Homework 1: Ansible

CMPE-172 Sec 01
Spring 2018
Team Name: **Team 2**
Team members:

- Timothy Davis
- Akshat Sharma

Code repository:

- https://github.com/timhdavis/cmpe-172-hw1-ansible

The purpose of this simple step-by-step process is to explain how our group setup Ansible to deploy a web page using a remote Ansible host.

Steps followed:

- Loosely followed tutorial: http://www.bogotobogo.com/DevOps/Ansible/Ansible_SettingUp_Webservers_Nginx_Install_Env_Configure_Deploy_App.php
- Used laptop (Mac OSX) as **control** and AWS EC2 instance as a **host**.
- Installed Ansible on laptop:
  - http://docs.ansible.com/ansible/latest/intro_installation.html
  - Followed instructions for Mac OSX. Used the following commands to install:

    ```
    $ sudo easy_install pip
    $ sudo easy_install pip
    ```
- Created an AWS EC2 instance (Linux).
  - Saved created key used for ssh into EC2 instance.
  - The permissions on the keyfile need to be changed in order to SSH into the EC2 instance using the keyfile. We change the permissions on key file using "**chmod 400 <keyfile>**"
  - Used the following command to change the permissions on the keyfile:

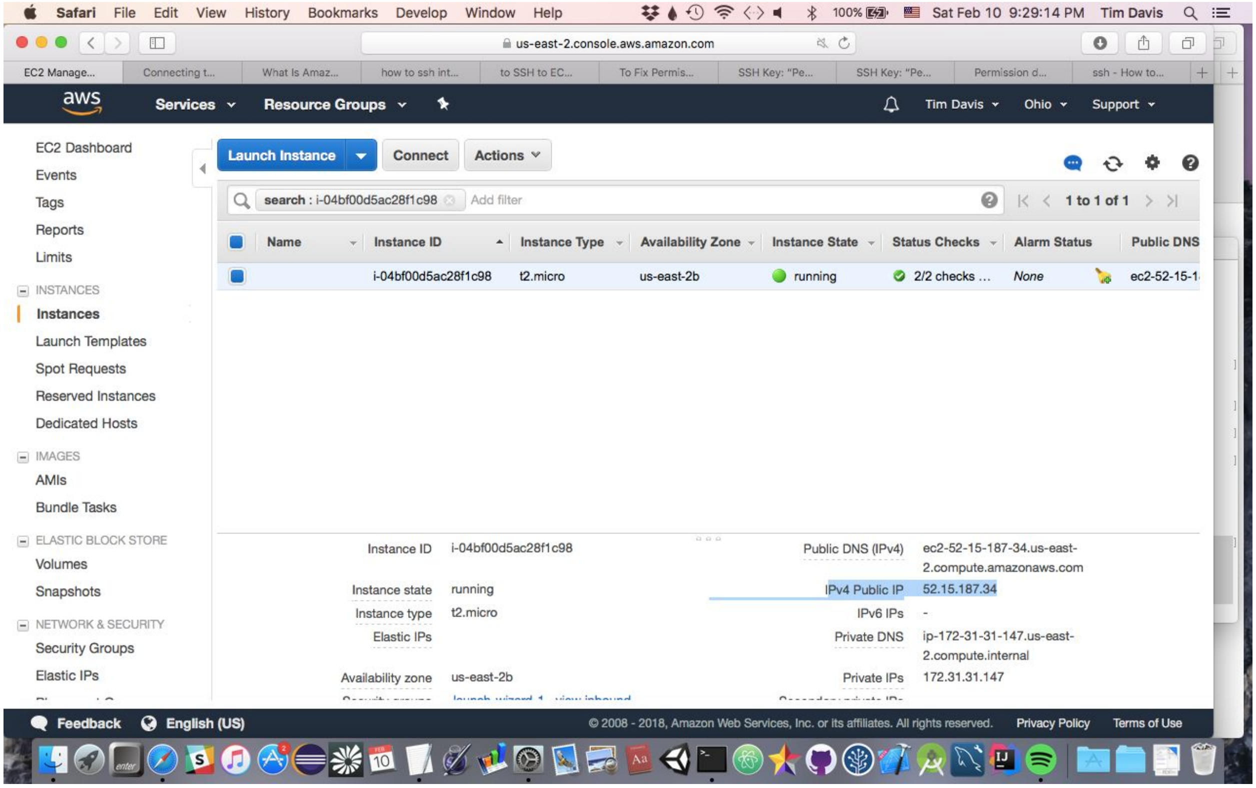
    ```
    chmod 400 ssh -i
    /Users/timhdavis/Downloads/aws-ec2-key1.pem.txt
    ```
- SSH into EC2 instance from laptop.
  - Used the following command:

```
tim-daviss-macbook:Ansible timhdavis$ ssh -i
/Users/timhdavis/Downloads/aws-ec2-key1.pem.txt
ec2-user@ec2-52-15-187-34.us-east-2.compute.amazonaws.com
```

  - Successful SSH will show the AWS EC2 connection confirmation:

```
       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2017.09-release-notes/
4 package(s) needed for security, out of 5 available
Run "sudo yum update" to apply all updates.
```

- Find the Public IP of the host.
  - We can view the AWS EC2 dashboard to find out the IPv4 Public IP:



search : i-04bf00d5ac28f1c98

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
	i-04bf00d5ac28f1c98	t2.micro	us-east-2b	running	2/2 checks ...	None	ec2-52-15-1

Instance ID i-04bf00d5ac28f1c98

Instance state running

Instance type t2.micro

Elastic IPs

Availability zone us-east-2b

Public DNS (IPv4) ec2-52-15-187-34.us-east-2.compute.amazonaws.com

IPv4 Public IP 52.15.187.34

IPv6 IPs -

Private DNS ip-172-31-31-147.us-east-2.compute.internal

Private IPs 172.31.31.147

  - We can see the IP address is **52.15.187.34**

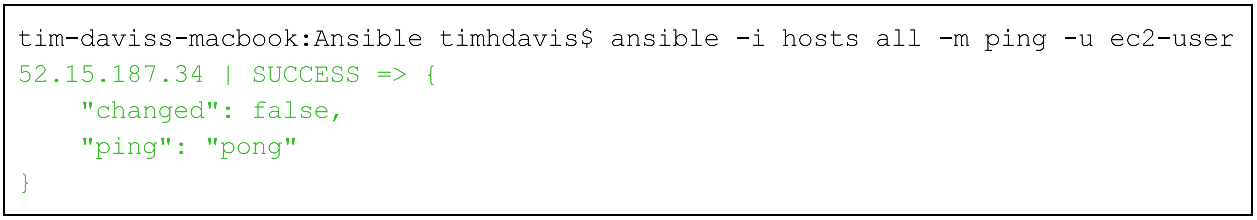
- Create Hosts file:

```
[webservers:vars]
ansible_ssh_private_key_file=./aws-ec2-key1.pem.txt
```

```
[webservers]
52.15.187.34
```

Hosts File: *hosts*

    - We put the EC2 public IP address (52.15.187.34) under a “webservers” field (to be used by our playbook to identify this host group).
    - We also added a “webservers:vars” field that allows us to say where the key needed to SSH into the EC2 instance is stored.
- We can check that we can reach our host with Ansible. Here is a successful ansible ping to EC2 instance:



```
tim-daviss-macbook:Ansible timhdavis$ ansible -i hosts all -m ping -u ec2-user
52.15.187.34 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```

- Create the Ansible Playbook to deploy our host:

```
---

- hosts: webservers
  remote_user: ec2-user

  vars:
   - MyMessage: "Hello world!"

  tasks:
   - name: Nginx setup
     become: true
     yum: pkg=nginx state=installed update_cache=true

   - name: index.html copy
     become: true
     template: src=index.html.j2
dest=/home/ec2-user/nginx/index.html
...
```

Playbook File: *server-setup.yaml*

  - The YAML file starts with “---” and ends with “...”
  - We list the hosts that this playbook should affect. We are using our “webservers” host group defined in the hosts file which contains the IP address of our EC2 instance.
  - We then state that the user for our EC2 instance is name “ec2-user”
  - We set a variable “MyMessage”, which will be used in an index.html page that we will display.
  - Next we define that tasks for the play:
    - The first task is to setup NGINX (our web server) on the host. We use “become: true” in order to use *sudo* to perform the command.
    - The second task is to move the index.html page (defined below) to the host (the page that we will deploy). We need to use “become: true” again in order to use *sudo* to perform the command.
- Before we run the playbook, we can create the index.html page that will be displayed:

```
<html>
<body>

<h1>Ansible Demo</h1>
<p>{{MyMessage}}</p>

</body>
</html>
```

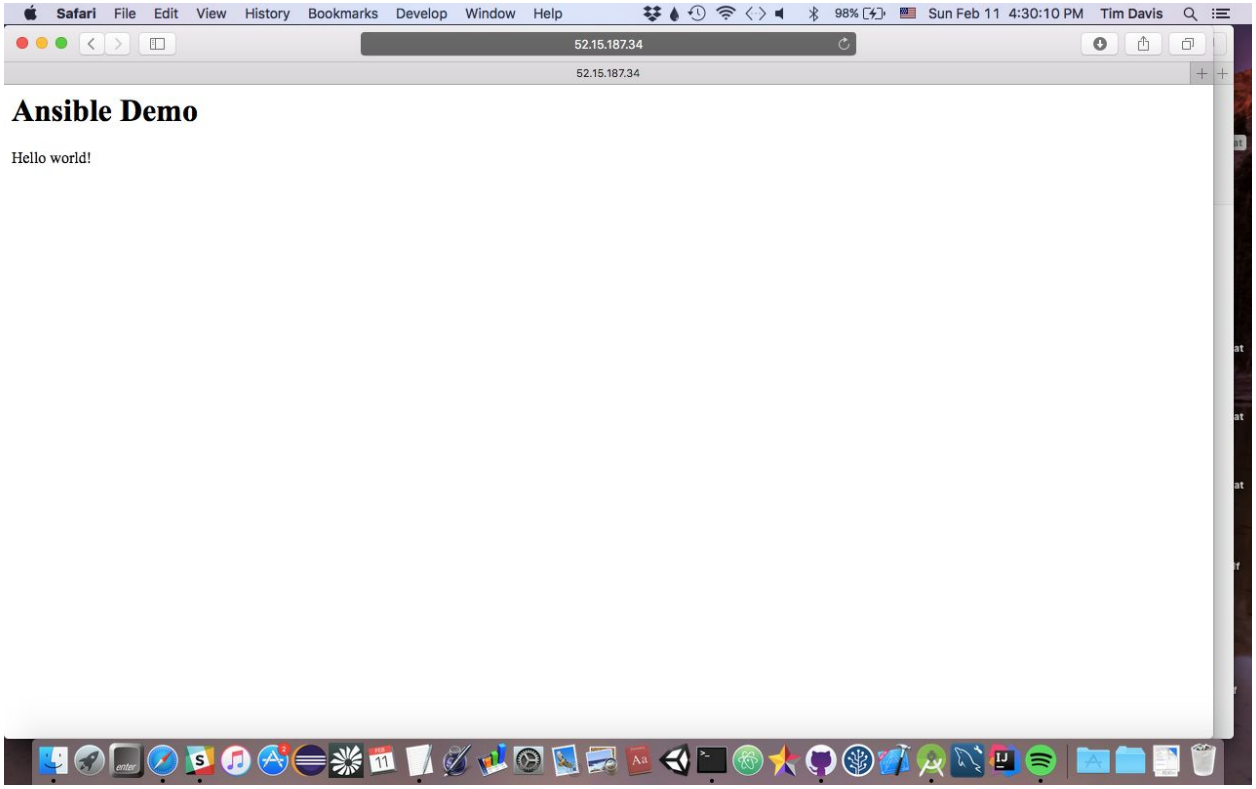
Web page: index.html.j2

  - We place our variable “MyMessage” defined in the playbook between “{{“ and “}}”.
- Now we can run the playbook:

```
tim-daviss-macbook:Ansible timhdavis$ ansible-playbook -i hosts server-setup.yaml
```

```
PLAY [webservers]
*********************************************************************************************
************************************************************************

TASK [Gathering Facts]
*********************************************************************************************
*******************************************************************
ok: [52.15.187.34]

TASK [Nginx setup]
*********************************************************************************************
***********************************************************************
ok: [52.15.187.34]

TASK [index.html copy]
*********************************************************************************************
*******************************************************************
changed: [52.15.187.34]

PLAY RECAP
*********************************************************************************************
*******************************************************************************
52.15.187.34               : ok=3    changed=1    unreachable=0    failed=0
```

  - All tasks in our Playbook successfully ran, meaning our web page has been deployed and the NGINX server is running.
- Visit the deployed page:
  - We go to our hosts IP in a web browser (http://52.15.187.34) to view the page we deployed:



Safari File Edit View History Bookmarks Develop Window Help
98%
Sun Feb 11 4:30:10 PM Tim Davis
52.15.187.34
52.15.187.34

Ansible Demo

Hello world!

  - The index.html page is displayed and the variable ("Hello world!") was correctly inserted into the HTML.
- Now we can write a playbook that stops the server:

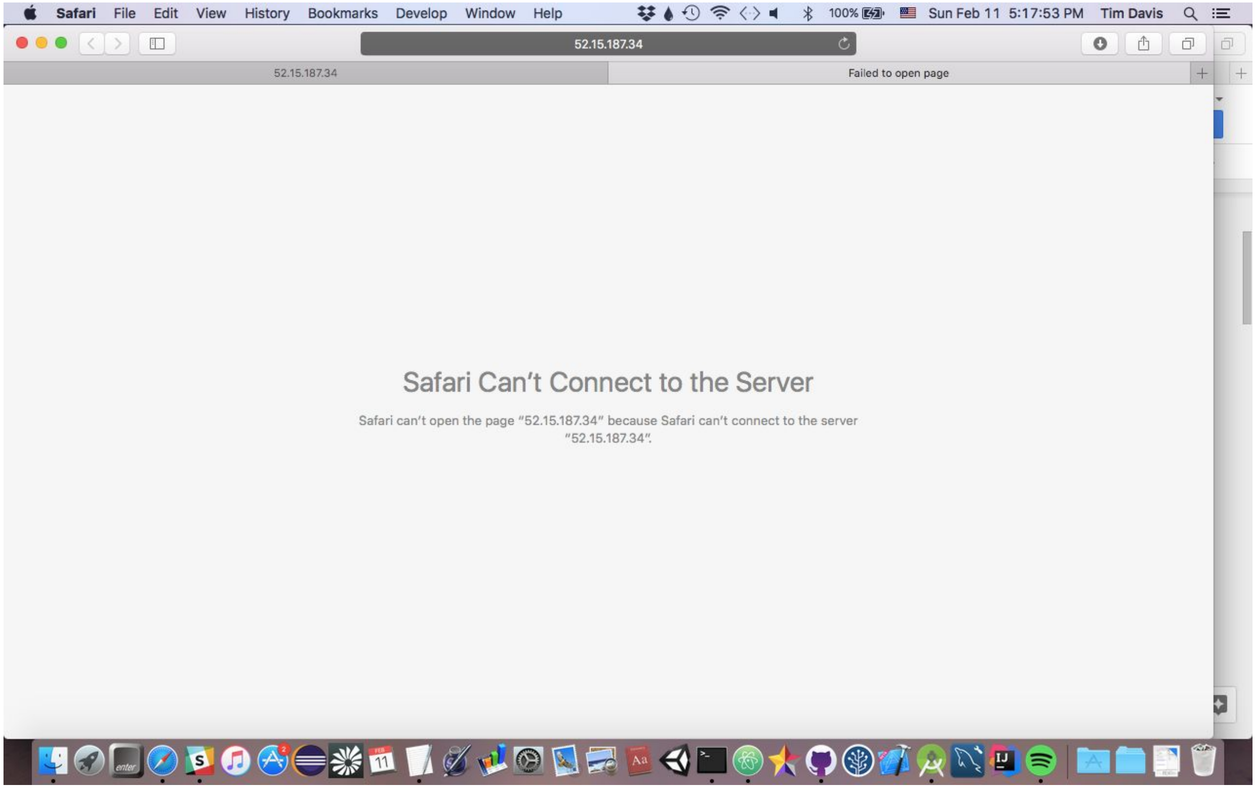
```
---

- hosts: webservers
  remote_user: ec2-user

  tasks:
   - name: stop nginx service
     become: true
     service: name=nginx state=stopped

...
```

Playbook file: server-stop.yaml

    - This playbook acts on the "webservers" host group (with our EC2 instance).
    - We are using the EC2 instance user "ec2-user".
    - The only task is to stop the NGINX server. We need to use *sudo*, so we use "become: true". We then change the state of the NGINX service to stopped.
- Run the Playbook to undeploy the web page:

```
tim-daviss-macbook:Ansible timhdavis$ ansible-playbook -i hosts server-stop.yaml

PLAY [webservers]
*********************************************************************************************
************************************************************************

TASK [Gathering Facts]
*********************************************************************************************
*******************************************************************
ok: [52.15.187.34]

TASK [stop nginx service]
*********************************************************************************************
****************************************************************
changed: [52.15.187.34]

PLAY RECAP
*********************************************************************************************
*******************************************************************************
52.15.187.34               : ok=2    changed=1    unreachable=0    failed=0
```

    - All tasks in our Playbook successfully ran, meaning our web page has been undeployed and the NGINX server has been stopped.
- Verify server stopped:
    - We reload the page to show that the server has stopped and the application has been undeployed:



Safari Can’t Connect to the Server

Safari can’t open the page “52.15.187.34” because Safari can’t connect to the server “52.15.187.34”.

Notes and files:

Playbooks syntax:

- About YAML syntax: http://docs.ansible.com/ansible/latest/YAMLSyntax.html

---

Hosts file:

- hosts

```
[webservers:vars]
ansible_ssh_private_key_file=./aws-ec2-key1.pem.txt

[webservers]
52.15.187.34
```

Web page to deploy:

- index.html.js

```
<html>
<body>

<h1>Ansible Demo</h1>
<p>{{MyMessage}}</p>

</body>
</html>
```

Playbooks:

- server-setup.yaml

```
---

- hosts: webservers
  remote_user: ec2-user

  vars:
   - MyMessage: "Hello world!"

  tasks:
   - name: Nginx setup
     become: true
     yum: pkg=nginx state=installed update_cache=true

   - name: index.html copy
     become: true
     template: src=index.html.j2
dest=/home/ec2-user/nginx/index.html
...
```

- server-stop.yaml

```
---

- hosts: webservers
  remote_user: ec2-user
  tasks:
   - name: stop nginx service
     become: true
     service: name=nginx state=stopped

...
```