3 - Infrastructure Provisioning (IaC)

<table>
<tr><th>#</th><th>Step</th><th></th></tr>
<tr><td>1</td><td>Terraform Cloud workspaces</td><td>
We need to maintain two workspaces - one for the Fabric Kubernetes cluster and one for the openIDL applications.

To create the workspaces use the tool located in senofi/openidl-devops:

1. Go to <code>openidl-devops/aws-infrastructure/environments/<env-folder>/terraform-cloud</code> and run

<pre>terragrunt plan</pre>
If everything looks ok, execute <code>terragrunt apply</code>. This should create two workspaces and a var set in Terraform Cloud.
2. The apply from step 1 should fail as the execution of the script is happening on the Terraform Cloud servers and the local config in <code>~/.terraformrc</code> is only useful to the communication from the localhost to the Terraform Cloud servers. In order to fix this, find in Terraform Cloud the new workspace that has the format of <code><org_id>-<env>-tf_cloud</code> and go to <code>Variables</code> section where a new environment variable will be needed with the name <code>TFE_CLOUD</code> , type environment variable and value the Terraform token that was previously saved in <code>~/.terraformrc</code>
3. Create a new KMS key (symetric, encrypt/decrypt) in the AWS console. The name is not important but use a meaningful name that will associate it with this environment. Use it to populate the property in the next step
4. Go to <code>openidl-devops/automation/terraform-cloud</code> and update <code>configuration.properties</code> Make sure that the var set name matches what is in Terraform Cloud
5. Create SSH keys

<pre>ssh-keygen -t rsa -f app_eks_worker_nodes_ssh_key.pem

ssh-keygen -t rsa -f blk_eks_worker_nodes_ssh_key.pem

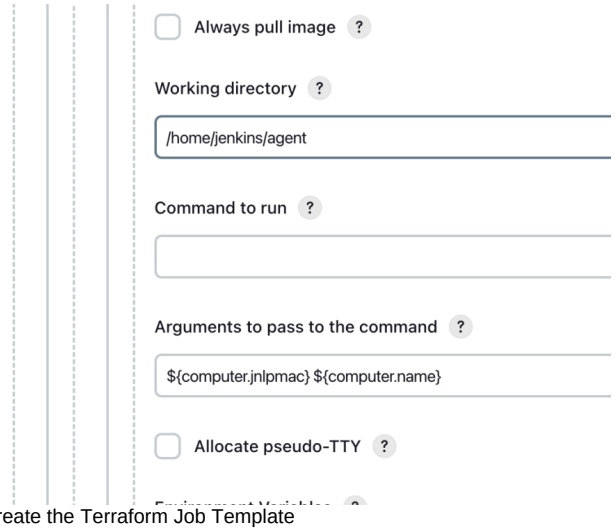
ssh-keygen -t rsa -f bastion_ssh_key.pem</pre>
6. Populate the variable set by executing the following command in <code>openidl-devops/automation/terraform-cloud</code>

<pre>pip install -r requirements.txt
python populate-variable-set.py</pre>
7. Copy the contents of the public keys and populate them in Terraform Cloud UI under Variable Sets <the newly created varset>

a. app_eks_worker_nodes_ssh_key.pem.pub
b. bastion_ssh_key.pem.pub
c. blk_eks_worker_nodes_ssh_key.pem.pub
</td></tr>
<tr><td>2</td><td>Configure Jenkins</td><td></td></tr>
</table>

1. Set Jenkins node label ‘openidl’ in Kubernetes Cloud by going to Manage Jenkins Manage Nodes and Clouds Configure Clouds. Make sure that under Pod Template details the labels field contains the value ‘openidl’.

Also, remove the prepopulated ‘sleep’ command if it is set on the pod template:



2. Create the Terraform Job Template
    a. Terraform Token Secret - Login to Jenkins go to Manage Jenkins Manage Credentials Stores scoped to Jenkins (Jenkins) Global Credentials (unrestricted) Add credentials

    b. Choose Kind as secret text, enter secret text like Token in “secret” field and name the secret ID as unique since it will be used in pipeline code.
    c. Git Credentials - Add a new credential

		3. Terraform Job a. Go to Jenkins New Item. Use a name such as `Terraform Job` b. Select job type as PIPELINE and proceed. c. Select Definition as Pipeline Script from SCM d. Select SCM as Git e. Key in the Infrastructure code repository (openidl-gitops) URL. f. Select the Git credential created above g. Specify the relevant branch “refs/heads/<branch-name>”. h. Set script path to `jenkins-jobs/jenkinsfile-tf`
3	Run Terraform Job	1. Run the Jenkins Terraform Job 2. Open the console log for the job. Once the job asks for an input accept and choose the apply option 3. The job runs a second plan into the Kubernetes workspace in Terraform Cloud. When asked - accept and apply the changes 4. Go to the AWS Console and find EKS (Elastic Kubernetes Service). Choose the blk cluster and go to Add-Ons. Find the EBS plugin and add it to the list. The plugin makes sure volumes could be created in Kubernetes
4	Add Amazon EBS CSI Driver Add-on	1. Go to AWS Console Elastic Kubernetes Service 2. Find the new app and k8s clusters that were created in the previous step 3. Open each of them and go to Add-ons tab 4. Select Get more add-ons and select the Amazon EBS CSI Driver 5. Choose Next on the bottom of the page 6. Review and click Create button 7. Repeat for the other cluster