**Lab 2: Scale Cloud Services Using Orchestration**

Adam Livingston

University of Arizona

NETV 379 SUM22 101-104 201-202: Cloud Computing

Professor Henry Werchan

26 June 2022

**Intro and Background**

In this lab, students were assigned to use their existing Amazon Web Services (AWS) account and skills learned from Lab 1. Their previously-created web server from Lab 1 will be configured with basic orchestration and automation by using Ansible-based configuration management. This lab requires some basic proficiency in Windows CLI and the Ubuntu Linux distribution. The steps annotated are for Windows 10 users. After instructions are given to type information, it is implied that the user presses the Enter key afterwards.

**Step 1: Remotely Access EC2 Server from Lab 1**

**Note**: that these steps are annotated by **not** using the new AWS interface. After successful login to AWS account, you will be at the AWS Management Console. Click on the “EC2” icon. The opened page will now default to your existing instances. Ensure that your instance is running, indicated under that column “Instance State” and will read “Running” with a green check mark to its left. Using steps from Lab 1, log in to your EC2 instance using Windows Command Prompt. After logging in, there may be a prompt that states “System restart required”. Type “sudo systemctl restart apache2” to restart. You will then be prompted “Enter passphrase”. Press Enter for no passphrase. Press Enter again to confirm. Your command line is now back in the .ssh directory.

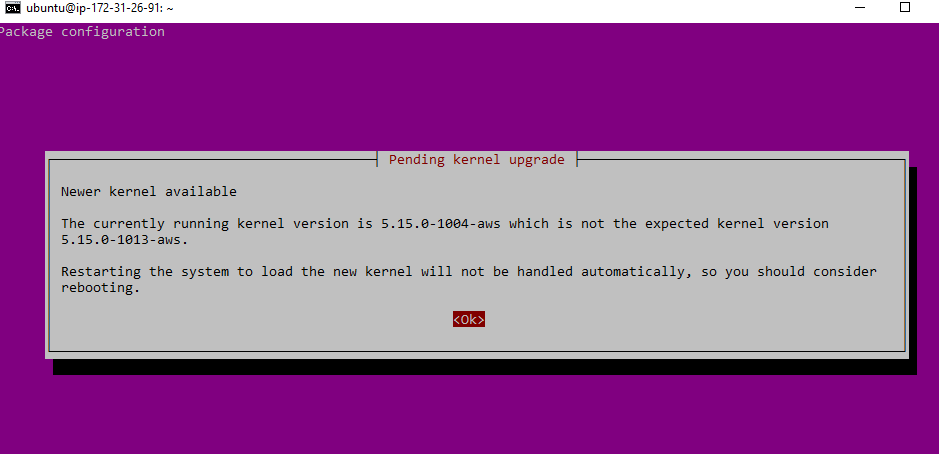
**Step 2: Connect to Localhost via SSH**

In order to setup Ansible, we need to connect as a localhost. In order to do so, we need to create a key pair (like from Lab 1) via the EC2 server. Type “cd .ssh” to access the Secure Shell (SSH) directory. Type “ssh-keygen” to generate a public/private RSA key pair. You well then be prompted “Enter file in which to save the key”. Press Enter.

We want to take that generated key and catenate its contents and append it to the authorized keys. Type “cat id\_rsa.pub >> authorized\_keys” to make that generated key an authorized key. Type “cd ..” to return to home command line. Type “ssh localhost” then type “yes” when you are asked if you want to continue connecting. We are now connected to ourselves with an established keypair.

**Step 3: Install Ansible**

Some software on the server may have updates that need to be installed. Type “sudo apt-get update” to update the software package. We also need to install software properties. Type “sudo apt-get install software-properties-common” and when prompted, type “y”. A screen may appear that states a newer kernel is available (*Fig.1*). Press Enter. It will then ask what services you want to update. Press Enter. The server is restarted and you need to reconnect to localhost as you did in Step 2.

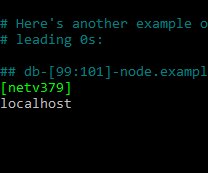
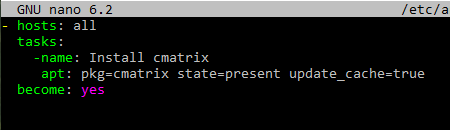


*Fig.1*

Now we need to add the Ansible Personal Package Archive (PPA) to extend the scope of the package library of Ubuntu. Type “sudo apt-add-repository ppa:ansible/ansible”. Press Enter to continue. Get updates by typing “sudo apt-get update”. We can now install ansible by typing “sudo apt-get install ansible” then typing “y” to continue. You may get a similar screen to *Fig.1.* Enter through and you are now back on the Ubuntu command line. Ansible is now installed.

**Step 4: Configure Ansible**

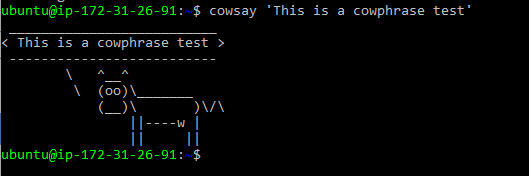
We need to create a custom Ansible inventory file. Type “cat /etc/ansible/hosts” to view the hosts in the Ansible inventory. Type “sudo nano /etc/ansible/hosts” to edit that inventory. You are now in the hosts file and can edit it. Using the arrow keys, scroll down to the bottom of the file and type the data located in *Fig.2*. Press Ctrl+X. Press Y. Press Enter. Localhost is now added in the approved Ansible hosts file. To verify this worked, type “ansible all -m ping”. If it worked, you will see “localhost | SUCCESS” as the return message.



*Fig.2 Fig.3*

For a fun program, type “sudo nano /etc/ansible/cmatrix.yml”. Enter in the information according to *Fig.3* and exit. Type “cmatrix” into the command line. You can press Ctrl+C to exit the program. This is one way to install a program with ansible. This way can also be more efficient than typing out the entire entry and have it placed in the command line.

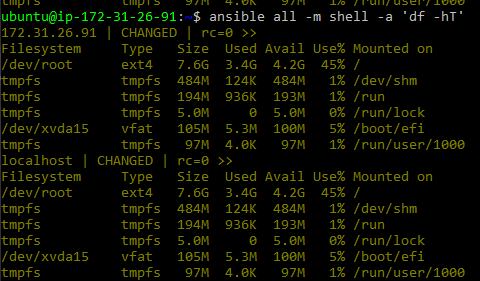
Another way to install programs using Ansible Shell (in this example is “cowsay”) is to use the command “ansible all -m shell -a ‘sudo apt-get install cowsay’”. This program has a graphic of a cow saying a phrase that you type with the command “cowsay ‘[phrase here]’” (see *Fig.4* for demonstration).



*Fig.4*

Even though these are silly programs for entertainment value, Ansible is a powerful tool and we want to expand the inventory file. By adding more hosts to the inventory file, instead of commands being ran multiple times for each host, the commands only need to be ran once and apply to all hosts. In this case, you can append the /etc/ansible/hosts file with your local IP address (**Note:** You can append the document with any number of IP addresses or usernames). This is demonstrative of the efficiency that Ansible brings.

To complete the last lab step, *Fig.5* is a screenshot of an Ansible query of free and used home folder disk space against the NETV379 inventory hosts.



*Fig.5*