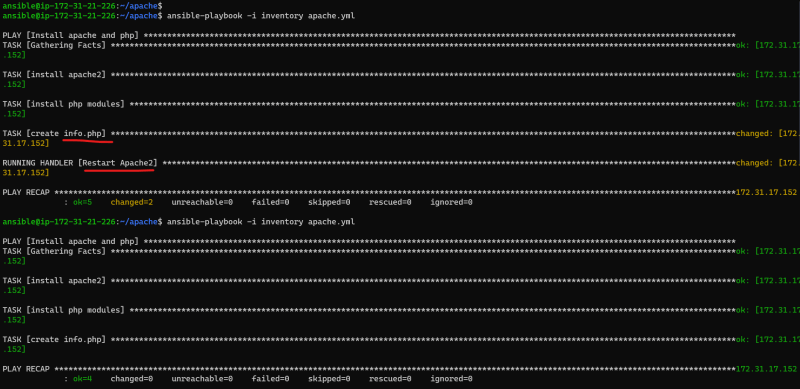
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/90b24cb8b94fea4ec69763f276b69c390e68d9f3) for the changes done
* Let’s do syntax check 
* Now lets apply changes 
* Now we need to create a file using ansible. Manual steps

# Create a file in /var/www/html/info.php "<?phpphpinfo(); ?>"

echo "<?phpphpinfo(); ?>" | sudo tee /var/www/html/info.php

* To create empty files we have file module but to create files with pre-existing content we have either copy the files from ansible control node to nodes or create a file from content using copy module [Refer Here](https://docs.ansible.com/ansible/latest/collections/ansible/builtin/copy_module.html)
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/5a0ce359acab616ef5ce44bff164f5075a8c37de) for the changes  
* Now we need to restart the apache service. Manual steps

sudo service apache2 restart

* [Refer Here](https://docs.ansible.com/ansible/latest/collections/ansible/builtin/service_module.html) for the service module. Lets enable the apache2 service to start automatically during system restarts and ensure the service is restarted after php info page creation.
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/aaba0bd890d7ab6310be819de8133b15dd9442bc) for the changes done  
* Let’s try to rerun the playbook 
* Ideally, we would like to restart apache only when info.phpis created.
  + We need to execute Restart Apache2 Task only when create info.php task is executed.
  + In ansible when we want to execute some tasks as a reaction to other tasks we need to use handlers.
  + [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/2b154459e2285e1aa94a43582c401d238e70925b) for the changeset containing restarting the service as handler 
  + [Refer Here](https://docs.ansible.com/ansible/latest/user_guide/playbooks_handlers.html) for handler documentation
* The playbook which we have written

---

- name: Install apache and php

hosts: all

become: yes

tasks:

- name: install apache2

apt:

name: apache2

update\_cache: yes

state: present

- name: install php modules

apt:

name:

- php

- libapache2-mod-php

- php-mysql

state: present

- name: create info.php

copy:

dest: /var/www/html/info.php

content: |

<?php

phpinfo();

?>

notify:

- Restart Apache2

handlers:

- name: Restart Apache2

service:

name: apache2

enabled: yes

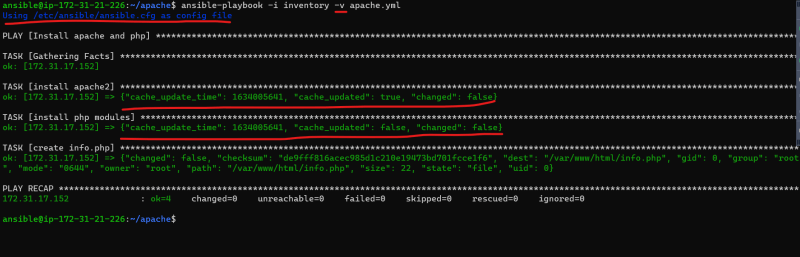
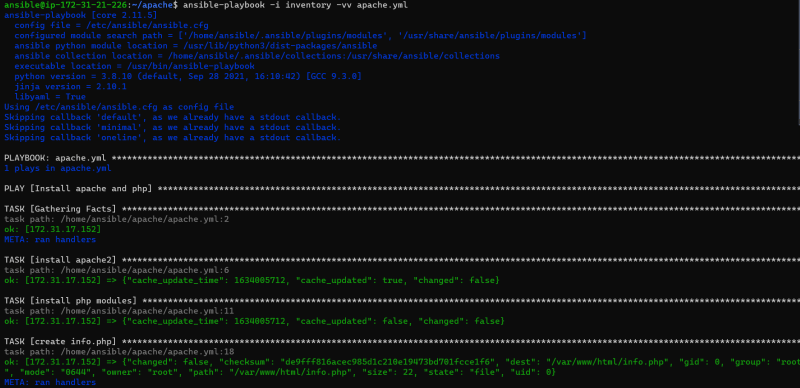
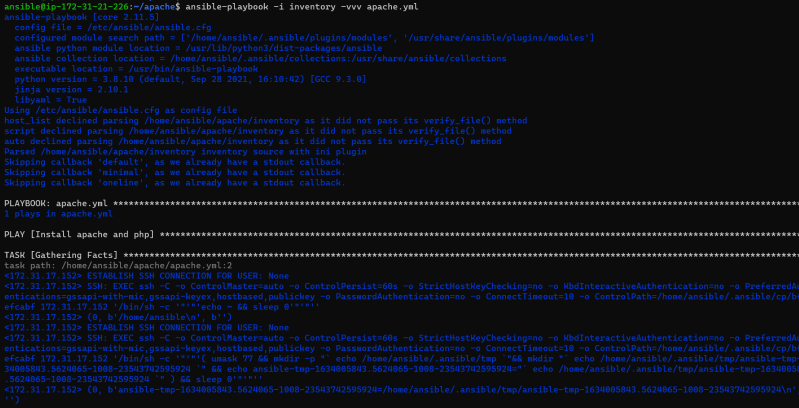
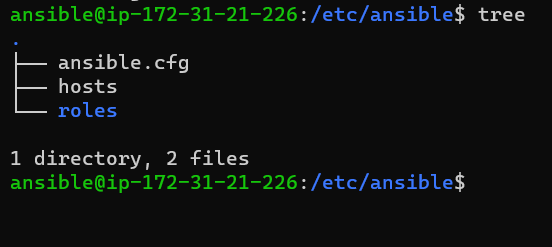
state: restarted

* To be effective in playbook development, we should
  + install softwares/deploy applications manually to know the steps
  + Become good in searching right ansible modules
  + Good with YAML understanding
* Exercise:
  + Try to install the following applications manually on any linux machine
    - Tomcat
    - Nginx
    - mysql
    - Postgres
    - Mongo db

OCTOBER 12, 2021

# DevOps Classroom Series – 12/Oct/2021

## Ansible Continued

* While running the playbooks we can see more logs of whats happening when the playbook executes
  + -v: 
  + -vv: 
  + -vvv: 
  + -vvvv: Connection debugging
* Ansible configurations are store in the directory /etc/ansible in the file ansible.cfg 
* The hosts file in /etc/ansible is the default inventory file.
  + While running playbook’s if you dont provide any inventory then this default inventory is picked up
* The behavior of ansible like default inventory, module search path & many more are picked up from ansible configuration file at /etc/ansible/ansible.cfg

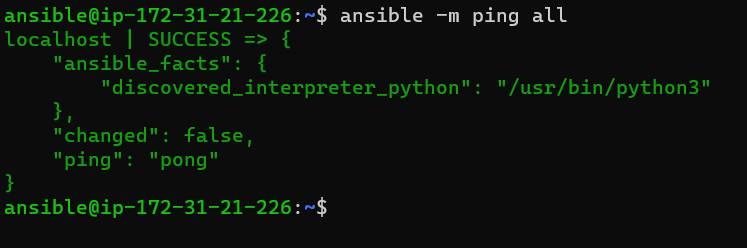
## Adhoc Command

* From ansible we can run modules, This module is expressed as yaml and we use it tasks or handler section which leads to playbook.
* Ansible modules can be executed directly as command line which is referred as adhoc command

ansible -m <name of module> -a <arguments>

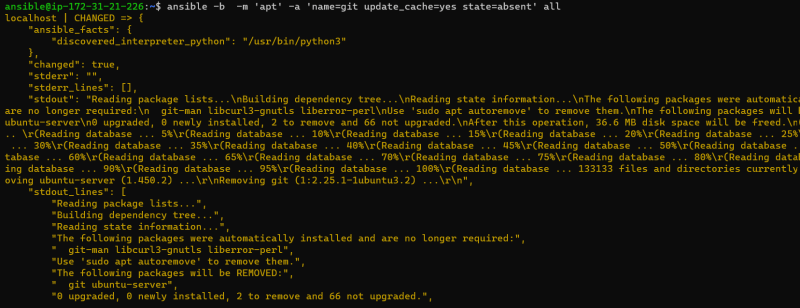
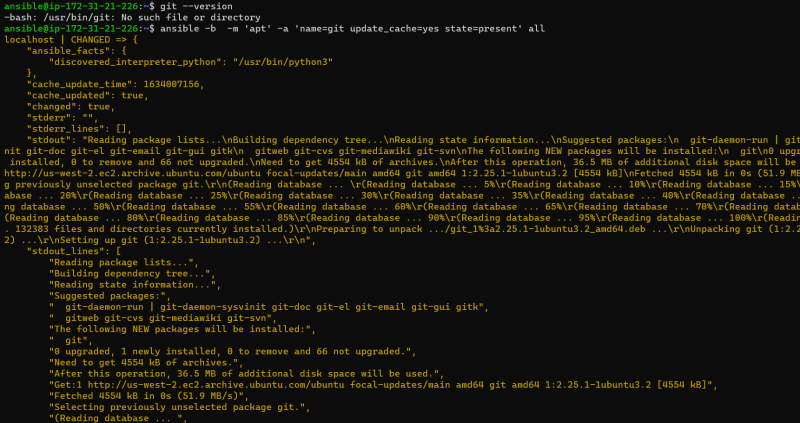
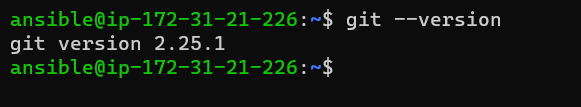
* We have used the ping module to check the connection

ansible -i inventory -m ping all



* Let’s try to install git on the localhost using adhoc command. The manual command is sudo apt update && sudo apt install git -y
* Let’s use the adhoc command documentation [Refer Here](https://docs.ansible.com/ansible/latest/user_guide/intro_adhoc.html)

ansible -b -m 'apt' -a 'name=git update\_cache=yes state=present' all

## Inventory in Ansible

* Generally when we work with deployments we would have number of servers to deal and servers are grouped according to the applications they run for example webservers, db servers, ubuntu server etc..
* In the inventory file we can create groups

[webserver]

172.31.17.152

[dbserver]

172.31.29.133

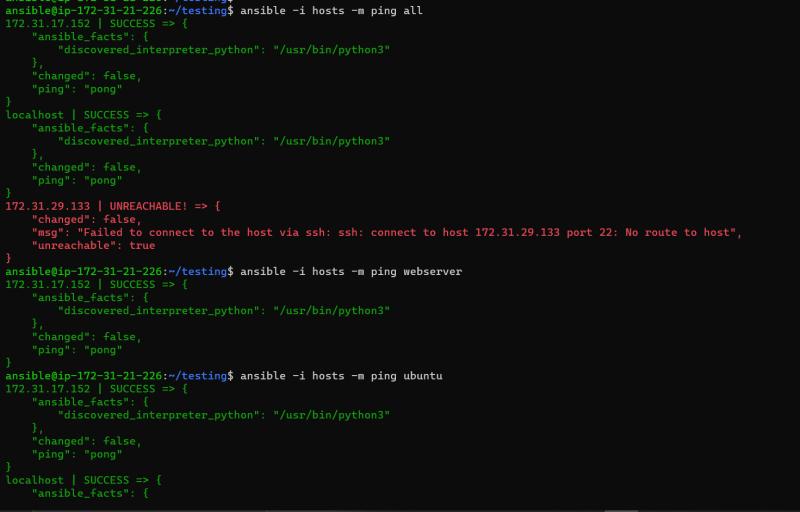
[ubuntu]

172.31.17.152

localhost

[centos]

172.31.29.133

* Then for exploring we have run some ansible ping adhoc commands 
* Ansible has two kinds of inventory
  + Static Inventory
  + Dynamic Inventory
* Next Steps:
  + Different formats for writing inventories
  + Static vs Dynamic Inventory
  + Ansible Variables

### Share this:

OCTOBER 13, 2021

# DevOps Classroom Series – 13/Oct/2021

## Ansible Inventory and Variables Continued

* Let’s start with a use case to install apache and php info on centos 7

sudo yum install httpd -y

sudosystemctl start httpd.service

sudosystemctl enable httpd.service

sudo yum install phpphp-mysqlphp-fpm -y

echo "<?phpphpinfo(); ?>" | sudo tee /var/www/html/info.php

sudosystemctl restart httpd.service

* If i have to write a playbook for this it would be some thing like

---

- name: Install apache and php

hosts: centos

become: yes

tasks:

- name: install apache

yum:

name: httpd

state: present

- name: install php modules

yum:

name:

- php

- php-fpm

- php-mysql

state: present

- name: create info.php

copy:

dest: /var/www/html/info.php

content: |

<?php

phpinfo();

?>

notify:

- Restart Apache

handlers:

- name: Restart Apache

service:

name: httpd

enabled: yes

state: restarted

* Our intention is not to have two different files, have one playbook for apache installation for ubuntu& Centos
* Ideally we should be writing conditional statements

if node is centos

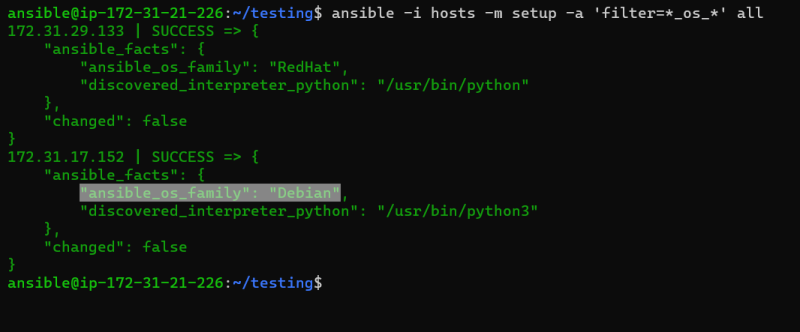
run centos tasks

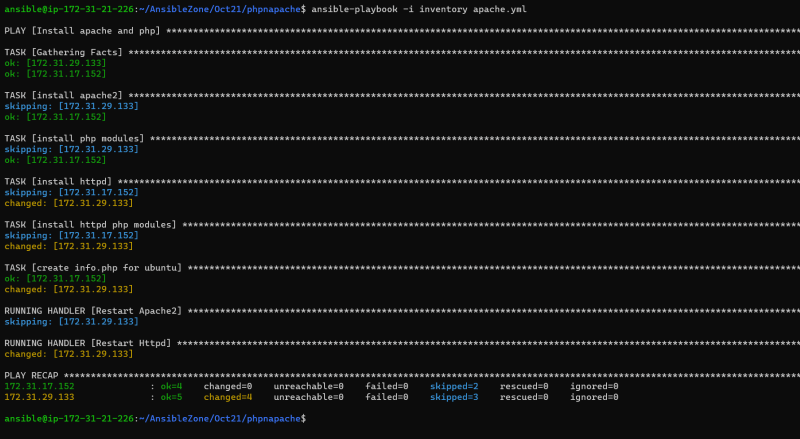
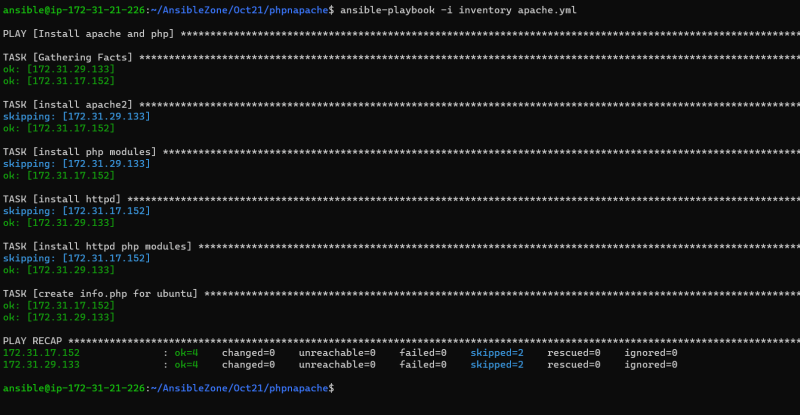
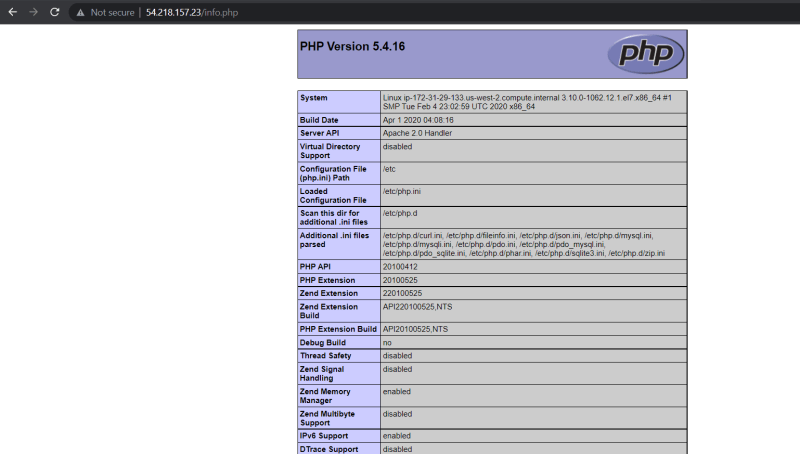
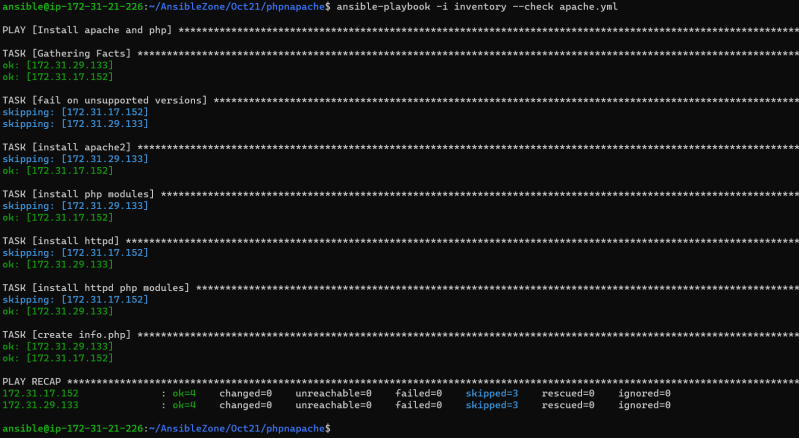
else if node is ubuntu

runubuntu tasks

* We should now find out a way of how ansible identifies whether the node is ubuntu or centos & then write conditionals
* Ansible can collect all the facts(information) about the nodes using the setup module [Refer Here](https://docs.ansible.com/ansible/latest/collections/ansible/builtin/setup_module.html)
* Run the following

ansible -i hosts -m setup all



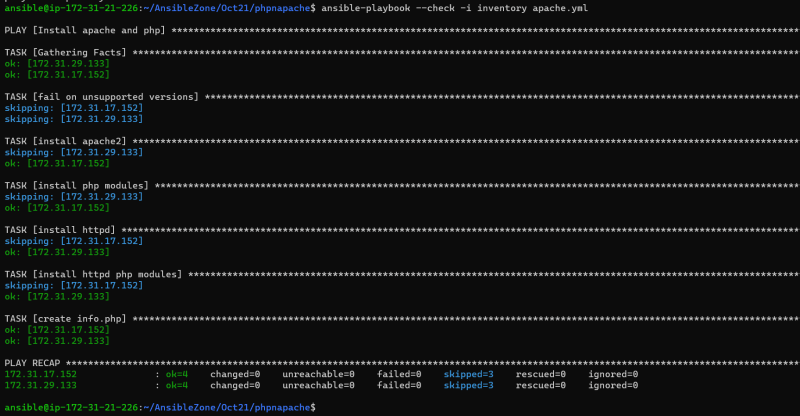
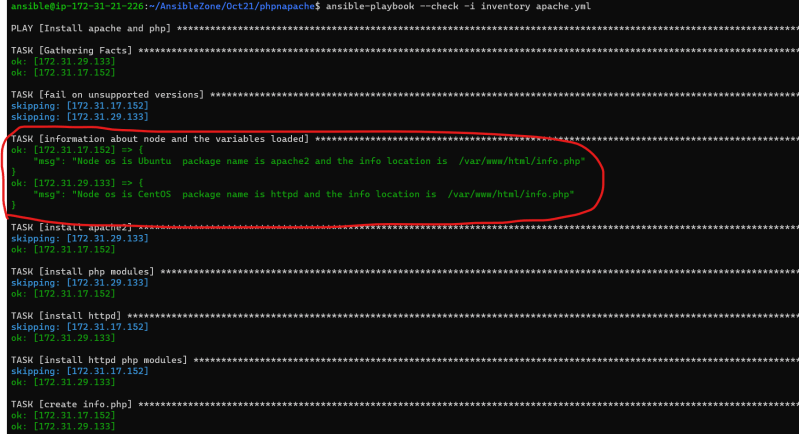
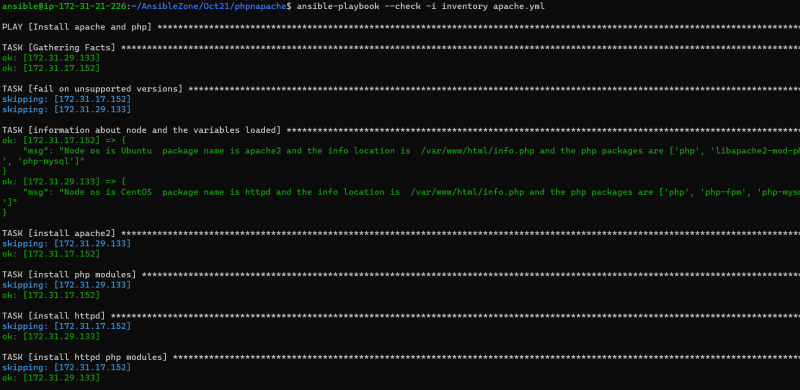
* Now let’s adjust playbooks to use conditionals based on ansible facts [Refer Here](https://docs.ansible.com/ansible/latest/user_guide/playbooks_conditionals.html#conditionals-based-on-ansible-facts)
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/96361c2961df69459f6a2ac273510edad9a18a7e) for the changes made to accommodate multiple distributions
* Let’s try to run the playbook   
* The playbook which we have written is supposed to work on Ubuntu 20 and centos 7, for this lets try to add multiple conditional statements and ensure our playbook fails on unsupported versions with a message to the user [Refer Here](https://docs.ansible.com/ansible/latest/collections/ansible/builtin/fail_module.html) for fail module
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/c0f55a47b8fcc49fb641b065960383d60d89f4b3) for the changes and [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/cbee99829aa1ab25b40e6ac730c1c63c0fb844c9) for the conditionals corrected 

### Share this:

OCTOBER 14, 2021

# DevOps Classroom Series – 14/Oct/2021NOTE BOOK COMPLETED

## Ansible Contd

* Let’s start using variables by defining them in the inventory [Refer Here](https://docs.ansible.com/ansible/latest/user_guide/playbooks_variables.html)
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/b8ce78168d067eeaa82ae016df3696cb02e75030) for the changeset containing group level and host level variables defined in the inventory.
* Corrected the syntax issue by adding quotes [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/3fce5a1ab43a44ffaf70fa564e8673d33ba95915) for the changeset 
* To improve the readability when the playbook executes lets add some messages by using debug module [Refer Here](https://docs.ansible.com/ansible/2.8/modules/debug_module.html#debug-module) 
* Now add variables for php packages [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/3fce5a1ab43a44ffaf70fa564e8673d33ba95915) for the changes 
* Also refer to the changesets with fix and improvements [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/5aae0da18a4295916d7e8ea0e7068c663f6b02d8)
* In Ansible we have a module called as package which when executed on Ubuntu node behaves like apt and when run on red hat nodes behaves like yum [Refer Here](https://docs.ansible.com/ansible/2.8/modules/package_module.html)
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/9e1f854e05bf90842456a9641762c54224b10e09) for the changeset containing changes
* Areas of improvement:
  + Writing inventory files with both server details and variables will be unmaintebled as servers grow, so we need to have different locations for variables and inventories.
  + PHP Info page is still hardcoded in playbook
* Exercise: Write an Ansible Playbook to deploy the following application [Refer Here](https://docs.nopcommerce.com/en/installation-and-upgrading/installing-nopcommerce/installing-on-linux.html). You can skip the database part.

### Share this:

OCTOBER 17, 2021

# DevOps Classroom Series – 16/Oct/2021

## Defining Variables

* As we have seen variables can be defined at host level or group level in the inventory, We can also define variables in the playbook
* To understand some of the concepts of ansible we will be doing the installation of the tomcat server. [Refer Here](https://linuxize.com/post/how-to-install-tomcat-9-on-ubuntu-20-04/) for the manual steps

sudo apt update

sudo apt install openjdk-11-jdk -y

sudouseradd -m -U -d /opt/tomcat -s /bin/false tomcat

cd /tmp

wget https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.54/bin/apache-tomcat-9.0.54.tar.gz

sudo tar -xf /tmp/apache-tomcat-9.0.54.tar.gz -C /opt/tomcat/

sudo ln -s /opt/tomcat/apache-tomcat-9.0.54 /opt/tomcat/latest

sudochown -R tomcat: /opt/tomcat

sudosh -c 'chmod +x /opt/tomcat/latest/bin/\*.sh'

sudonano /etc/systemd/system/tomcat.service

# Contents

###

[Unit]

Description=Tomcat 9 servlet container

After=network.target

[Service]

Type=forking

User=tomcat

Group=tomcat

Environment="JAVA\_HOME=/usr/lib/jvm/java-11-openjdk-amd64"

Environment="JAVA\_OPTS=-Djava.security.egd=file:///dev/urandom -Djava.awt.headless=true"

Environment="CATALINA\_BASE=/opt/tomcat/latest"

Environment="CATALINA\_HOME=/opt/tomcat/latest"

Environment="CATALINA\_PID=/opt/tomcat/latest/temp/tomcat.pid"

Environment="CATALINA\_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC"

ExecStart=/opt/tomcat/latest/bin/startup.sh

ExecStop=/opt/tomcat/latest/bin/shutdown.sh

[Install]

WantedBy=multi-user.target

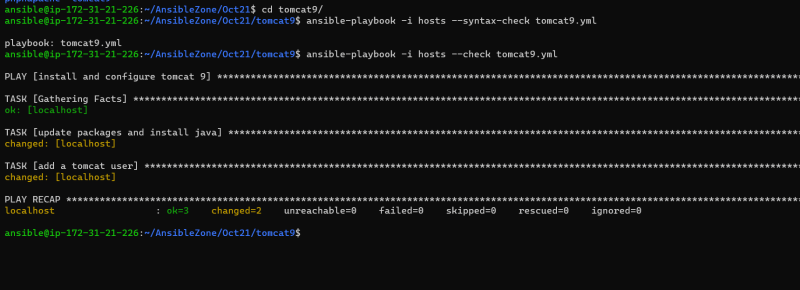
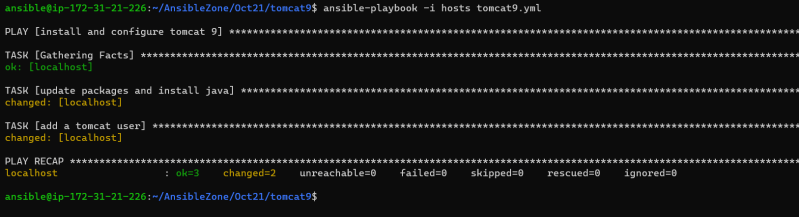
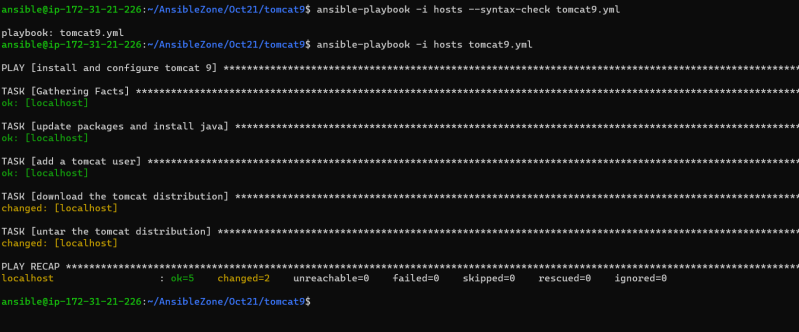
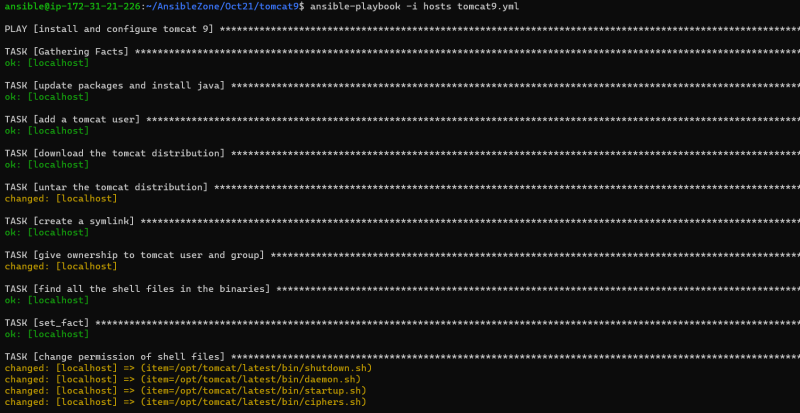
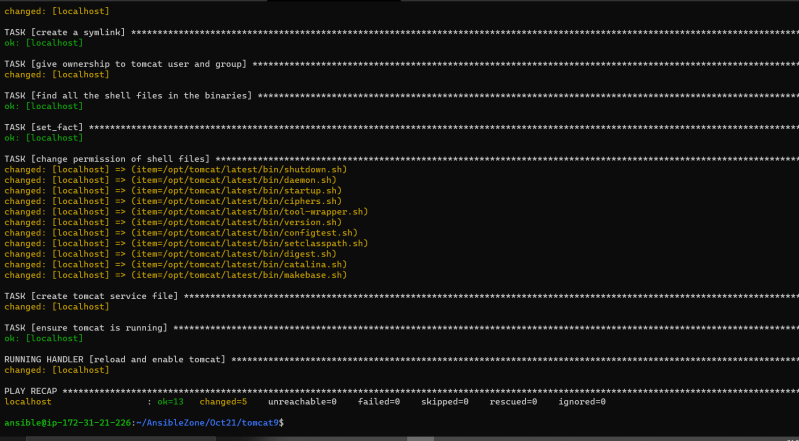
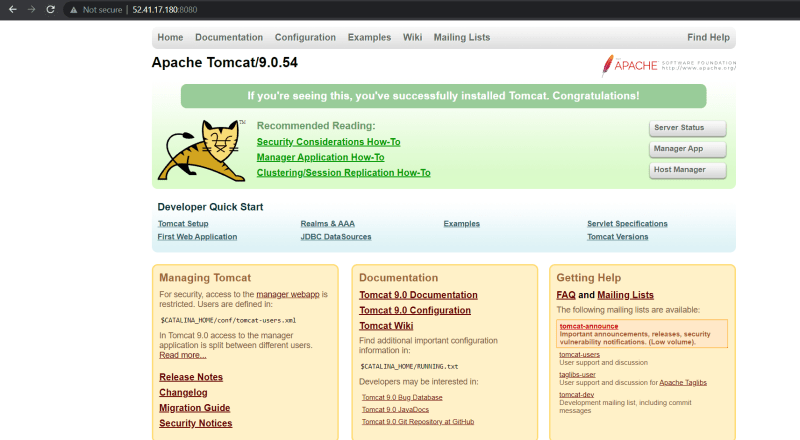
###

sudosystemctl daemon-reload

sudosystemctl enable tomcat.service

sudosystemctl start tomcat.service

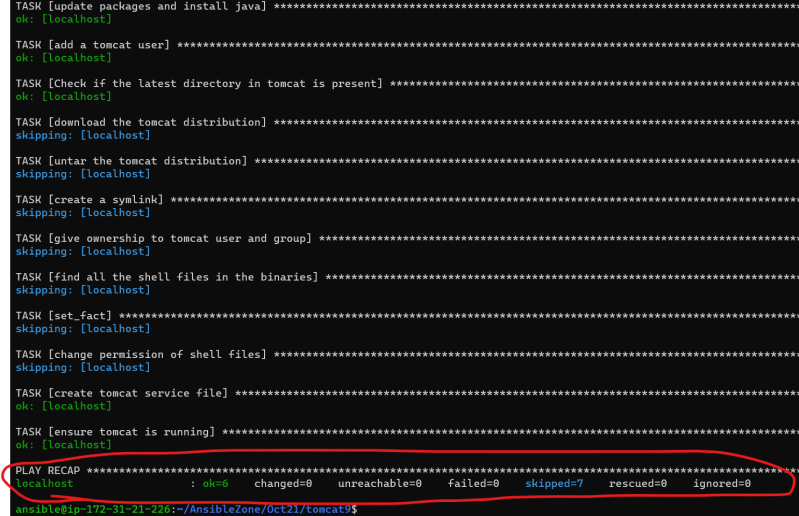
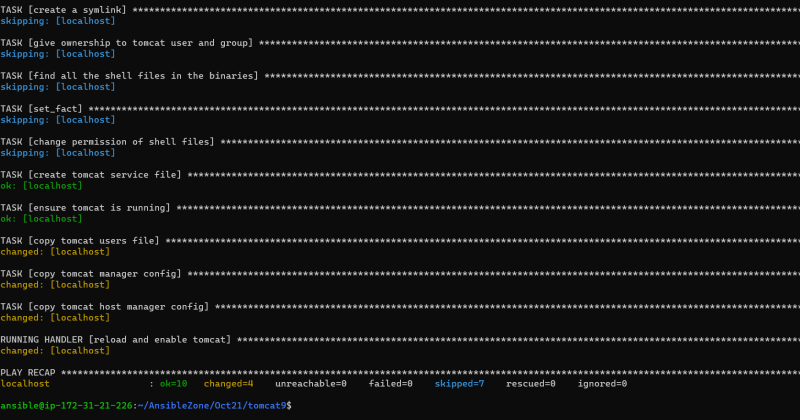
sudosystemctl status tomcat.service

* For rest of steps [Refer Here](https://linuxize.com/post/how-to-install-tomcat-9-on-ubuntu-20-04/#configuring-tomcat-web-management-interface)
* In the current changeset we have installed the java package and created the user with variables defined in the playbook itself [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/a86d17da1650e03b06b9618be02809412220eefe)
* Execute syntax check and dry run 
* Now execute the playbook 
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/cfe2ae5d4ac5a87c99ee4b4579643831924abeb0) for the changeset containing the download and untar 
* We need to give execute permissions to all .sh files in /opt/tomcat/latest/bin. In ansible first lets get all the .sh files in the folder /opt/tomcat/latest/bin using ansible find module [Refer Here](https://docs.ansible.com/ansible/latest/collections/ansible/builtin/find_module.html) and store the response ina variable shfiles
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/6324ba5bb1e773a257caa2d599b00115d475eb20) for the changeset containing the changes to assign permissions to right set of files
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/9e4c6db6919dcde5e4cc0751b3a709eaa8a2febb) for the fix with symlink
* Now execute the playbook 
* Now let’s use the copy module to copy the service file from playbook folder to the remote node [Refer Here](https://docs.ansible.com/ansible/latest/collections/ansible/builtin/copy_module.html)
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/5351058e5e36a2c6591eee37171eceb5f7ea23ec) this changeset for tomcat9 service & daemon reloads
* Corrected few failure [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/6be66e111a069c8f924ab2abe61ad09eefe672d9) this changeset and executed the ansible playbook 
* Now tried to access the tomcat page using browser 
* Exercise:
  + Ensure when we execute ansible playbook to install tomcat on node where tomcat is already running the changed should be zero
  + Try to make changes in this playbook to make it work on centos 7 (8)/RHEL 7 (8)

OCTOBER 17, 2021

# DevOps Classroom Series – 17/Oct/2021

## Ansible Contd

* Rather than defining the variables in the playbook, we have created a file defaults.yml and defined the variables over there. [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/36b731d0bdd85123060ab3532b6efdfa01ab631d) for the changes
* Now to avoid unnecessary executions, we can use stat module [Refer Here](https://docs.ansible.com/ansible/latest/collections/ansible/builtin/stat_module.html).
* Used stat and register to store the value and use it in other tasks.
* Now execute the command 
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/544bd2ba2d3565d605017fed371abdbe756210d2) for the changes to contain the files to be copied 
* Now we have written a playbook which works on ubuntu servers to install tomcat9 and jdk 11.
* Lets try to create a better folder structure for our playbook [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/2cf631976e4616218481b3de5f7b187d75344f9b) for the changeset 
* Now lets extend our playbook to work on centos 7 [Refer Here](https://linuxize.com/post/how-to-install-tomcat-9-on-centos-7/)
* To make our playbook to work on centos7 we need to just change the way we install java
* [Refer Here](https://github.com/asquarezone/AnsibleZone) for the changeset containing changes to run on ubuntu and centos and also has host and group variables
* [Refer Here](https://github.com/asquarezone/AnsibleZone/tree/master/Oct21/tomcat9) for the playbook.
* Note: This playbook is better than the previous apache playbook but it still needs imporvement
  + Copying static files leads to problems, if there is way where we can copy the files with some dynamic content

## Hint for nop commerce installation

* The playbook for automating first few steps are as shown below

---

- name: install nop commerce

become: yes

hosts: all

tasks:

- name: download the debian file

get\_url:

url: https://packages.microsoft.com/config/ubuntu/20.04/packages-microsoft-prod.deb

dest: /tmp/packages-microsoft-prod.deb

- name: install the debian package

apt:

deb: /tmp/packages-microsoft-prod.deb

- name: install .net core runtimes

apt:

name: "{{ item }}"

update\_cache: yes

state: present

with\_items:

- apt-transport-https

- aspnetcore-runtime-3.1

-

* [Refer Here](https://docs.nopcommerce.com/en/installation-and-upgrading/installing-nopcommerce/installing-on-linux.html) for all the steps.

OCTOBER 21, 2021

# DevOps Classroom Series – 21/Oct/2021

## Exercise – Deploying a Spring boot Application

* This activity is about deploying a spring boot application
* The application can be downloaded from [Refer Here](https://referenceapplicationskhaja.s3.us-west-2.amazonaws.com/spring-petclinic-2.4.2.jar)
* Steps of configuring the machine
  + Install Open jdk 11
  + Download the jar file to some folder on the node /usr/share/spc/spring-petclinic.jar
  + Create a user
* sudo useraddspc
* sudo password spc
* sudo chownspc:spc /usr/share/spc/spring-petclinic.jar
* sudo chmod 500 /usr/share/spc/spring-petclinic.jar
  + Create a service file /etc/systemd/system/spc.service
  + [Unit]
  + Description=Spring petclinic application
  + After=network.target
  + [Service]
  + User=spc
  + ExecStart=/usr/bin/java -jar /usr/share/spc/spring-petclinic.jar SuccessExitStatus=143
  + [Install]
  + WantedBy=multi-user.target
  + Application should be accessible on [http://publicip:8080](http://publicip:8080/)
* Exercise:
  + Ensure the manual steps are working
  + Write a playbook to make this spring petclinic application deploy on ubuntu and then centos
  + hint:
    - java installation:
      * ubuntu: sudo apt update && sudo apt install openjdk-11-jdk -y
      * centos: sudo yum install java-11-openjdk-devel –y

OCTOBER 22, 2021

# DevOps Classroom Series – 22/Oct/2021

## Templating using Jinja2

* Ansible uses Jinja2 templating to enable dynamic expressions and access to variables.
* First and foremost advantage of using Jinja2 is we can avoid static file copies.
* Couple of classes ago, we had problem with tomcat.service as the path of the java was different for centos and ubuntu systems. To solve this we created two service files and two tasks
* We can avoid this by using templates
* Lets create a new folder for templates and create a tomcat.service.j2 file
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/af36f994fa9b0415e5a87c9aa4bfb65bb1fb545c) for the changeset containing the usage of jinja2 templates
* [Refer Here](https://docs.ansible.com/ansible/latest/user_guide/playbooks_templating.html) for the official docs
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/e8ccc776036d9213b5680e4555a4babf41c6bb08) for some example jinja2 filters
* Using map filter and list filter

- hosts: localhost

connection: local

gather\_facts: no

vars:

names:

- first: Paul

last: Thompson

- first: Rod

last: Johnson

tasks:

- debug:

msg={{ names | map(attribute='first') | list }}

- debug:

msg={{ names | map(attribute='last') | list }}

- debug:

msg={{ names | map('upper') | list }}

- debug:

msg={{ names | map(attribute='last') | map('upper') | list }}

## Ansible Behavioral Parameters

* The list of behavior parameters

| **Name** | **Default** | **Description** |
| --- | --- | --- |
| ansible\_host | Name of the host | Hostname or IP address to SSH |
| ansible\_port | 22 | port to ssh to |
| ansible\_user | root | User to SSH as |
| ansible\_password | (None) | Password to use for SSH Authentication |
| ansible\_ssh\_private\_key\_file | (None) | SSH Private key to use for SSH authentication |
| ansible\_python\_interpreter | /usr/bin/python | Python interpreter on the host |
| ansible\_connection | smart | How Ansible will connect to host |

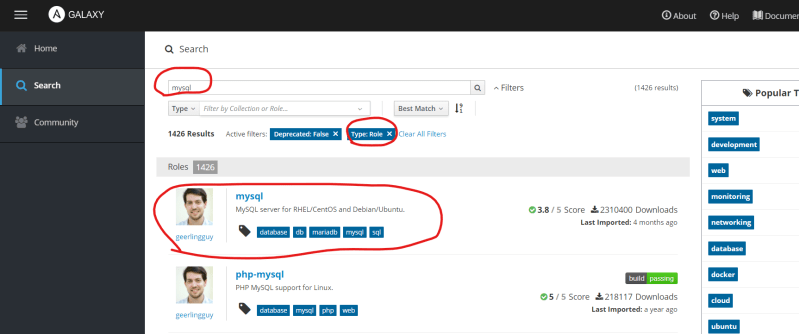
OCTOBER 24, 2021

# DevOps Classroom Series – 23/Oct/2021

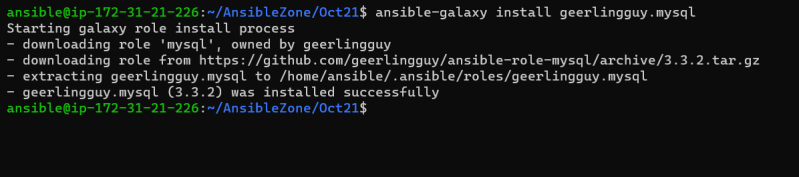
## What has to be done if we donot find the right module

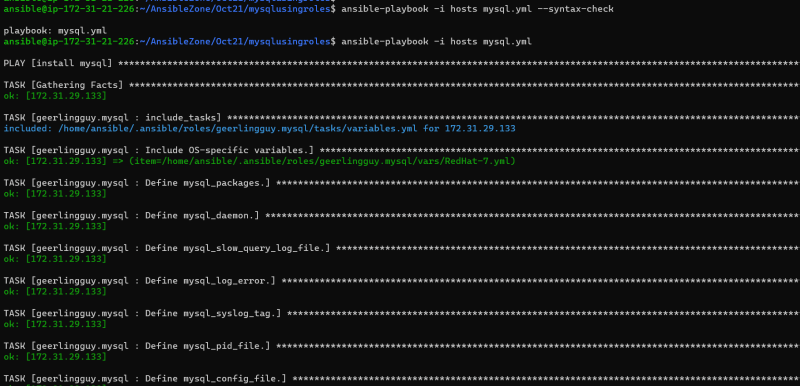
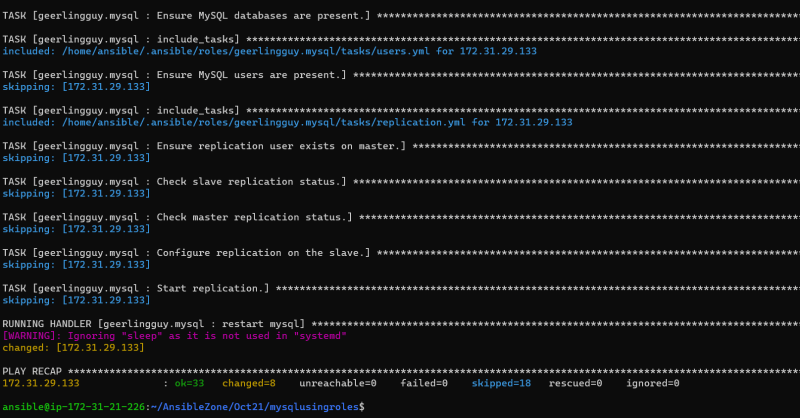
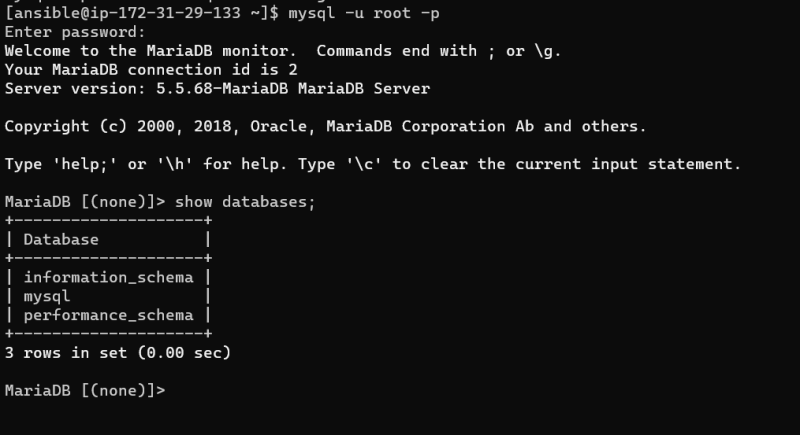
* If we don’t find a module, we have two options
  + Create your own custom module:
    - To do this we need to know how to code in python [Refer Here](https://docs.ansible.com/ansible/latest/dev_guide/developing_modules_general.html)
  + Directly execute the linux/windows command
    - If you are automating any linux/unix/mac we can use shell module [Refer Here](https://docs.ansible.com/ansible/2.8/modules/shell_module.html#shell-module)
    - If you are automating on windows we can use win\_shell module [Refer Here](https://docs.ansible.com/ansible/2.8/modules/win_shell_module.html#win-shell-module)
    - These modules are not idempotent i.e. whenever the playbook is executed the task execute all the time, so we need to come up with idempotencei.e using some kind of when condition

## Ansible Galaxy

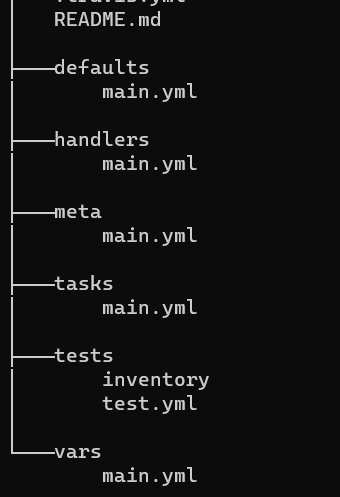
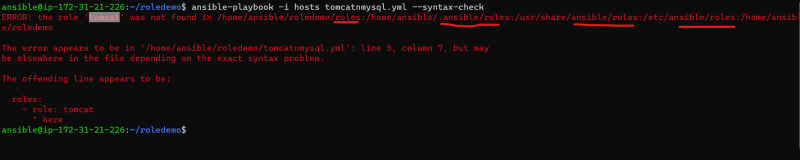
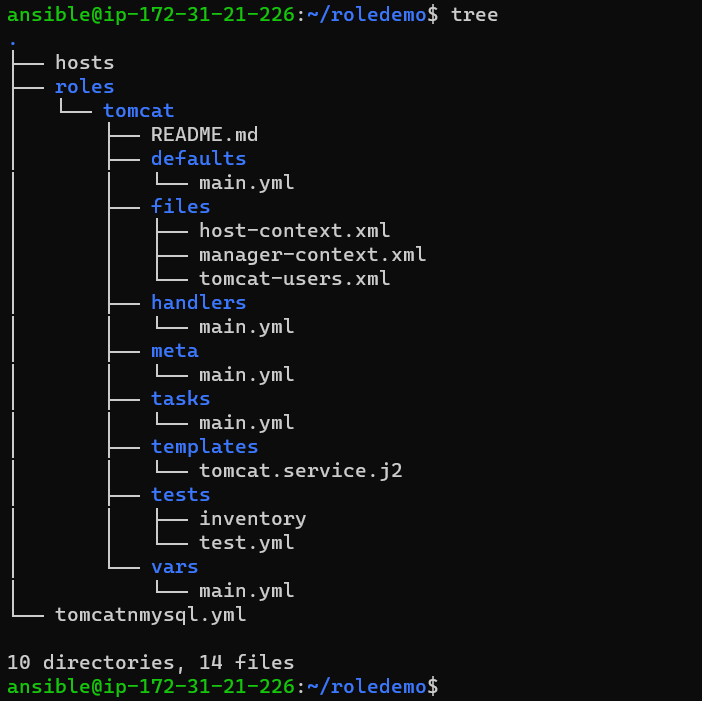
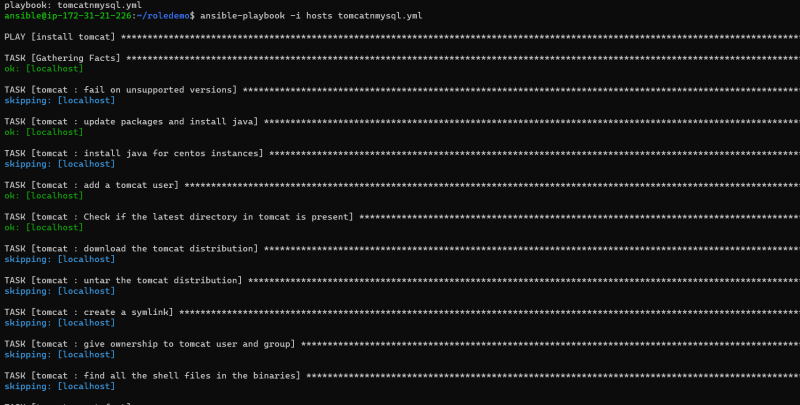
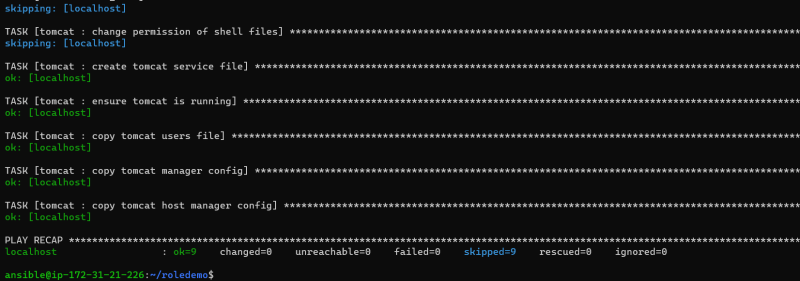
* This hosts the work done by the community to deploy applications, configure softwares etc..
* We can use this work in our playbooks.
* There are two possible ways of sharing work in Ansible Galaxy [Refer Here](https://galaxy.ansible.com/)
  + Roles
  + Collections
* Activity: Lets try to find the role to install mysql in node 2 
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/765315180e30b50bf25cdc2282c3996e3f371869) for the changes done
* Log in to the ansible control node and install role

ansible-galaxy install geerlingguy.mysql



* Now let’s try to run our playbook   
* Using community roles makes our deployments simple & in many case we are expected to create roles for the application deployments so that they can be reused.
* Note: Playbooks also can be reused by using import playbooks [Refer Here](https://docs.ansible.com/ansible/latest/collections/ansible/builtin/import_playbook_module.html)

## Activity 2: Let’s try to create a role from the tomcat playbook which we have created

* In Ansible, role is the primary mechanism for breaking a playbook into multiple files to simplify writing playbooks & making them easier to reuse
* Basic structure of ansible role [Refer Here](https://github.com/asquarezone/AnsibleZone/tree/master/Oct21/role-structure/sample) 
* [Refer Here](https://docs.ansible.com/ansible/latest/user_guide/playbooks_reuse_roles.html#role-directory-structure) for the official docs of the role directory structure
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/6bb500b7f1e5219e74f3695858bd3f92c0cb51f6) this changeset for tomcat role
* Ansible searches for the role in the following folder and if it doesn’t find it throws error
  + current dir of playbook/roles
  + ~/.ansible/roles
  + /usr/share/ansible/roles
  + /etc/ansible/roles 
* Lets copy tomcat to localdir/roles 
* Now execute the role  
* [Refer Here](https://github.com/asquarezone/AnsibleZone/commit/06a90a545bc95ae2cfea4e2c194c7a2f14cf7067) for the changeset containing os specific vars
* The inventory used is

localhost

centosansible\_host=172.31.29.133

* The playbook for install tomcat using role

---

- name: install tomcat

hosts: all

roles:

- role: '/home/ansible/AnsibleZone/Oct21/myroles/tomcat'

become: yes

## Include vs Import

* The import\_\* and include\_\*: Ansible has two modes of operation for reusable content with including and importing .
* The main difference is
  + All import\_\* statements are preprocessed at the time playbooks are parsed
  + All include\_\* statements are process as they are encountered during execution
  + import is static and include is dynamic

## Exercise:

* Create a role to deploy the nop commerce and spring petclinic as service

## Tags

* In Ansible we can add tags to a single task or include statement
* Lets look at the following playbook

---

- name: learning tags

become: yes

hosts: all

tasks:

- name: install git

package:

name: git

state: present

tags:

- essential

- server

- name: install tree

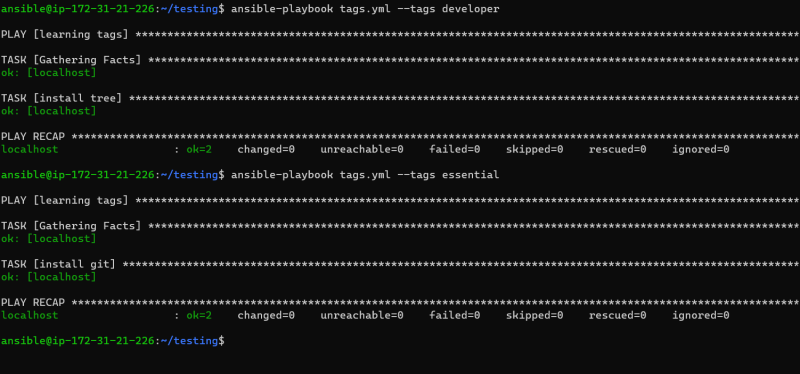
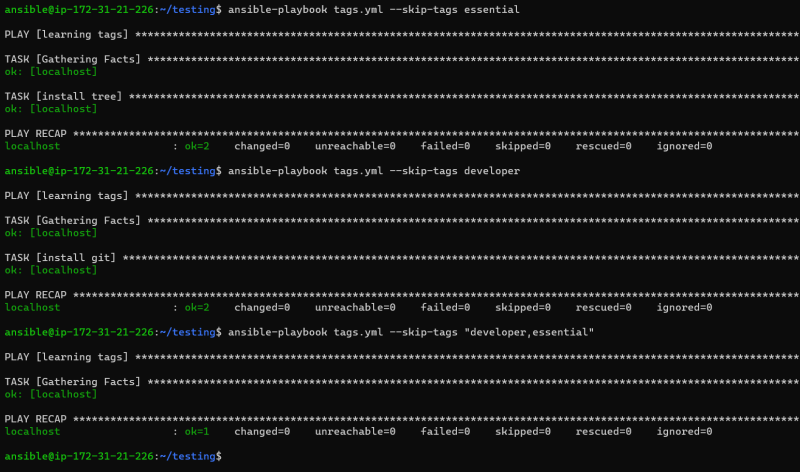
package:

name: tree

state: present

tags:

- developer

* Lets play with tags 
* We can also skip tags 

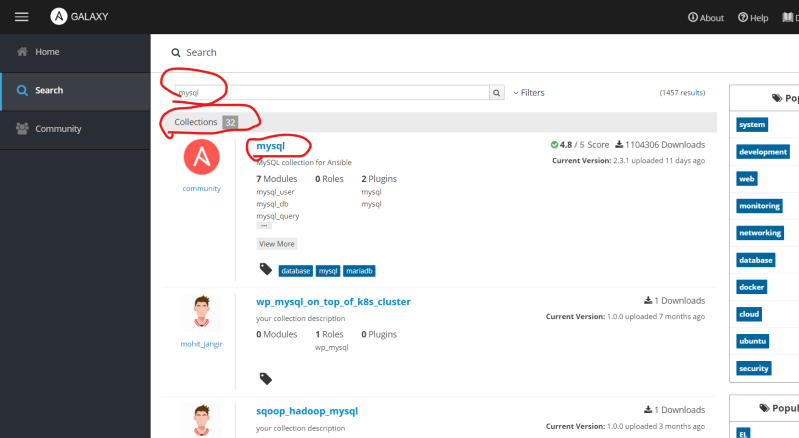
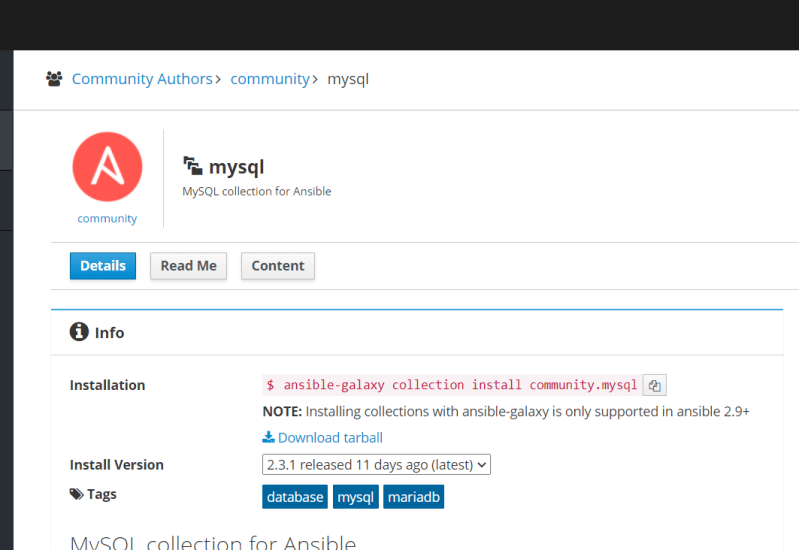
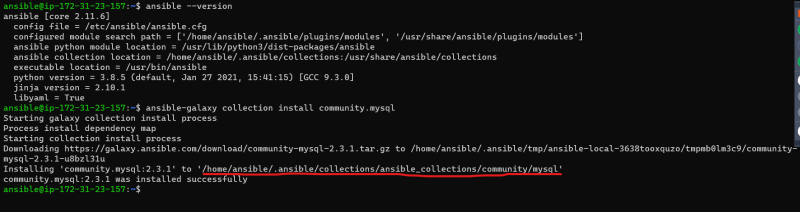
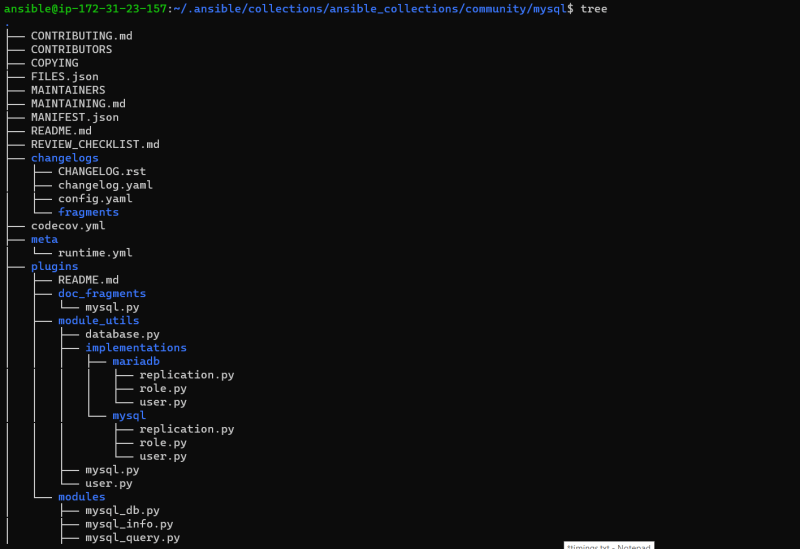
## Topics left

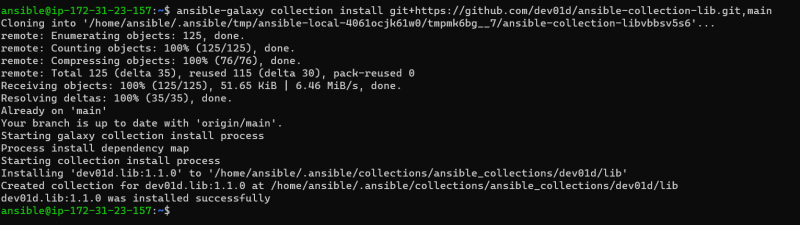
* Ansible Vault
* Ansible Collections
* Ansible Parallelism
* Ansible Dynamic Inventory
* Ansible Tower
* Ansible on Windows

OCTOBER 31, 2021

# DevOps Classroom Series – 31/Oct/2021

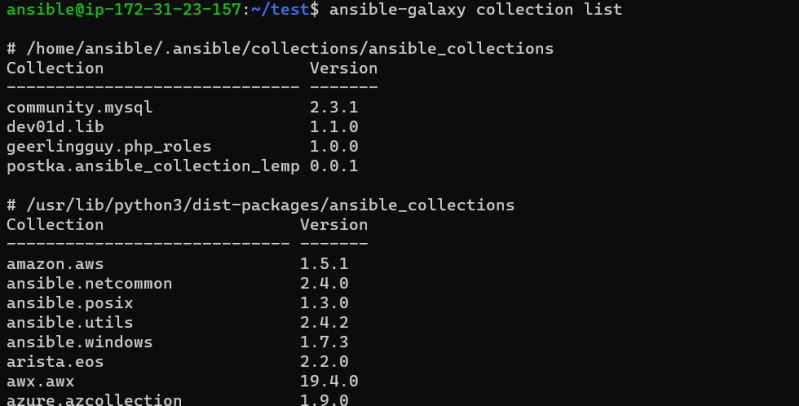
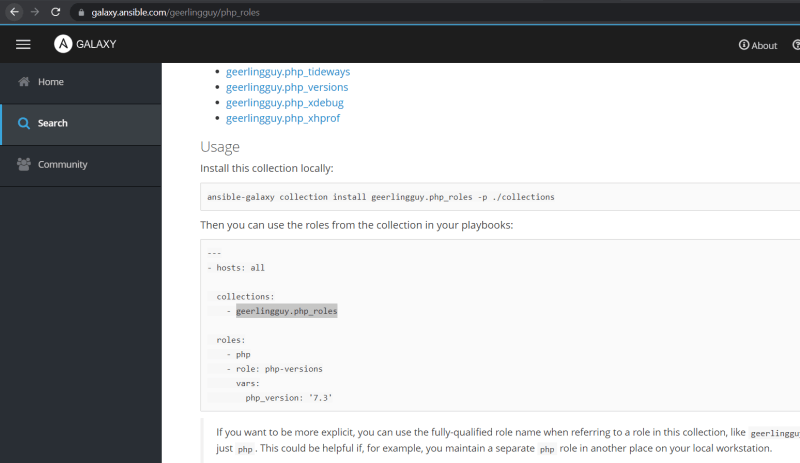
## Ansible Collections

* Ansible collections are distribution format for Ansible content including
  + playbooks
  + modules
  + roles
  + plugins
* Ansible Collections Can be installed from multiple sources
  + ansible-galaxy:
    - Lets install collections mysql from ansible galaxy    
  + git repository:
    - Ansible collections can be installed from git repository
  + ansible-galaxy collection install git+<url>,<branch>



* + requirements.yml file:
    - If you need to download multiple collections from one command, we can build a requirements.yml file
    - In this file we can specify roles as well as collections
  + ---
  + roles:
  + - name: geerlingguy.mysql
  + version: 3.3.2
  + collections:
  + - name: geerlingguy.php\_roles
  + version: 1.0.0
  + source: https://galaxy.ansible.com/
  + - name: https://github.com/Postka/ansible-collection-lemp.git
  + type: git
  + version: master



* Listing Ansible collections 
* Using collections in the playbook 
* Sample yaml file

---

- hosts: all

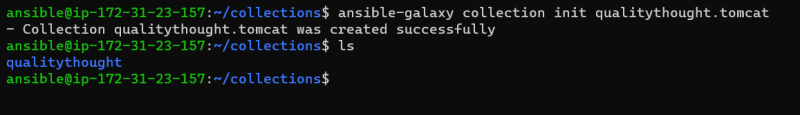
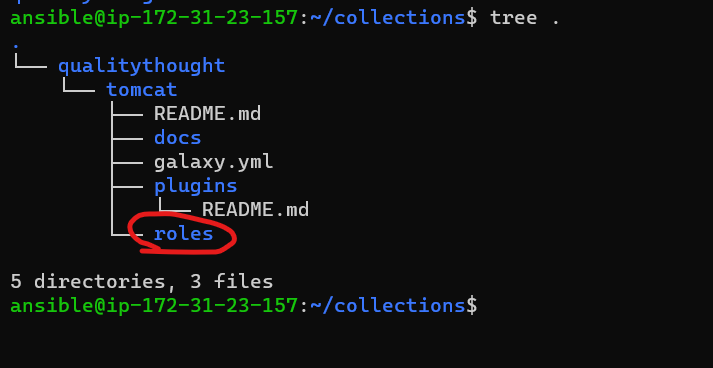
become: yes

collections:

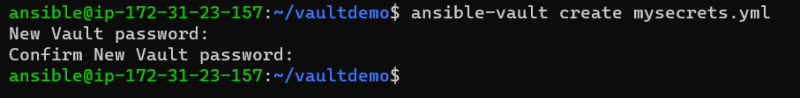
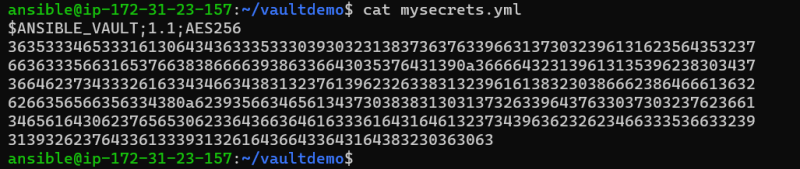
- geerlingguy.php\_roles

roles:

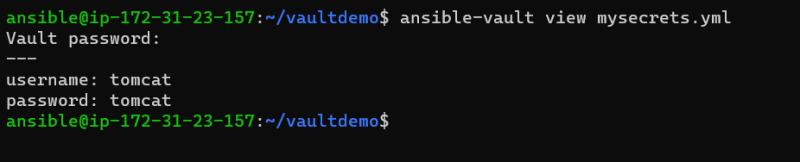
- php

* Creating a collection skeleton:  
* Exercise: Try to copy our tomcat role in the roles folder and publish this whole folder to the new git repository and use from another playbook.

## Ansible Vault

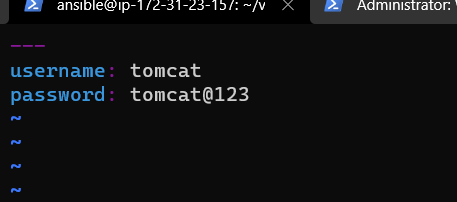
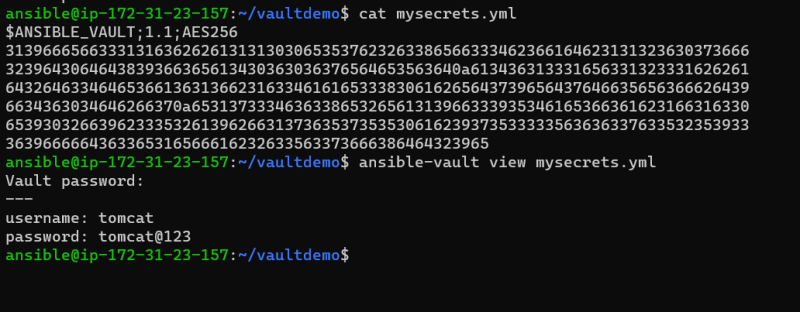
* While using Ansible, you may require to protect some confidential or secret information in playbooks, This may include
  + SSH private and public keys
  + passwords
* Storing sensitive information in plain text is not recommended.
* Ansible provides us with a feature known as Ansible Vault, which helps secure secret information.
* Ansible Vault can encrypt variables, entire files and YAML Playbooks
* Creating an encrypted file in ansible 
* View the encrypted file 
* How to view the encrypted values using ansible-vault

ansible-vault view mysecrets.yml



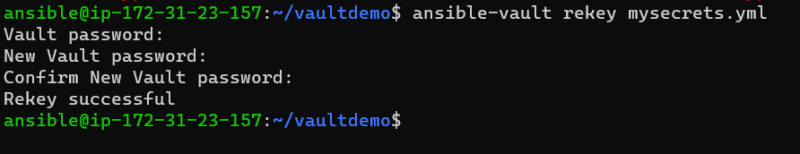
* How to edit an encrypted file in Ansible

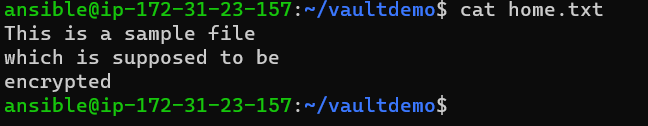
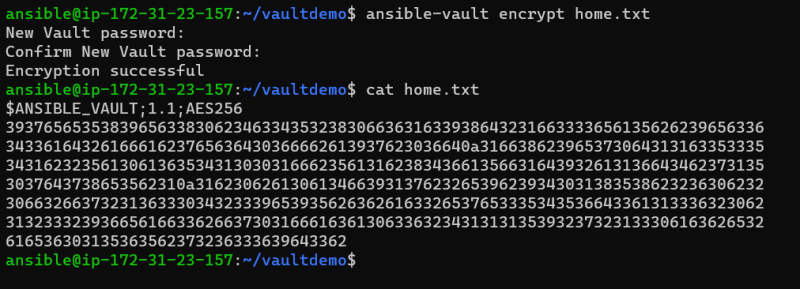
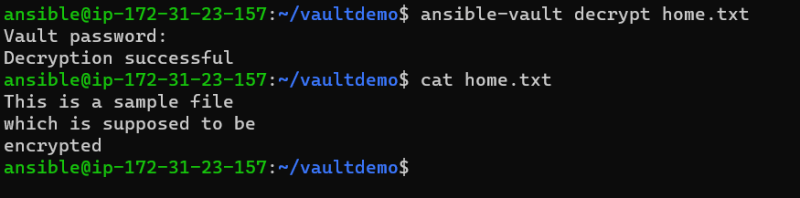
ansible-vault edit mysecrets.yml

* How to change Ansible Vault Password

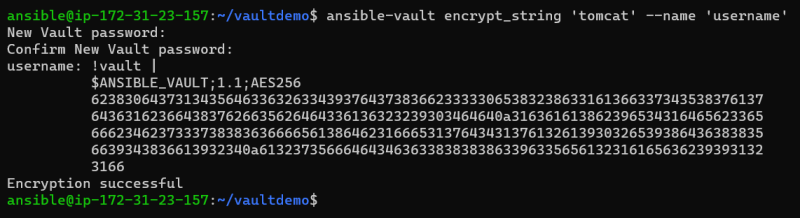
ansible-vault rekey mysecrets.yml



* How to encrypt a file using Ansible Vault  
* How to decrypt a file using Ansible Vault 
* How to encrypt a specific value

ansible-vaultencrypt\_string '<value to be encrypted>' --name '<variable name>'

ansible-vaultencrypt\_string 'tomcat' --name 'username'



* Lets create a playbook deploy.yaml

---

- name: understanding usage of vault

hosts: all

vars:

username: tomcat

tasks:

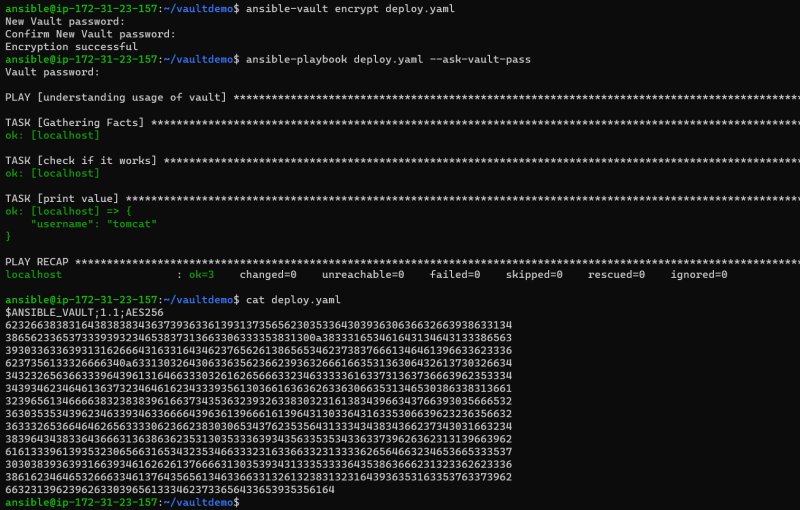
- name: check if it works

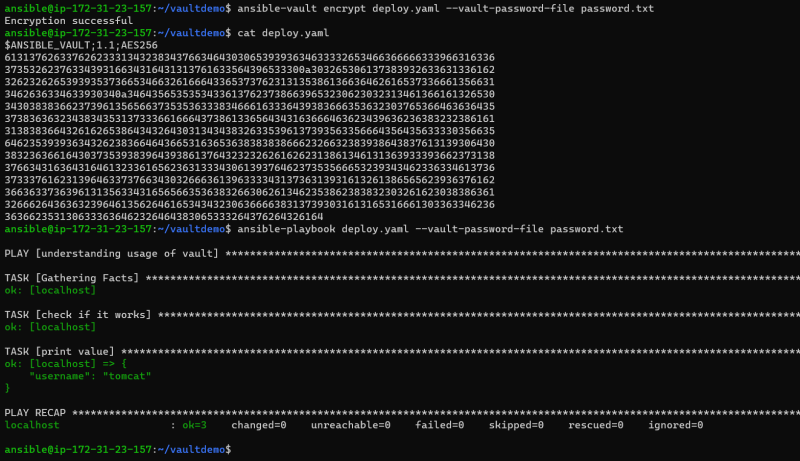
ping:

- name: print value

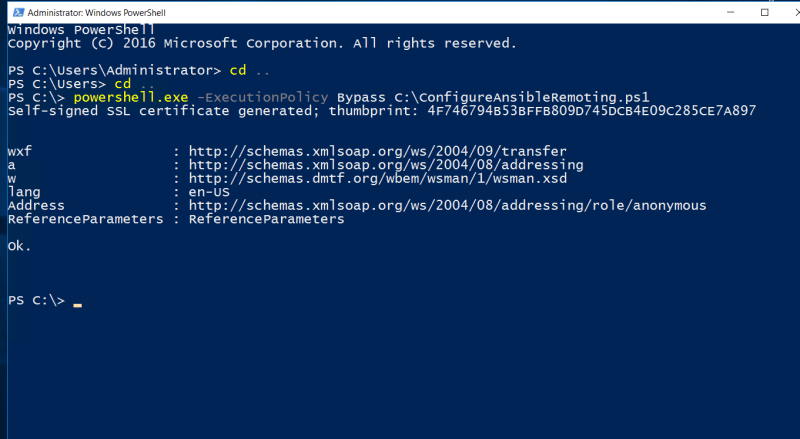
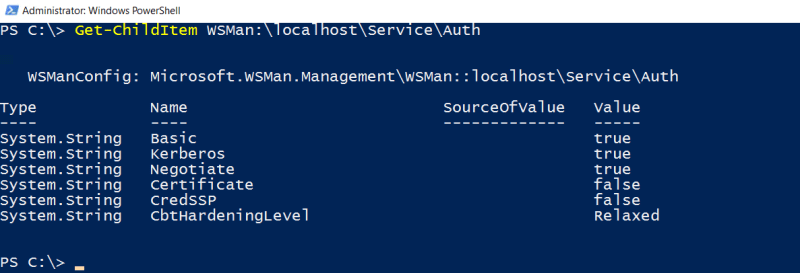
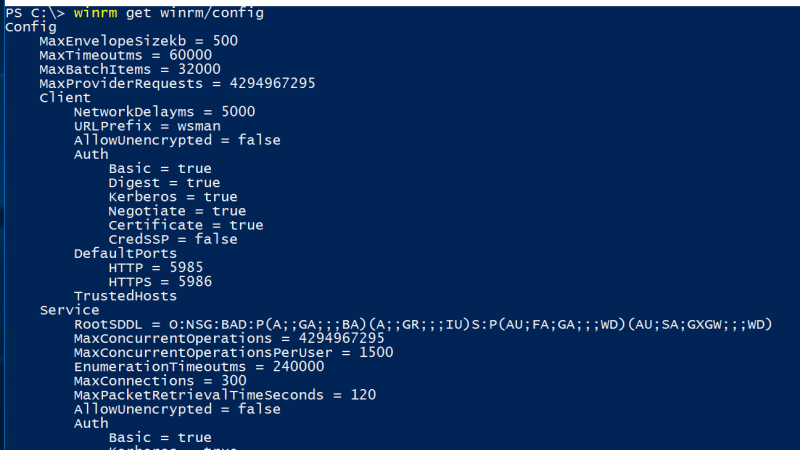
debug:

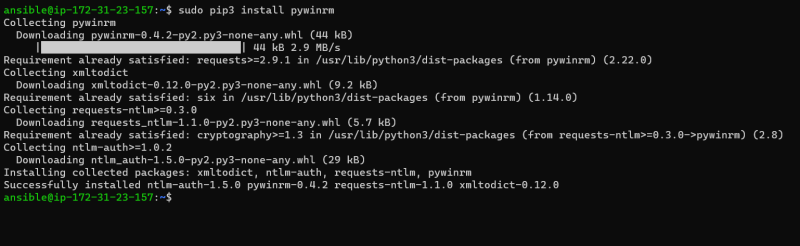
var: username

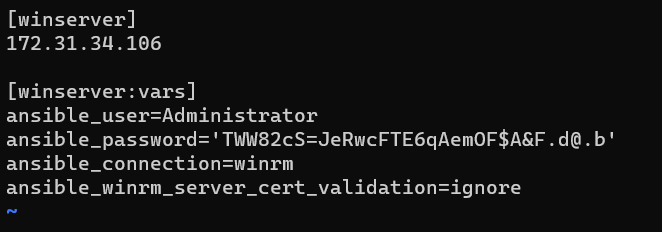
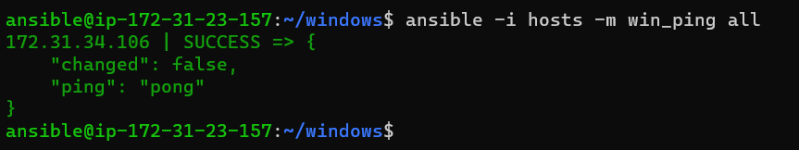


* Ansible vault can be used with password files as well 

## Ansible with Windows

* Create a Windows 2016 Server
* Ansible use winRM protocol to establish the communication with Windows Hosts
* Ansible requires atleastPowershell 3.0 or newer and atleast .net 4.0
* Steps for configuring windows Node:
  + Login into windows node
  + Launch Powershell as Admin and execute the following steps [Refer Here](https://docs.ansible.com/ansible/latest/user_guide/windows_setup.html#winrm-setup)
  + Create a file with the following content c:\ConfigureAnsibleRemoting.ps1
* [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
* $url = "https://raw.githubusercontent.com/ansible/ansible/devel/examples/scripts/ConfigureRemotingForAnsible.ps1"
* $file = "$env:temp\ConfigureRemotingForAnsible.ps1"
* (New-Object -TypeNameSystem.Net.WebClient).DownloadFile($url, $file)
* powershell.exe -ExecutionPolicyByPass -File $file
  + Now execute this file as shown in the below image   
  + Now login into the ansible control node and install pywinrm
* sudo apt install python3-pip -y
* sudo pip3 install pywinrm



* + Create an inventory file with behavior variables 
  + For linux machines to check the connectivity we use ping module for windows systems we have win\_ping [Refer Here](https://docs.ansible.com/ansible/2.9/modules/win_ping_module.html) 
  + [Refer Here](https://docs.ansible.com/ansible/2.9/modules/list_of_windows_modules.html) for the ansible windows modules
* Inventory for windows server with behavior variables

[winserver]

172.31.34.106

[winserver:vars]

ansible\_user=Administrator

ansible\_password='TWW82cS=JeRwcFTE6qAemOF$A&F.d@.b'

ansible\_connection=winrm

ansible\_winrm\_server\_cert\_validation=ignore

* Now lets try to write an ansible module to install iis server or windows

---

- name: installing iis on windows

hosts: winserver

tasks:

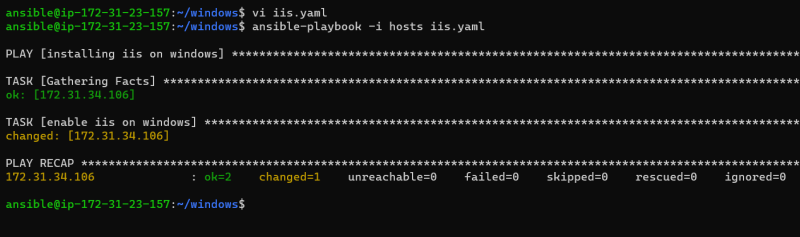
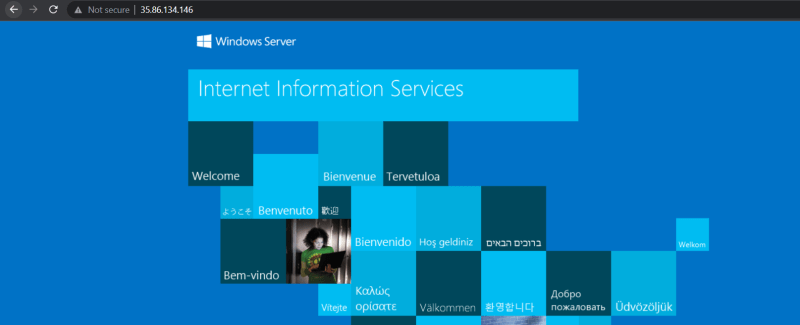
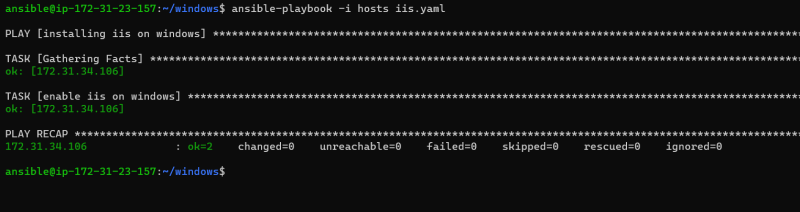
- name: enable iis on windows

win\_feature:

name: Web-Server

include\_management\_tools: yes

state: present

* Execution results  
* Letsreexecute the playbook to check idempotency 

### Share this:

[https://secure.gravatar.com/avatar/59911a510c38641070e0206183d394ee?s=48&d=identicon&r=g](https://directdevops.blog/author/learningthoughtsadmin/)

NOVEMBER 1, 2021

# DevOps Classroom Series – 31/Oct/2021

## Ansible Sample Deployments

* nop commerce [Refer Here](https://docs.nopcommerce.com/en/installation-and-upgrading/installing-nopcommerce/installing-on-linux.html)
* openmrs Enterprise [Refer Here](https://guide.openmrs.org/en/Getting%20Started/installation-and-initial-setup.html)
* Orange Hrm [Refer Here](http://telexcell.com/orange/installer/guide/)

## Ansible Tower Downloads

* Try downloading [Refer Here](https://www.redhat.com/en/technologies/management/ansible/try-it?extIdCarryOver=true&sc_cid=701f2000001OH7YAAW) and follow the instructions.

### Share this: