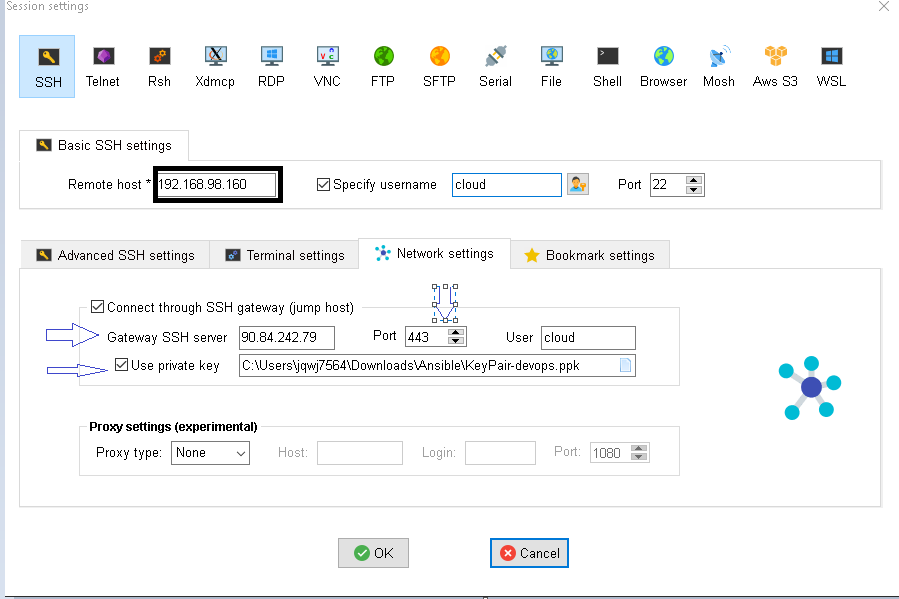
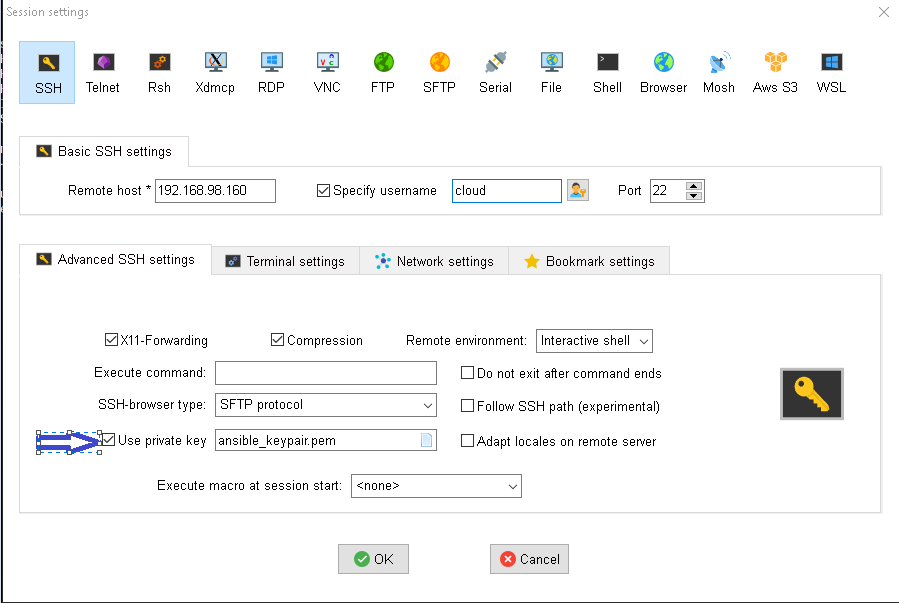
**Ansible**

**Labs Guide**

**Connect to Ansible Server & Client:**

* Connect on the your ansible VMs through Jump server “90.84.242.79” using *KeyPairdevops.ppk* on port 443.*-*
* Using *ansible\_keypair.pem* key to connect on your VM:



**Lab-0**

**Summary:**

Install ansible

**Steps:**

* Update OS
  + $ sudo yum update
* check PIP is exist
  + python3 -m pip -V

pip 21.0.1 from /usr/lib/python3.9/site-packages/pip (python 3.9)

* Install python3
  + sudo yum install python3
* install ansible
  + sudo yum install epel-release
  + sudo yum install ansible

**Lab-1**

**Summary:**

Explore ansible installation and Inventory file.

**Steps:**

* Check Ansible installation.

$ ansible --version

* Check Inventory path

$ vi /etc/ansible/hosts

* Add new managed host to the Inventory then save and exit
  + [webservers]
  + Webserver1 <client hostname>=<client IP> ansible\_user=cloud

Eg:

[webservers]

Webserver1 ansible\_host=192.168.66.189 ansible\_user=cloud

# where webserver1 dh asm

**Lab-1 cont.**

**Summary:**

* Execute sample ansible Ad Hoc commands to ping, show info and reboot on a target machine.

**Steps:**

* Make sure the default inventory file has the managed host.

[webservers]

Webserver1 ansible\_host=192.168.66.189 ansible\_user=root

## hana ana ba7ot el ip bta3 el client 3shan yab2a ya3rf ya connect 3leh

* Execute an Ad Hoc command to ping a managed host

$ ansible Webserver1 -m ping -k

$ ansible -m ping webservers --private-key=/etc/ansible/client.pem

* Execute an Ad Hoc command to setup a managed host

$ ansible Webserver1 -m setup -k

$ ansible -m setup webservers --private-key=/etc/ansible/client.pem

* Execute an Ad Hoc command to reboot a managed host

$ ansible Webserver1 -m reboot -k

$ ansible -m reboot webservers –become --private-key=/etc/ansible/client.pem

#--become == sudo fy ansible

**Lab-2**

**Summary:**

* Create a playbook to install and enable HTTPd service.

**Steps:**

* Check the managed host and make sure that no httpd is installed

$ rpm -qa httpd

* Create a new directory to hold the playbook and inventory files.

$ mkdir apache-project

$ cd apache-project

$ mkdir inventories

$ cd inventories

* Create a playbook file then save and exit

$ vi hosts

[webservers]

Webserver1 ansible\_host=192.168.98.195 ansible\_user=cloud ansible\_ssh\_private\_key\_file=/etc/ansible/client.pem

* Create a playbook file then save and exit

$ vi apache-playbook.yml

## Ta7t apache project

- hosts: webservers

become: yes

tasks:

- name: install httpd

yum:

name: httpd

state: installed

- name: install firewalld

yum:

name: firewalld

state: installed

- name: start and enable httpd

systemd:

name: httpd

state: started

enabled: yes

- name: permit traffic in default zone for https service

firewalld:

port: 80/tcp

permanent: yes

state: enabled

- name: reload service firewalld

systemd:

name: firewalld

state: reloaded

* Run the playbook

$ ansible-playbook apache-playbook.yml -i inventories/hosts

* Check the managed host and make sure that httpd is installed

$ rpm -qa httpd

* Check the httpd service status

$ systemctl status httpd

$ curl <http://192.168.98.195> > client

* Check the httpd service via curl by the managed host IP address.
* Create index.html file in the same directory of the playbook.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>This session about DevOps.</title>

</head>

<body style="background-color:Brown;">

<h1>This session about DevOps, lets GO!</h1>

</body>

</html>

* edit the playbook file to copy the file to the managed host then save and exit

$ vi apache-playbook.yml

Replace the playbook code with that code

---

- hosts: webservers

become: yes

tasks:

- name: install httpd

yum:

name: httpd

state: installed

- name: start and enable httpd

systemd:

name: httpd

state: started

enabled: yes

- name: permit traffic in default zone for https service

firewalld:

port: 80/tcp

permanent: yes

state: enabled

- name: reload service firewalld

systemd:

name: firewalld

state: reloaded

- name: copy index.html file to the managed host

copy:

src: index.html

dest: /var/www/html/index.html

* Rerun the playbook

$ ansible-playbook apache-playbook.yml -i inventories/hosts

* Check the httpd service via curl by the managed host IP address.
  + curl <http://192.168.98.195>

**Lab-3**

**Summary:**

* Modify the existing playbook to install and start HTTPd service using variables instead of hard-coding the values.

**Steps:**

* Modify the playbook, save and exist.

$ vi apache-playbook.yml

---

- hosts: webservers

become: yes

vars:

http\_port: 80/tcp

html\_page\_src: index.html

html\_page\_dest: /var/www/html/index.html

tasks:

- name: install httpd

yum:

name: httpd

state: installed

- name: start and enable httpd

systemd:

name: httpd

state: started

enabled: yes

- name: permit traffic in default zone for https service

firewalld:

port: "{{ http\_port }}"

permanent: yes

state: enabled

- name: reload service firewalld

systemd:

name: firewalld

state: reloaded

- name: copy index.html file to the managed host

copy:

src: "{{ html\_page\_src }}"

dest: "{{ html\_page\_dest }}"

* Run the playbook

$ ansible-playbook apache-playbook.yml -i inventories/hosts

* Modify the playbook, save and exist by removing the vars section.

$ vi apache-playbook.yml

---

- hosts: webservers

become: yes

#vars removed

tasks:

- name: install httpd

yum:

name: httpd

state: installed

- name: start and enable httpd

systemd:

name: httpd

state: started

enabled: yes

- name: permit traffic in default zone for https service

firewalld:

port: "{{ http\_port }}"

permanent: yes

state: enabled

- name: reload service firewalld

systemd:

name: firewalld

state: reloaded

- name: copy index.html file to the managed host

copy:

src: "{{ html\_page\_src }}"

dest: "{{ html\_page\_dest }}"

* Modify the inventory, save and exist to add vars.

$ vi inventories/hosts

[webservers]

Webserver1 ansible\_host=192.168.66.189 ansible\_user=root

[webservers:vars]

http\_port=80/tcp

html\_page\_src=index.html

html\_page\_dest=/var/www/html/index.html

* Rerun the playbook

$ ansible-playbook apache-playbook.yml -i inventories/hosts

* Check the httpd service via curl by the managed host IP address.

**Lab-4**

**Summary:**

* Modify the existing playbook to implement a handler of task “reload service firewalld” to reload it only in case of any change of firewall configuration.

**Steps:**

* Rerun the playbook and notice that the service reload always runs with no need.
* Modify the playbook, save and exist.

$ vi apache-playbook.yml

---

- hosts: webservers

become: yes

tasks:

- name: install httpd

yum:

name: httpd

state: installed

- name: start and enable httpd

systemd:

name: httpd

state: started

enabled: yes

- name: permit traffic in default zone for https service

firewalld:

port: "{{ http\_port }}"

permanent: yes

state: enabled

notify:

- reload service firewalld

- name: copy index.html file to the managed host

copy:

src: "{{ html\_page\_src }}"

dest: "{{ html\_page\_dest }}"

handlers:

- name: reload service firewalld

systemd:

name: firewalld

state: reloaded

* Rerun the playbook

$ ansible-playbook apache-playbook.yml -i inventories/hosts

* Check the output before and after:
  + Before





* + After



* Check the httpd service via curl by the managed host IP address.

**Lab-5**

**Steps:**

* Create playbook to touch multiple files.

vi lab5-playbook.yml

* Copy the playbook,

- hosts: all

tasks:

- name: creates users files

file:

path: /tmp/ansible-{{ item }}

state: touch

loop:

- sammy

- erika

- brian

* run the playbook

ansible-playbook lab5-playbook.yml -i inventories/hosts

**Lab-6**

**Summary:**

* Create playbook to shutdown if redhat OS based.

**Steps:**

* Create playbook to touch multiple files.

vi lab6-playbook.yml

* Copy the playbook

- hosts: all

tasks:

- name: Shut down Debian flavored systems

command: 'sudo /sbin/shutdown -t now'

#command: 'sudo /sbin/reboot'

when: ansible\_facts['os\_family'] == "RedHat"

• run the playbook

ansible-playbook lab6-playbook.yml -i inventories/hosts

**Lab-7**

**Summary:**

* Modify the apache-project existing playbook to implement a role that perform all the tasks related to apache installation and include it in the existing playbook.

**Steps:**

* Under existing directory of apache-project create a new directory of roles.

$ mkdir roles

* Under directory of roles create new directory of apache.

$ mkdir roles/apache

* Under directory of apache create new directories of tasks, handlers, defaults, vars, files, templates.

$ mkdir -p {tasks,handlers,defaults,vars,templates}

* Under tasks, create new file of main.yml and copy paste the below code.

$ vi tasks/main.yml

---

- name: install httpd

yum:

name: httpd

state: installed

- name: install firwall

yum:

name: firewalld

state: installed

- name: start and enable httpd

systemd:

name: httpd

state: started

enabled: yes

- name: permit traffic in default zone for https service

firewalld:

port: "{{ http\_port }}"

permanent: yes

state: enabled

notify:

- reload service firewalld

- name: copy index.html file to the managed host

copy:

src: "{{ html\_page\_src }}"

dest: "{{ html\_page\_dest }}"

* Under handlers, create new file of main.yml and copy paste the below code.

$ vi handlers/main.yml

---

- name: reload service firewalld

systemd:

name: firewalld

state: reloaded

* Under defaults, create new file of main.yml and copy paste the below code.

$ vi defaults/main.yml

http\_port: 80/tcp

html\_page\_src: index.html

html\_page\_dest: /var/www/html/index.html

* Modify the playbook, save and exist.

$ vi apache-playbook.yml

---

- hosts: webservers

become: yes

roles:

- apache

* Rerun the playbook

# Make sure that you removed vars from inventory file

$ ansible-playbook apache-playbook.yml -i inventories/hosts

* Check the httpd service via browser by the managed host IP address.

**Lab-8**

**Summary:**

* Configure NTP on the apache project by NTP role

**Steps:**

* create a new directory for your role. You can name it anything you like, but for this example, we'll call it "ntp\_config".

mkdir roles/ntp\_config

mkdir -p roles/ntp\_config/{tasks,handlers,defaults,vars,templates}

* create a new file called "main.yml" in the "tasks" directory.

vi roles/ntp\_config/tasks/main.yml

* Add the following code to the "main.yml" file:

---

- name: Install NTP package

 yum:

   name: ntp

   state: present

- name: Configure NTP

 lineinfile:

   path: /etc/ntp.conf

   regexp: '^server'

   line: 'server ntp.example.com'

 notify: restart ntpd

- name: Start NTP service

 service:

   name: ntpd

   state: started

   enabled: yes

- name: Set Timezone

 timezone:

   name: America/New\_York

   state: present

This role will do the following:

- Install the NTP package

- Configure the NTP server to use "ntp.example.com" to “server 0.africa.pool.ntp.org'”

- Start the NTP service and enable it to start at boot time

- Set the timezone to "America/New\_York" or “Africa/Cairo”

Note: You can replace "ntp.example.com" with the NTP server of your choice.

* add it in your playbook by adding the following code:

---

- name: Configure NTP on Linux servers

 hosts: all

 become: yes

 roles:

   - ntp\_config

**Lab-9**

**Summary:**

* Use ansible galaxy to download roles from GIT using URLs

**Steps:**

* Install Ansible Galaxy if you haven't already installed it.
  + $ sudo yuminstall ansible-galaxy
  + Sudo yum install git
* Create a roles/requirements.yml file in your Ansible project directory and add the following lines to it:

---

roles:

- src: https://github.com/Minaattiia/ntp-ansible-role.git

#scm: git

#version: main

* Run the following command to download the role:  
    
       $ ansible-galaxy install -r roles/requirements.yml