# Plan for tonight session

## Project 1

Requirement

VCS – Gitbash

SCM – Github

1. Pull request, Merge Request and Track the changes - Real Time
2. Edit the code on github and follow the steps required to pull a request, get approval and successfully merge the request.
3. Track the changes real time

## Project 2

Automate the Provisioning of CI server – Jenkins, using Terraform

Requirements

Terraform

Install terraform on windows

Choco installs terraform -y

AWS account - Make sure all your resources are deleted before starting this project

SCM - Github - You will need to fork my REPO and clone it to a local REPO in your system

My REPO in Github - https://github.com/joshking1/terraform-for-advanced-class-.git

## After forking the REPO, CD into the folder called [

**terraform-for-advanced-class-**

# cd /terraform-for-advanced-class-

What is in the folder

1. Configuration files

Let define them

Compute.tf - this is the file that contains the aws computing power – ec2 or virtual server. This can also be part of the resources.tf

Main.tf can also be called provider.tf - contains the information regarding the provider (AWS)

Network.tf – It is called network.tf because it contains the resources deployed in our two networks. The major network is what we call the Virtual Private Cloud, where we can deploy the smaller network, related to the major network, called subnets. This configuration file can also be called resources.tf because it contains our resources.

Variables.tf – contains the name of the variables that we are going to be calling when deploying the resources with terraform.

Name of the variable

Type = string

```
variable "public_subnet_1_cidr" {
  type = string
  description = "Public Subnet 1 cidr block"
```

or

```
variable "public_subnet_1_cidr" {
  description = "Public Subnet 1 cidr block"
```

terraform.tfvars – this file contains the information that terraform will require to when reading the variables.

Example

**Variables.tf**

variable "keyname" {

}

**Terraform.tfvars**

keyname = "lynn"

Finally, the OUTPUT.TF file - this is the file that contains the information regarding the outputs we want terraform to execute after provisioning the infrastructure.

We are going to kindly; request terraform to provide the IP address of the Jenkins server after provisioning process is complete.

To provision jenkins server, terraform will run a script in bash language

.sh is the extension requirement for the script, for example hug.sh

The script will be written in bash and will do the following

- Install JDK
- Install Apache maven
- Install Docker.io
- Start and enable docker deamon
- Install git
- Install nginx
- Install apache
- Install Jenkins
- Start and enable Jenkins deamon

END

Output of Jenkins server ip address in the aws formt

Copy as it is, go on the browser and paste it, and add 8080.

The administrator password will be requested

WE provisioned the Jenkins using BLUE OCEAN CONTAINER

SSH into your Jenkins server

To do so, carry out the following key commands

# chmod 400 (name of your key pair) – ensures your key pair is not public

# ssh ec2-user@your ip address

You are the root

GREAT !!!!!!!

Let get the administrator password for Jenkins

We are used a blue ocean container to provision Jenkins with terraform

Let check docker version

# docker -v

Docker should be available because terraform provisioned everything for us.

Run

# docker ps

Blue ocean container should be running on docker engine

Get inside the container

# docker exec [container id] cat /path provided by jenkins

Password will be available for you

Complete the remaining steps on the jenkins console

# Project 3

Ansible can also provision AWS infrastructure

How is this possible?

Ansible has Python modules that enable it to interact with AWS. Assuming you have already installed Ansible, consider installing python-pip:

Requirements

Jenkins server - already provisioned for us by terraform

Remotely access the jenkins server  - Do not use putty! Use Visual Studio Code (VSC)

Become the root - # sudo su

Install ansible in jenkins instance

Follow the following instructions

https://aws.amazon.com/blogs/infrastructure-and-automation/automate-ansible-playbook-deployment-amazon-ec2-github/

# amazon-linux-extras install epel

# yum update -y

# yum install ansible -y

# yum install nginx -y

# yum install git -y – this should already be installed for you by terraform.

# ansible –version

Should be available

Now we have ansible, what next

# cd /home

# touch hug.yml  - playbook to provision an instance with ANSIBLE

# touch nat.yml  - playbook to terminate the **previously** provisioned server

**Let us enjoy some troubleshooting so that we can run why we need boto to provision an instance with ansible**.

I am doing this intentionally, do not worry.

Go to github and folk my REPO containing the two playbooks

REPO path: https://github.com/joshking1/-ansible--ec-creation-termination.git

Clone it to your local machine

Open it with VSC

Start with hug.yml

# vi and not vim – try vim and see what happens. I recommend you use vi

# vi hug.yml  - put the information from the cloned yaml file into this

# save [ ESC + Shift + Colon + x ]  or Simply [ ESC + Shift + Colon + WQ]

Run the command

# ansible-playbook hug.yml

**Trouble shooting scenario #1**

**Boto module**

Ansible will create an EC2 server for you

Go to you aws account and confirm

**We are still inside jenkins server**

Let us terminate jenkins server using Ansible

Go to github and folk my REPO containing the two playbooks

REPO path: https://github.com/joshking1/-ansible--ec-creation-termination.git

Clone it to your local machine

Open it with VSC

This time us nat.yml

# vi and not vim – try vim and see what happens. I recommend you use vi

# vi hug.yml  - put the information from the cloned yaml file into this

# save [ ESC + Shift + Colon + x ]  or Simply [ ESC + Shift + Colon + WQ]

Run the command

# ansible-playbook nat.yml

**Troubleshooting scenario #2**

**Boto module**

Ansible will terminate an jenkins server created by terraform because that is the server that we are in.

Will terminate and log out!!!!!!!!!!!!!1

Go to you aws account and confirm that your jenkins server has been terminated

Clean up the remaining resources with terraform

You will need to give terraform a push to finish the clean up process because the ec2 we deployed with ansible used the resources already provisioned by terraform.