Lab: Adhoc Ansible

Overview

In this lab, we will configure Ansible to communicate with our servers and run some simple commands.

Lab: Adhoc Ansible

Instructions

Read this lab like a book, all text is there for a reason!

"→" denotes an action you must take

Use your favorite editor to edit files within the console. I suggest VI, nano, or emacs.

```
White boxes with black text denote commands and file contents
```

```
Black boxes with green text denote example output
```

Task 0: Install Ansible

Ansible is written in python, the most common way to install it is using the Python package manager, 'pip'.

Step 1: Install pip on your WorkSpace (if needed)

```
sudo easy_install pip
```

Step 2: Install Ansible

```
pip install --user ansible==2.8.3
```

```
Successfully installed ansible-2.8.3
```

That's it, you've installed Ansible!

Step 3: install sshpass

sshpass is required if we are going to use passwords to authenticate ansible.

```
wget http://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
sudo rpm -ivh epel-release-latest-7.noarch.rpm
sudo yum install sshpass
```

Task 1: Setup your inventory file

We now need a way to tell Ansible which hosts to manage and how to connect to them. We can easily do this by adding all of our hosts into an inventory file.

Step 0: Test your hosts

→ Test SSH connection to each of your VMs

```
ssh ubuntu@<VM_IP> "echo success"
```

Step 1: Add hosts to inventory

Inventory files are commonly named 'hosts'.

→ Create a file named 'hosts' in your home directory.

→ Using the docs for reference, add your two VMs into the hosts file.

Put one VM into a group named "db" and both of the VMs in a group named "web". So that one of the VMs is in both groups and the web group has 2 vms.

Step 2: Add auth variables

Now we need to tell ansible the username and password to use when connecting to our machines.

→ In your hosts file, add the following variables for all hosts, replace <PASSWORD> with your VM password.

```
ansible_connection=ssh
ansible_user=ubuntu
ansible_ssh_pass=<PASSWORD>
ansible_sudo_pass=<PASSWORD>
```

Task 1: Test connectivity

Step 0: Configure Ansible to not check hosts

→ Put the following contents into ~/.ansible.cfg

```
[defaults]
host_key_checking = False
```

Step 1: Execute the ping module

→ Run the following

```
ansible all -i hosts -m ping
```

```
35.239.103.48 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}

34.66.187.223 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
```

Bonus Setup passwordless SSH

Task 2: Install software

Lets install Apache Web Server on our machines. We will do this by using the 'apt' module saying we want 'apache2' to be present. Apt is the package manager for Ubuntu linux and 'Apache2' is the package we would like to install.

Step 1: Install Apache2 on both machines

→ Run the following

```
ansible all -i hosts -m "apt" -a "name=apache2 state=present" -b
```

```
34.66.187.223 | CHANGED => {
...
```

In order for ansible to install software on the remote machines, it needs to have permissions to do so. Our ubuntu user we have configured does not have all of the needed permissions, so here we provide the '-b' flag to tell ansible to run commands with sudo, effectively acting as the root user.

We can also see that the command caused a change on both hosts, showing us by the yellow output and the word "CHANGED".

→ Run the command again

```
ansible all -i hosts -m "apt" -a "name=apache2 state=present" -b
```

```
34.66.187.223 | SUCCESS => {
...
```

Here you can see nothing was changed, because both hosts already have the apache2 package installed.

Step 2: Confirm the install

→ Visit http://<VM1_IP> in the browser

You should see the Apache landing page

→ Visit http://<VM2_IP> in the browser

You should see the same on this host

Step 3: Cleanup

→ Ensure Apache is not installed

Lab: Adhoc Ansible

```
ansible all -i hosts -m "apt" -a "name=apache2 state=absent" -b
```

```
34.66.187.223 | CHANGED => {
...
```

→ Visit http://<VM1_IP> in the browser

You should get an error as it's no longer a webserver