Terraform activity

terraform apply -auto-approve (update terraform.tfvars as per requirement)

Jenkins Acitvity

sudo cat /var/lib/jenkins/secrets/initialAdminPassword

Jenkins Agent machine activity

docker login -u <dockerhub-username> -p <dockerhub-password>

cd /opt

sh install-scout.sh

ssh-keygen

cd ~/.ssh

cat id_ed25519.pub >> authorized_keys

cat id_ed25519

Sonarqube activity

sudo -i

su - postgres

createuser sonar

psql

ALTER USER sonar WITH ENCRYPTED password 'sonar';

CREATE DATABASE sonarqube OWNER sonar;

GRANT ALL PRIVILEGES ON DATABASE sonarqube to sonar;

\q

exit

sudo systemctl daemon-reload

sudo systemctl enable --now sonar

sudo systemctl start sonar

Goto Administration -> Configuration -> Webhooks -> Create -> Add in URL <http://jenkins-public-ip:8080>/sonarqube-webhook/>

Kubernetes activity

From the local system run below command to update local kubeconfig file.

aws eks update-kubeconfig --region us-east-1 --name EKS-Cluster

Apply cluster-role.yml & role-binding.yml

kubectl apply -f cluster-role.yml

kubectl apply -f role-binding.yml

Edit the configmap

kubectl edit -n kube-system configmap/aws-auth

Add the following

- rolearn: arn:aws:iam::aws_acc_id:role/eks-admin

  username: eks-admin

  groups:

  - system:masters

Looks similar below

```
apiVersion: v1
data:
  mapRoles: |
    - rolearn: arn:aws:iam::873330726955:role/eks-admin
      username: eks-admin
      groups:
      - system:masters
    - groups:
      - system:bootstrappers
      - system:nodes
      rolearn: arn:aws:iam::873330726955:role/eks-worker-node-role
      username: system:node:{{EC2PrivateDNSName}}
kind: ConfigMap
```

Goto IAM users in AWS console and create a cli credential of the user eks-jenkins (created by terraform)

Login to Jenkins-Agent machine and configure the credential

aws configure --profile manager

Update the kubeconfig file

vim ~/.aws/config

Add below details

[profile eks-admin]

role_arn = arn:aws:iam::aws_acc_id:role/eks-admin

source_profile = manager

Looks similar to the below

```
[profile manager]
region = us-east-1
[profile eks-admin]
role_arn = arn:aws:iam::873330726955:role/eks-admin
source_profile = manager
~
```

kubeconfig update

aws eks update-kubeconfig --region us-east-1 --name EKS-Cluster --profile eks-admin

Jenkins Activity

Add credentials of github and dockerhub

Add github credentials (dockerhub similar)

- Goto Manage Jenkins -> Credentials -> Domains (Global) -> Add Credentials ->
- Kind = username with password
- Username = Prabhat-roy
- Password = Github Token
- ID = github
- Create

Create a ssh credential of Jenkins agent machine using username private key (created earlier steps).

- Goto Manage Jenkins -> Credentials -> Domains (Global) -> Add Credentials ->
- Kind = SSH username with private key
- ID = ubuntu
- Description = Ubuntu Cred
- Username = ubuntu
- Private key -> Enter directly -> Add -> paste the value of id_ed25519 from agent machine -> Create

Login to sonarqube on browser using admin/admin credentials and create a credential through my account and add that in jenkins as secret text.

Credentials looks similar below

Credentials

T	P	Store ↓	Domain	ID	Name
		System	(global)	Jenkins-Agent	ubuntu (Jenkins-Agent)
		System	(global)	sonar	sonarqube cred
		System	(global)	dockerhub	prabhatrkumaroy/****** (dockerhub cred)
		System	(global)	github	prabhat-roy/****** (github cred)

Go to manage Jenkins and below plugins

- SSH
- SSh Agent
- SSH Pipeline Steps
- Docker
- Docker Commons
- Docker Pipeline
- Docker API
- docker-build-step
- Amazon ECR
- Kubernetes
- Kubernetes Client API
- Kubernetes Credentials
- Kubernetes CLI
- Kubernetes :: Pipeline :: DevOps Steps
- SonarQube Scanner
- Sonar Quality Gates

- Quality Gates
- OWASP Dependency-Check
- OWASP Dependency-Track
- Official OWASP ZAP
- ZAP Pipeline
- nodejs

Add agent machine (steps)

Goto Manage Jenkins -> Nodes -> New Node -> Name, Type (Permanent Agent) -> Remote root directory -> /home/ubuntu -> Labels (Agent) -> Launch Method -> via SSH -> Host -> private ip of the agent -> Credentials -> choose from dropdown -> Host Key verification Strategy -> non verifying -> Save

System configuration

Go to manage Jenkins -> System -> SonarQube Servers -> Add -> Name -> SonarQube -> Server URL -> http://10.0.1.170:9000 -> Server authentication token -> sonarqube credential (from dropdown) -> Apply

Tools configuration

Goto Manage Jenkins -> Tools -> JDK installations -> Add -> Name -> Java -> JAVA_HOME -> /usr/lib/jvm/java-21-openjdk-amd64 -> Add -> SonarQube Scanner installations -> Add -> SonarQube Scanner -> Name -> sonar -> Install automatically -> Install from Maven Central -> latest -> Add -> Maven installations -> Add Maven -> Name -> Maven -> MAVEN_HOME -> /opt/apache-maven-3.9.9/ -> Add -> Dependency-Check installations -> Add Dependency-Check -> Name -> DP-Check -> Install from github.com -> Install from github.com -> version -> latest -> Add -> Add Node JS -> latest -> Apply

Pipeline configuration

Dashboard -> New item -> Name -> pipeline -> ok

Build Triggers -> GitHub hook trigger for GITScm polling -> Pipeline -> Pipeline script from SCM -> SCM -> git -> Repository URL -> https://github.com/prabhat-roy/java-deployment-eks-using-jenkins-terraform.git -> Credentials -> github (select from dropdown) -> Branches to build -> Branch Specifier (blank for 'any') -> */main -> Script Path -> Jenkinsfile.

Github webhook configuration

Go to repository -> settings -> webhook -> Webhooks / Manage webhook -> Payload URL -> http://jenkins_public_ip:8080/github-webhook/ -> Pushes -> Pull requests

Output

Copy the URL after execution of the pipeline script and paste into browser. Output similar to below.

Java Maven Application

Java Maven application deployment to EKS AWS using helm terraform