r



**AIOPS IAAS DEPLOYMENT**

**By: NAUT Mosaic Automation Team**

**Table of Contents**

[**1.** **Introduction** 2](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380227)

[a. Objective 2](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380228)

[b. Assumptions 2](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380229)

[**2.** **Implementation Plan** 3](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380230)

[**3.** **AIOPS Architecture** 4](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380231)

[**4.** **Pre-Implementation check** 5](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380232)

[**5.** **Implementation Steps** 6](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380234)

[a. JAVA installation 6](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380235)

[b. MySQL installation 7](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380236)

[c. Keycloak Configuration 8](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380237)

[d. PostgreSQL Setup 1](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380239)4

[e. Kafka Configuration 1](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380238)8

[f. Docker, Kubernetes Installation and Configuration 21](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380240)

[g. Configuring YAMLs 24](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380241)

[**6.** **Post Installation checks** 39](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380243)

[**7.** **Backup and Restore** 40](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380244)

[a. MySQL Backup](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380245) 40

[b. PostgreSQL Backup 40](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380246)

[c. Kafka Backup 40](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380247)

[**8.** **Removal of the existing deployment** 41](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380248)

[a. Remove Java 41](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380249)

[b. Remove MySQL 41](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380250)

[c. Remove keycloak 41](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380251)

[d. Remove Postgresql 41](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380252)

[e. Remove kafka 42](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380253)

[f. Remove Kubernetes 42](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380254)

[**9.** **Conclusion** 43](file:///C:/Users/10662080/Downloads/AIOps%20Paas%20Deployment%20Document.docx#_Toc43380257)

# **Introduction**

## **Objective**

The objective of this document is to improve the understanding of the complete product implementation by examining the different technology stacks, services and deployments.

1. **Assumptions**

We assume that whomsoever referring this document for AIOPS deployments must have:

* Basic understanding of Docker, Kubernetes, Kafka and Databases.
* Knowledge of Networks
* VPN connection

1. **Implementation Plan**



1. **AIOPS Architecture**

A close up of a map

Description automatically generated

# **Pre-Implementation check**



# **Implementation Steps**

## **JAVA Installation**

* To install Java

*# sudo apt install java-1.8.0-openjdk*

1. **MYSQL Installation**

* Install the mysql server default version 5.7 for keycloak configuration.

*# sudo apt install mysql-server*

* To check the MySQL version.

*# mysql –V*

* Start the mysql daemon service.

*# sudo systemctl start mysql.service*

* To see the temporary password

*# sudo grep 'temporary password’ /var/log/mysql.log (use recent password)*

* For secure installation of Mysql to restrict the access of the database for security.

*# sudo mysql\_secure\_installation*

* Login to Mysql

*# mysql -u root –p*

* Creating User in database with password for Keycloak configuration

*# CREATE USER 'keycloak'@'%' IDENTIFIED BY 'Keycloak@123';*

* Create database in Mysql

*#**CREATE DATABASE keycloak CHARACTER SET utf8 COLLATE utf8\_unicode\_ci;*

* Giving all privileges to keycloak user on valid host

*# GRANT ALL PRIVILEGES ON keycloak.\* TO 'keycloak'@'%';*

## **Keycloak Configuration**

* To Create a directory named app

*# sudo mkdir /app*

* To give the ownership of app directory to user “automation”

*# sudo chown automation:root -R /app*

* To get inside app directory

*#* *cd /app*

* Download the Jboss zip file

*# wget --user maxiq --ask-password* [*http://35.168.48.43/repo/keycloak/jboss.zip*](http://35.168.48.43/repo/keycloak/jboss.zip) *(do this in browser)*

* Unzip <filename.extension>

*# unzip jboss.zip*

* Set variables into .bashrc, for that refer attach file keycloak-env-final.txt(attached herewith)



Edit the parameters : Hostname and Java\_Home. Paste this notepad file to .bashrc.

*# sudo vi /home/automation/.bashrc*

* After saving bashrc file do, (in /home/automation)

*# source .bashrc*

* Open the standalone.xml file and edit the content.

*# sudo vi /app/jboss/keycloak/standalone/configuration/standalone.xml*

Replace <<private ip>> in the following lines.

<inet-address value="${jboss.bind.address.management:<<private ip>>}"/> <inet-address value="${jboss.bind.address:<<private ip>>}"/>

* Move to Jboss folder

*# cd $JBOSS\_HOME*

* Start the process

*# nohup ./bin/standalone.sh &*

* http://IP:8080 (in browser)
* To create admin user and password refer below command

*# ./bin/add-user-keycloak.sh -u keycloak -p keycloak*

* Kill the process and restart Keycloak again

*# sudo lsof –i:8080*

*#* sudo kill -9 << PID >>

* Start the process again

*# nohup ./bin/standalone.sh &*

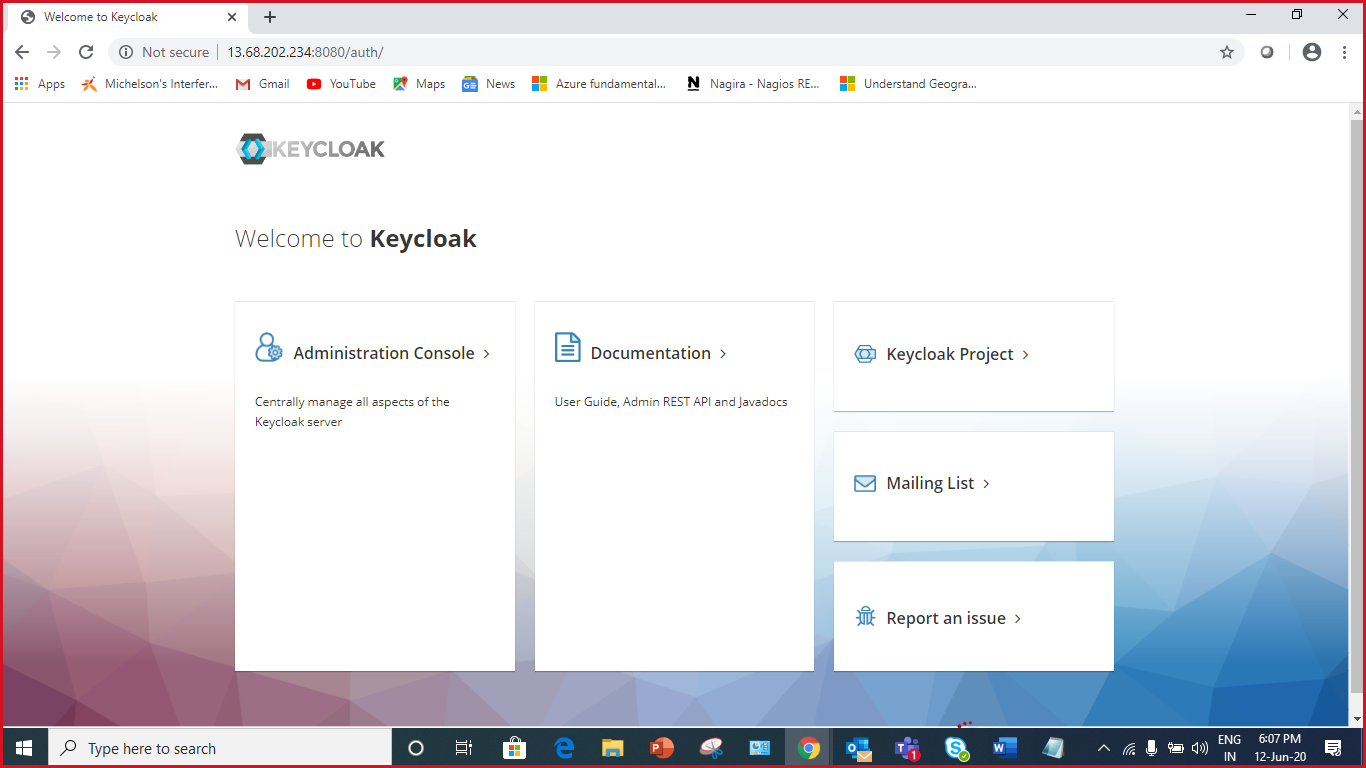
* To create realm master refer below command

*# ./bin/kcadm.sh config credentials --server http://<<private IP>>:8080/auth --realm master --user keycloak --password keycloak*

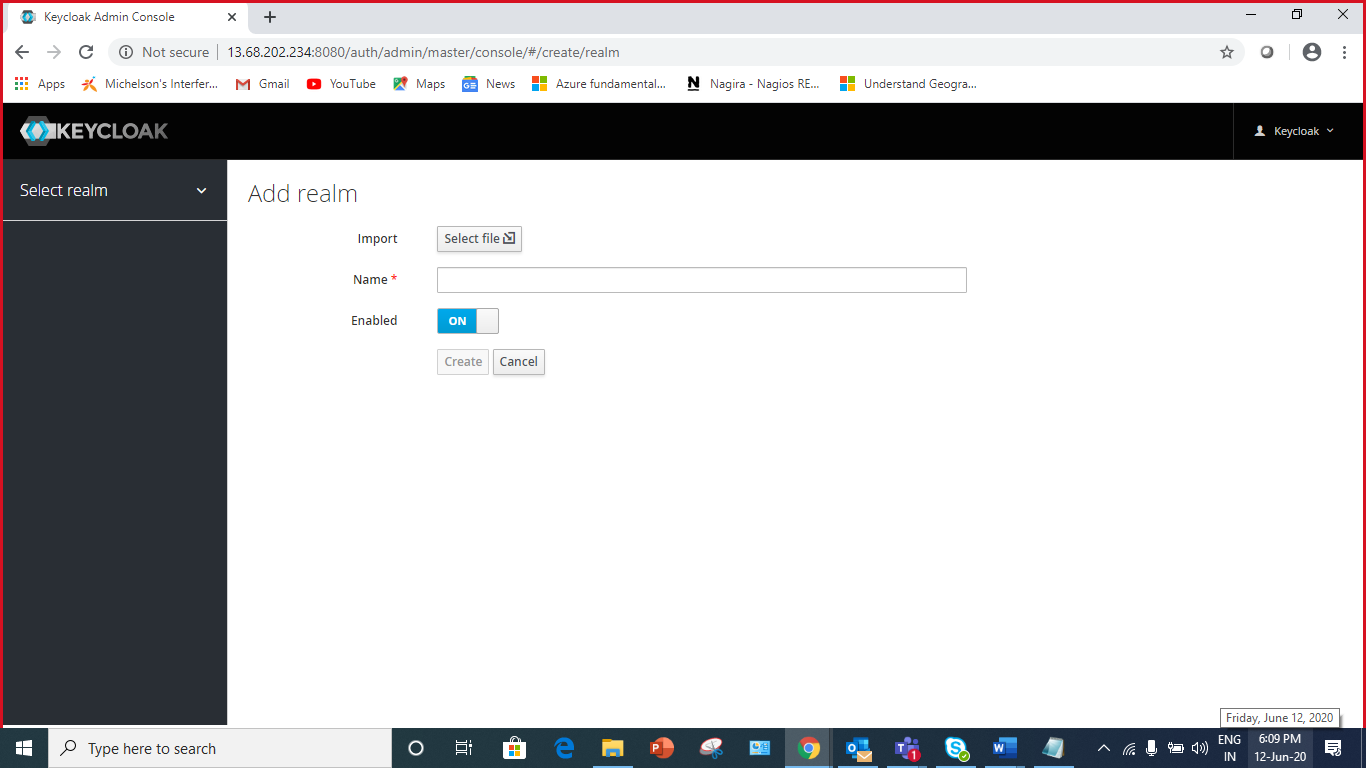
* To authorize SSL refer below command

*# ./bin/kcadm.sh update realms/master --server http:// <<private IP>>:8080/auth -s sslRequired=NONE*

* Open browser and type http://IP:8080



* Go to Admin console and login (here we used uname : keycloak pwd : keycloak)
* Create realm – mosaic



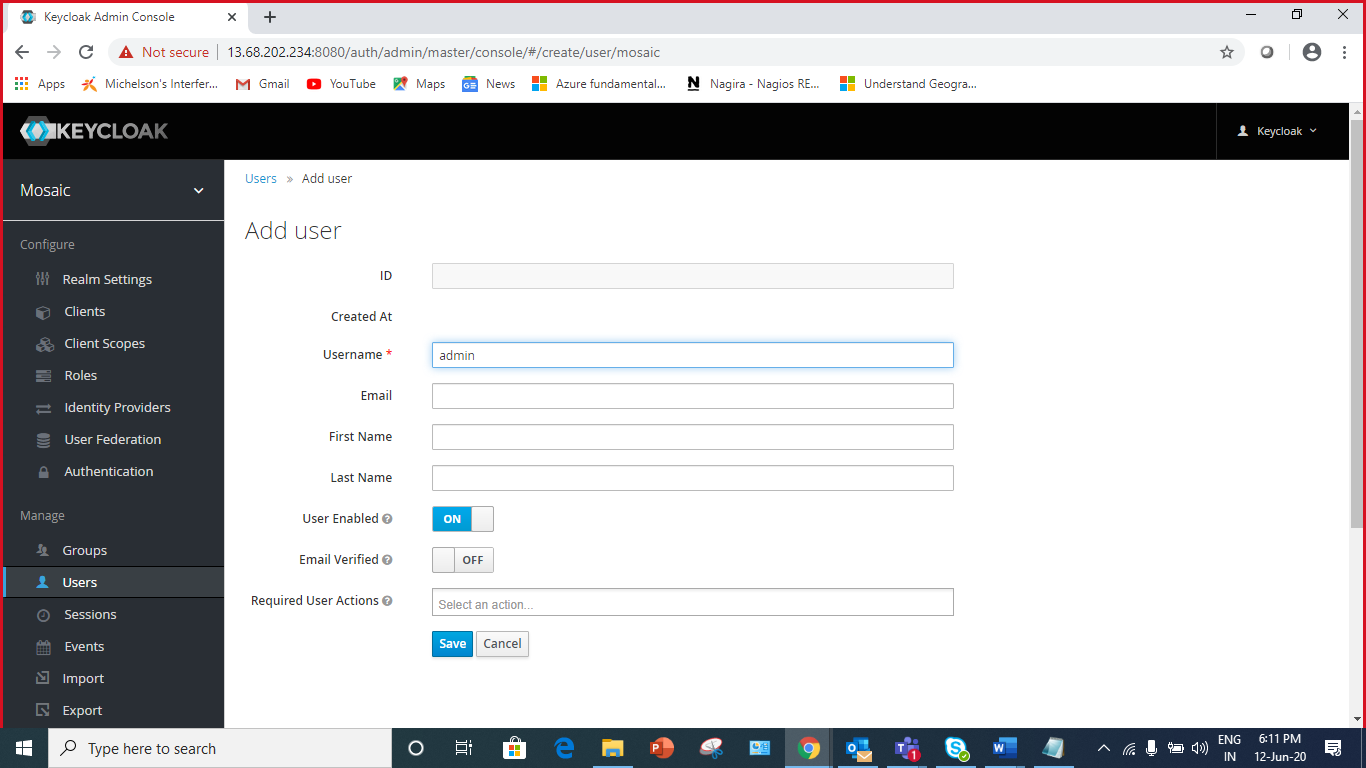
* Go to import tab Import and import realm mosaic.json file from your computer (attached herewith)



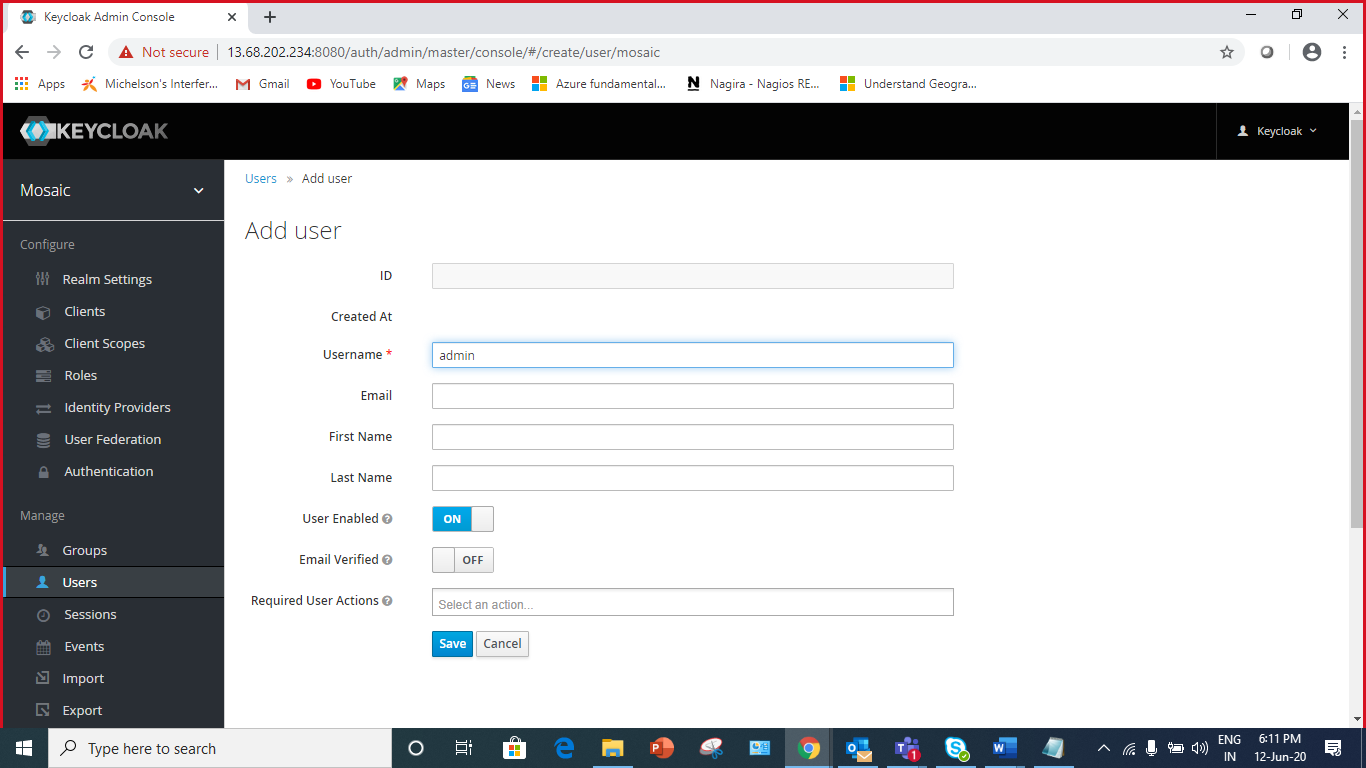
* Go to Clients > Keycloak-gatekeeper > Valid redirect URL:

Remove all previous entries and keep http://<<DNS>>/\*

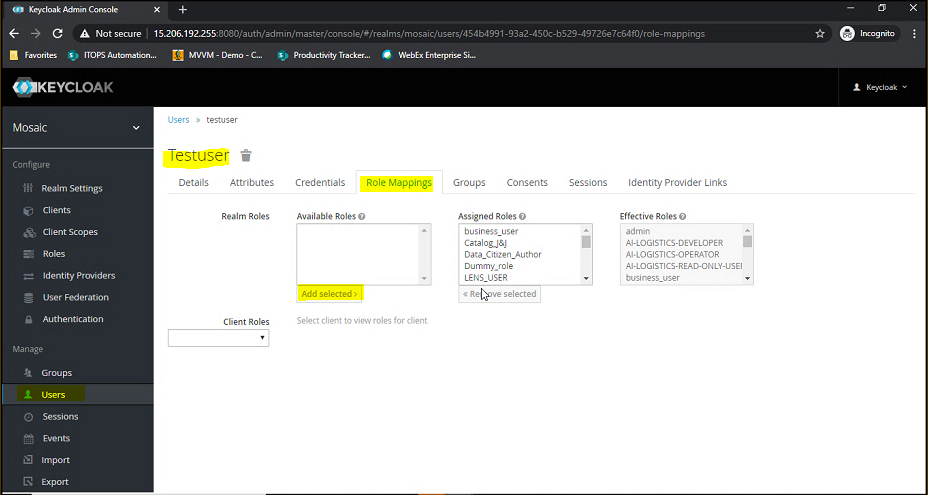
* Go to identity providers tab and delete the entry - Mosaic SSO
* Go to Users tab and create new user



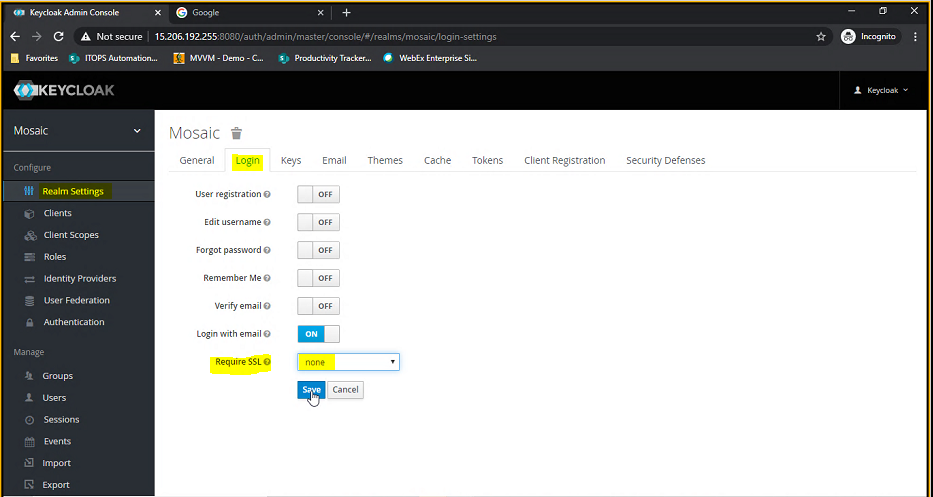
* Go to Users >> Groups : add to mosaic-aiops and mosaic



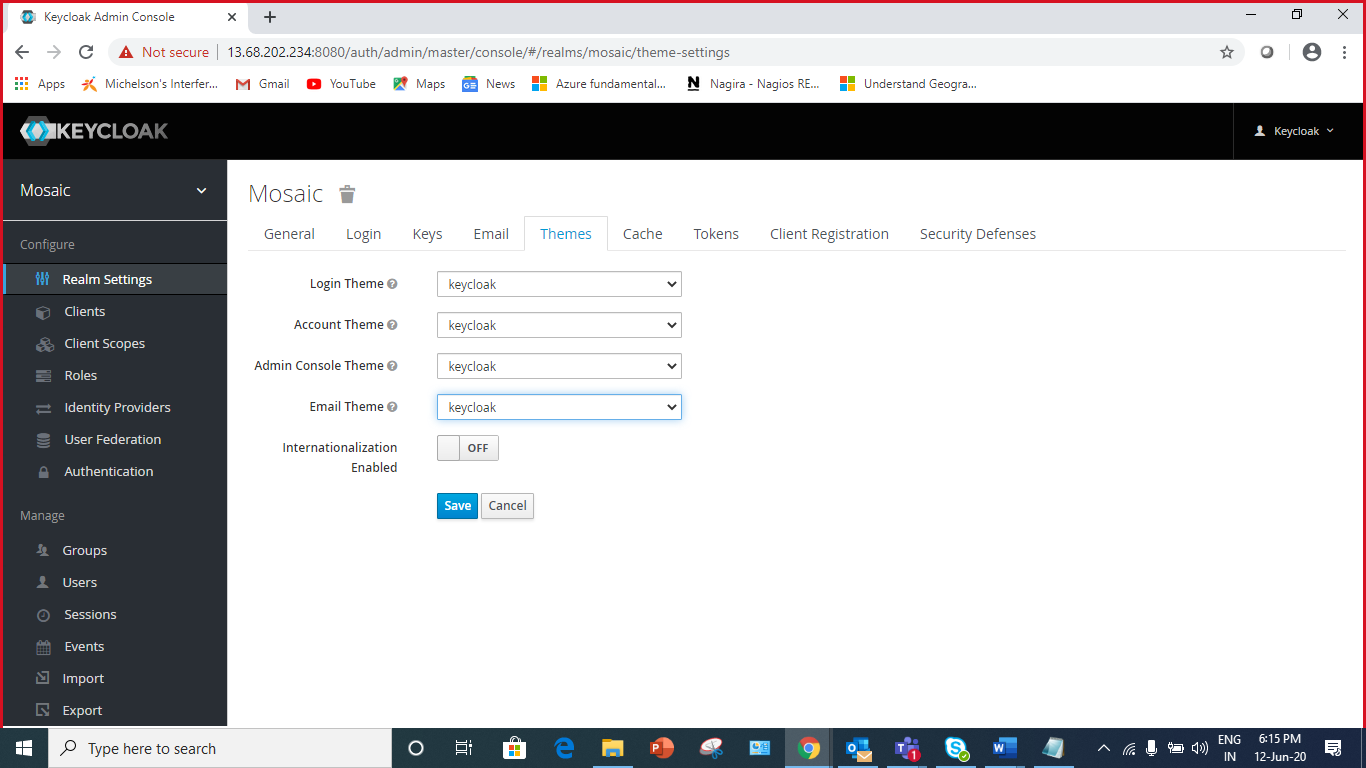
* Go to users>>role mappings>>give SE-LI role to user (if it is not present, create the role SE-L1 under Roles)



* Login to Keycloak go to realm setting for Mosaic realm and edit login tab >> Required SSL = none

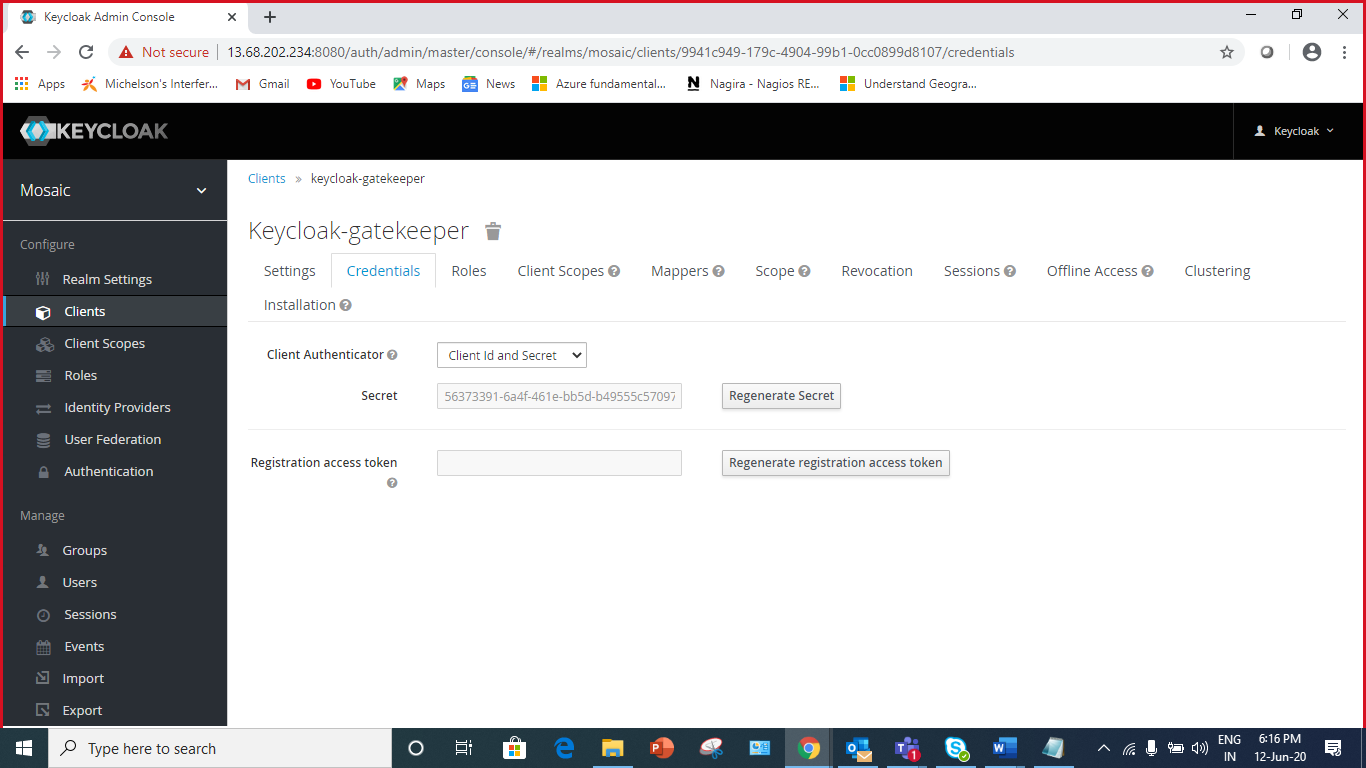


* Under Themes tab, login theme and account theme = ‘keycloak’



* Go to Authentication tab >> Required actions – check update password, update profile and verify email
* To get secret:

Go to Clients tab >> keycloak-gatekeeper >> credentials >> secret >> regenerate secret



## 

## **PostgreSQL Installation**

* To import the repository Key for PostgreSQL packages:

*# sudo apt-get install wget ca-certificates*

*# wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -*

* Adding PostgreSQL apt repository to sources

*# sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/ $(lsb\_release -sc)-pgdg main" >> /etc/apt/sources.list.d/PostgreSQL.list'*

* Update the repository

*# sudo apt-get update && sudo apt-get upgrade*

* Installation of PostgreSQL

*# sudo apt-get install postgresql-9.6*

* In order log into to use Postgres account-

*# sudo su postgres*

* Access ­­­­Postgres(default) console

*# psql*

Basic commands in PSQL-

\l: Show all the database

\du: Show user/roles

\conninfo: Connection info

\dn: List all the schemas

\dt: Show all tables

\du: List user

* Change the password of postgres user

*# ALTER USER POSTGRES WITH PASSWORD ‘yournewpassword’;*

* Creating AIOPS Database

*# Create database aiops;*

* Keep the SQL files in VM
* Inserting schemas and data in PSQL
* Move to 1\_schemas folder

*# cd 1\_schemas*

* Execute below commands

(Note - follow the below commands in sequence)

*# psql -d aiops < create-user-with-access.sql*

*# psql -d aiops < action\_manager.sql*

*# psql -d aiops < aiops\_automation.sql*

*# psql -d aiops < cognitive.sql*

*# psql -d aiops < dam.sql*

*# psql -d aiops < email\_parser.sql*

*# psql -d aiops < mosaic\_parser.sql*

*# psql -d aiops < notification.sql*

*# psql -d aiops < public.sql*

*# psql -d aiops < rule\_engine.sql*

*# psql -d aiops < scheduler\_service.sql*

* Move to 2\_cmdb\_insert folder

*#cd 2\_cmdb\_insert*

* Execute below commands

*# psql –d aiops < aiops\_cmdb.sql*

* Move to 3\_other\_insert folder

*# cd 3\_other\_insert*

* Execute below commands

(Note - follow the below commands in sequence)

*# psql -d aiops < action\_manager\_data.sql*

*# psql -d aiops < aiops\_automation\_data.sql*

*# psql -d aiops < cmdb\_data.sql*

*# psql -d aiops < cognitive\_data.sql*

*# psql -d aiops < dam.sql*

*# psql -d aiops < email\_parser\_data.sql*

*# psql -d aiops < mosaic\_parser\_data.sql*

*# psql -d aiops < notification\_data.sql*

*# psql -d aiops < public\_data.sql*

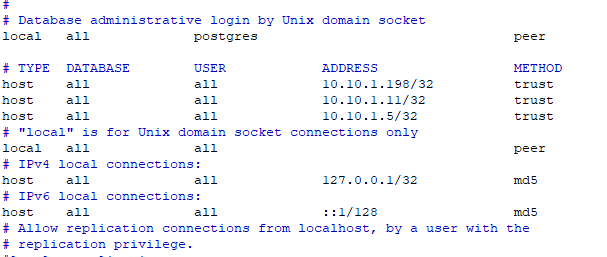
*# psql -d aiops < rule\_engine\_data.sql*

* PostgreSQL configurations: For nodes connection

*# sudo vi /etc/postgresql/9.6/main/pg\_hba.conf*

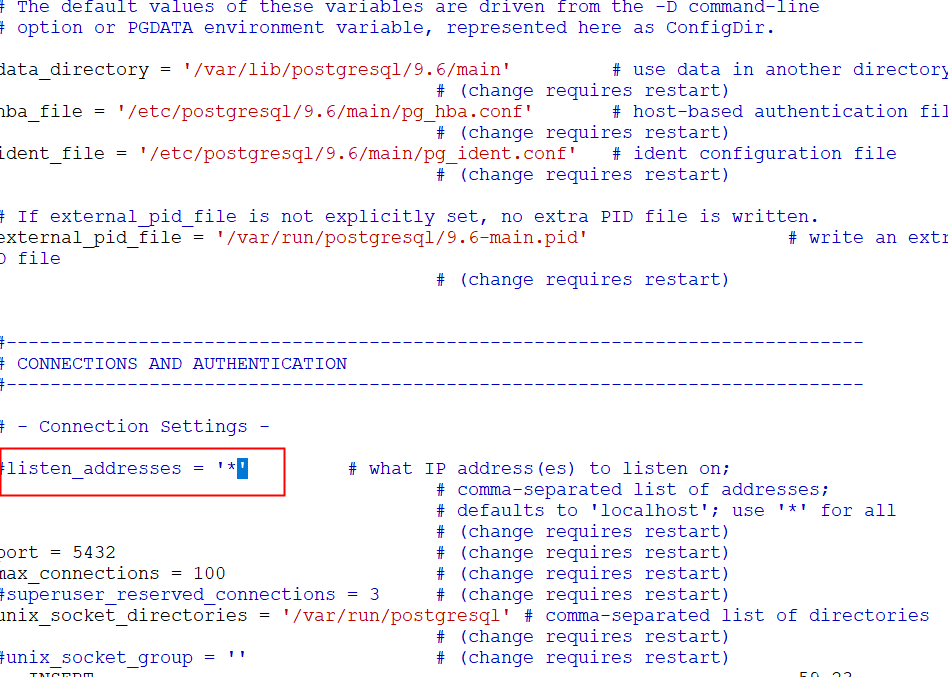
Here, in this file we need to add Master, Worker1 and Worker 2 information.

(These changes are made to access psql database from pods)

In the fields: Type, Database, User, Address and Method

* Then, go to postgresql.conf file to edit the listen addresses= local host into \* (This specifies that it will listen to all other ip address)

*# sudo vi /etc/postgresql/9.6/main/postgresql.conf*



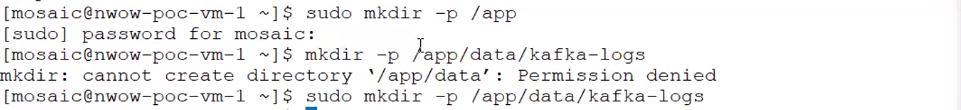
## **Kafka Configuration**

**VM1-**

* Create /app directory

*# sudo mkdir -p /app*

* Create kafka-logs directory under /app

*# sudo mkdir -p /app/data/kafka-logs*

* Give permission on /app directory

*# sudo chown mosaic:mosaic -R /app*

* Move inside /app direcotry

*# cd /app*

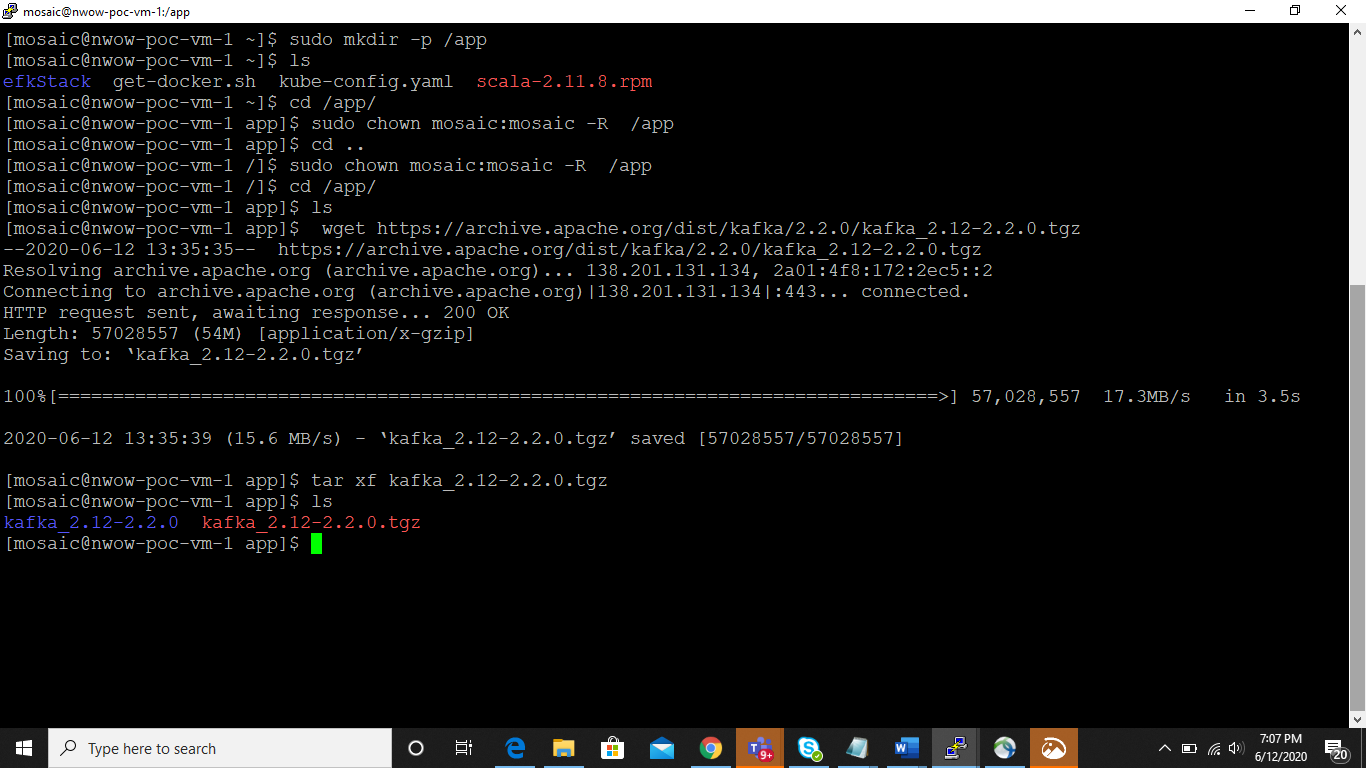


* Download Kafka binaries

*# wget* [*https://archive.apache.org/dist/kafka/2.2.0/kafka\_2.12-2.2.0.tgz*](https://archive.apache.org/dist/kafka/2.2.0/kafka_2.12-2.2.0.tgz)

* Extract the Kafka binaries

*# tar xf kafka\_2.12-2.2.0.tgz*

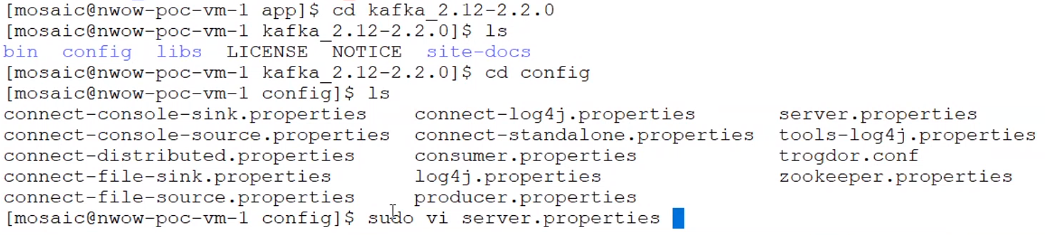


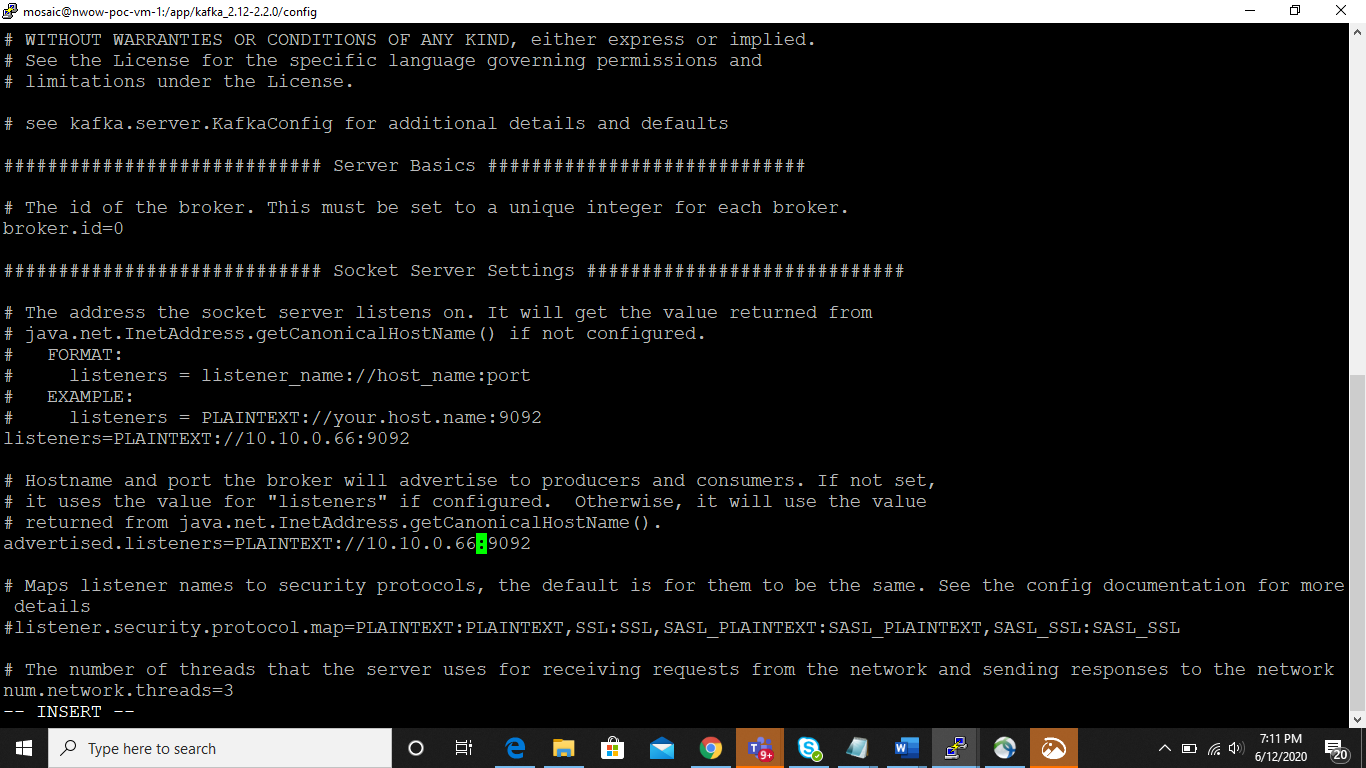
* Copy kafka binaries to other VM’s

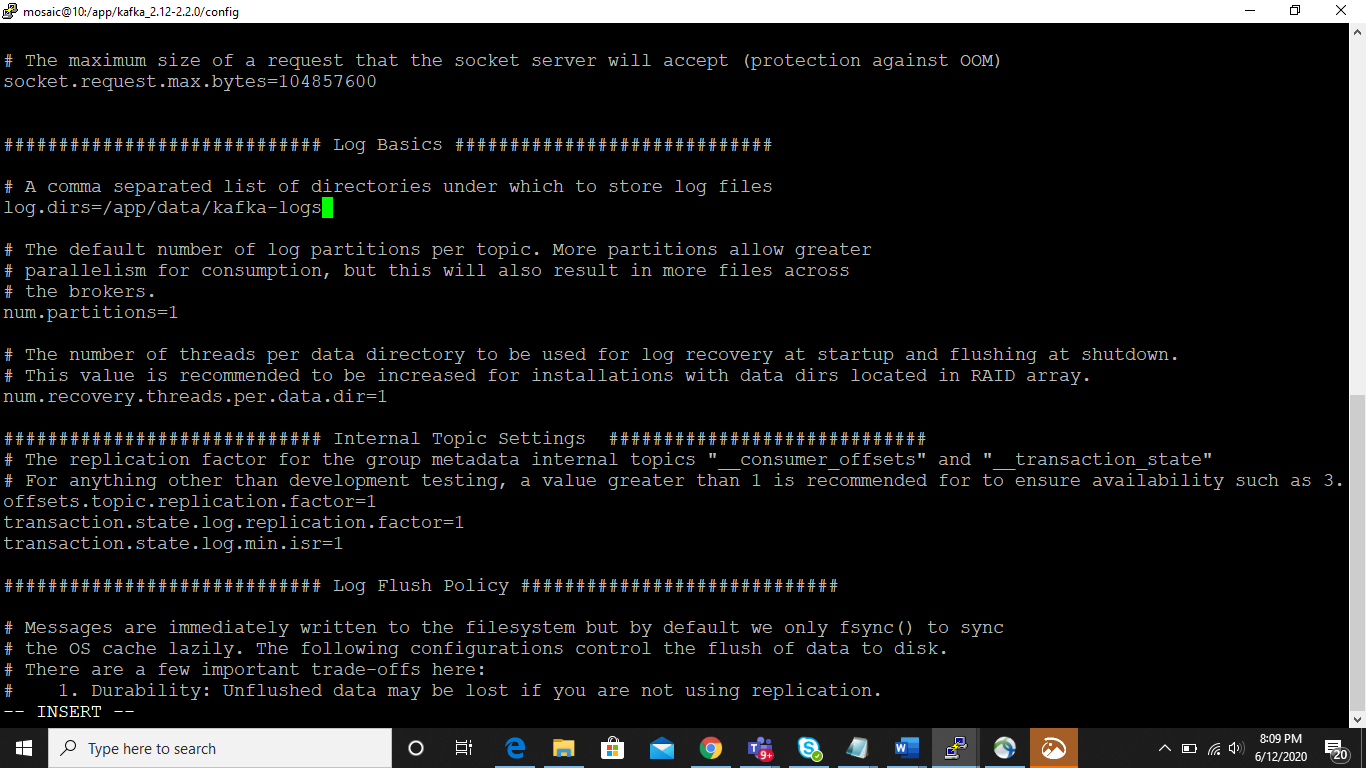
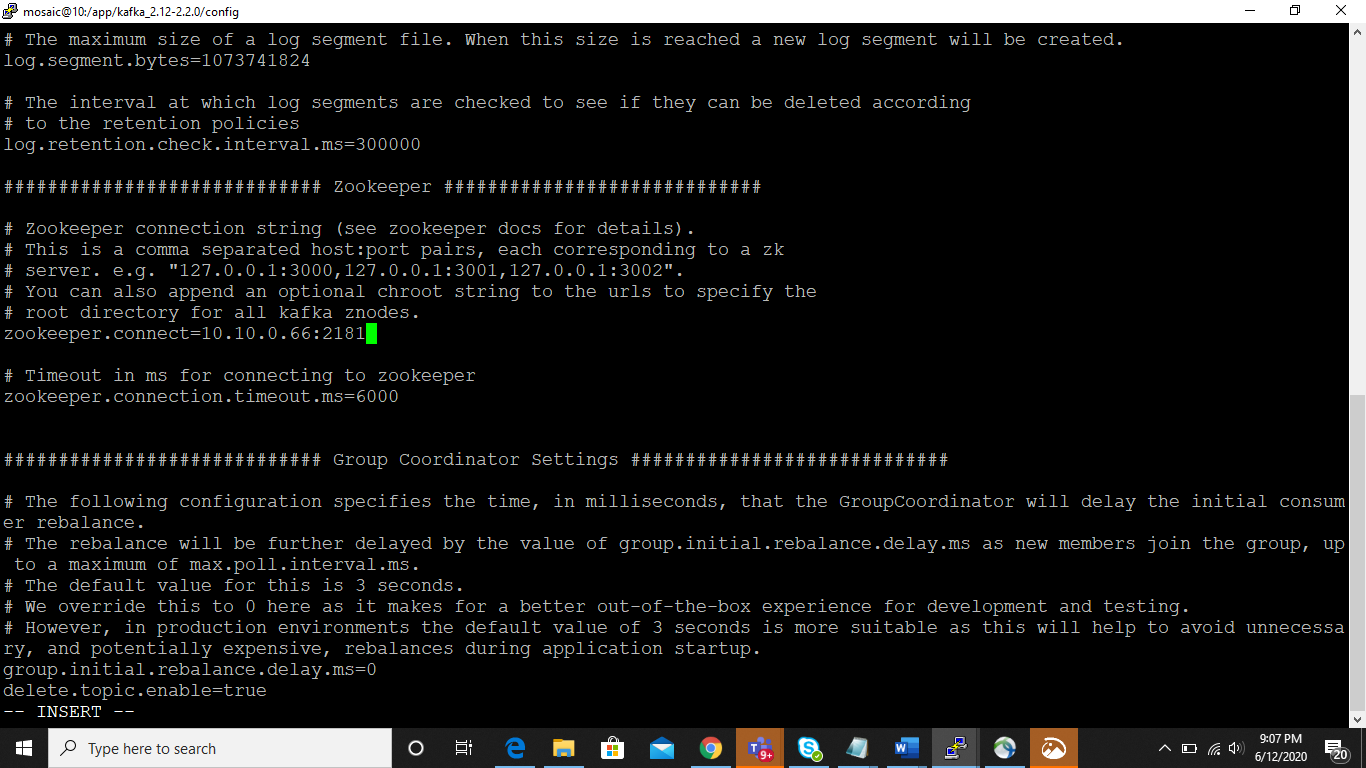
*# scp -r kafka\_2.12-2.2. <<private ip of other VM>>:/app (first create directories in VM-2 and 3 then copy)*

*#scp -r kafka\_2.12-2.2. <<private ip of other VM>> :/app*

**VM1, VM2, VM3-**

* Open “server.properties” file using vi editor by being inside the path kafka\_2.12-2.2.0/ config/
* zookeeper.connect=<<ip>>:2181 (uncomment this line and this will remain same in all servers and it will be private ip )
* Here ip – is the internal/private ip of the master node server where the zookeeper lies.
* broker.id= 0 (This value will change by each node 0,1,2 for all three nodes.)
* log.dirs = /app/data/kafka-logs (Here mention the directory path which you have created for storing kafka logs).
* advertised.listeners=PLAINTEXT://<<ip>>:9092 (Here ip value is dynamic on all three servers Private IP of individual server must be mentioned here for this key.)
* listeners=PLAINTEXT://<<ip>>:9092 (This value is dynamic on all three servers Private IP of individual server must be mentioned here for this key.)



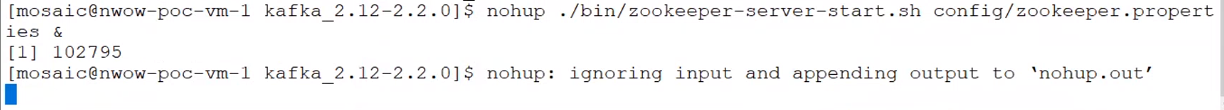


* For both advertised listeners and listeners ip must be given according to the nodes as described below.

i.e., if you are changing the file in master node then that master node’s ip has to be mentioned in the place of ip (<<ip>>) and if you are changing the file in worker node then that worker node’s ip has to be mentioned here.

* Make these changes in server.properties file and save the file in all three nodes accordingly (master, worker1, worker2)
* Starting Zookeper in vm1-

*# nohup ./bin/zookeeper-server-start.sh config/zookeeper.properties &*

**

* Starting Kafka Server in all VMs

*# nohup bin/kafka-server-start.sh config/server.properties 1>kafka1.log 2>>kafka2.log &*

* To list all the brokers/worker nodes

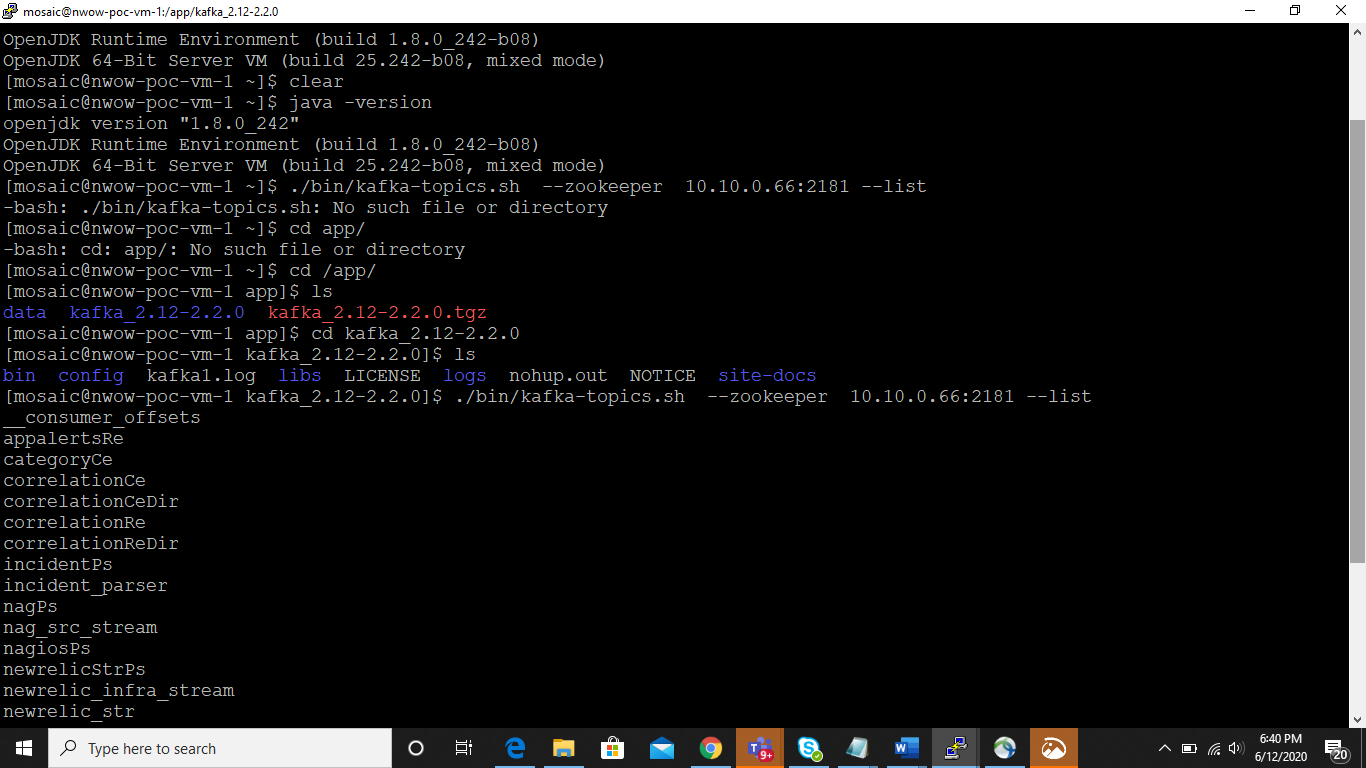
*# ./bin/zookeeper-shell.sh localhost:2181 ls /brokers/ids*

* To create topics

*# ./bin/kafka-topics.sh --create --zookeeper <<zookeeper server ip>>:2181 --replication-factor 3 --partitions 1 --topic topic-name*

**

* To list the topics

*# ./bin/kafka-topics.sh --zookeeper <<zookeeper server ip>>:2181 –list*

1. **Docker & Kubernetes Installation and Configuration**

## **The following steps need to be executed on Master as well Worker nodes**

* To get the Docker gpg key

*#curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add*

* To add the Docker repository

*# sudo add-apt-repository "deb [arch=amd64]*[*https://download.docker.com/linux/ubuntu*](https://download.docker.com/linux/ubuntu) *$(lsb\_release -cs) stable"*

**Note:** If you get bionic-release error, then

*# sudo apt install ca-certificates*

*# sudo update ca-certificates*

*# sudo add apt-repository ppa:some/ppa*

*# sudo apt-get install software-properties-common*

*# sudo apt-get update*

* Update all the packages

*# sudo apt-get update*

* To get the Kubernetes gpg key

*#curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -*

* Add the Kubernetes repository

*# cat << EOF | sudo tee /etc/apt/sources.list.d/Kubernetes.list*

*>deb* [*https://apt.kubernetes.io/*](https://apt.kubernetes.io/) *kubernetes-xenial main*

*>EOF*

### Installation of Docker-ce, Kubeadm, Kubectl, Kubelet with version

*# sudo apt-get install -y docker-ce=18.06.1~ce~3-0~ubuntu kubelet=1.13.5-00 kubeadm=1.13.5-00 kubectl=1.13.5-00*

* To check Docker, Kubeadm, Kubelet and Kubectl version

*# sudo docker –v*

*# sudo kubeadm version*

*# sudo kubectl version*

*# sudo kubelet --version*

* Hold them at the current version

*#sudo apt-mark hold docker-ce kubelet kubeadm kubectl*

### To enable Iptables, Add the iptables rule to sysctl.conf

*# echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf*

* To enable iptables immediately

*# sudo sysctl -p*

* **The following steps need to be executed only on Master node**

### 

### To initialize the cluster

*# sudo kubeadm init --pod-network-cidr=10.244.0.0/16 (Copy the kubeadm join command to add worker to master node)*

### To setup local kubeconfig

*# mkdir -p $HOME/.kube*

*# sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config*

*# sudo chown $(id -u):$(id -g) $HOME/.kube/config*

### To apply Flannel CNI network overlay

*# kubectl apply -f* [*https://raw.githubusercontent.com/coreos/flannel/a70459be0084506e4ec919aa1c114638878db11b/Documentation/kube-flannel.ym*](https://raw.githubusercontent.com/coreos/flannel/a70459be0084506e4ec919aa1c114638878db11b/Documentation/kube-flannel.ym)*l*

* **The following steps need to be executed only on worker nodes**

### To join the worker nodes to the cluster

For example-

*# sudo kubeadm join 172.20.68.19:6443 --token****bo1x1y.u1f2tv1mefib5zsc****--discovery-token-ca-cert-hash sha256:83f7a11ac1dd477b159330fb495cd2efe2b1aa83583027133a45753efd5b6fce*

* To see the token which has been initialized in master

*# kubectl tokens list*

* If by some chance you misplaced the kubeadm join command, you generate another one on the master node by running the below command

*# kubeadm token create --print-join-command*

* **The following step need to be executed on Master Node**

### To verify whether worker nodes joined in the cluster

*# kubectl get nodes*

# 

# **YAML FILE CONFIGURATIONS:**

# **Creating Namespaces**

* Create 2 Namespaces:

1.ingress-nginx

2.Keycloak-gatekeeper

* Create Namespace “ingress-nginx”

*# kubectl create namespace ingress-nginx*

* Create Namespace “keycloak-gatekeeper”

***#*** *kubectl create namespace keycloak-gatekeeper*

* Check the created Namespaces

*# kubectl get ns*

* **Applying aiops-secrets**
* Go inside aiops-secrets directory.

/home/automation/K8-LAB-AIOPS/aiops-secrets/

*# cd /home/automation/K8-LAB-AIOPS/aiops-secrets/*

* Check the files present inside the directory

*# ls*

There is only 1 file i.e., secrets.yaml.

No changes required here.

* Apply the secret

Make sure you are inside the aiops-secretsdirectory

*# kubectl apply -f .*

Or

* To create secrets

*# kubectl create secret docker-registry docker-credentials --docker-*

*server=registry.lti-aiq.in:443 --docker-username=rahul --docker-*

*password=3efvA2ehRFng\_u2yU2gQ**-n keycloak-gatekeeper*

*# kubectl create secret docker-registry docker-credentials --docker-*

*server=registry.lti-aiq.in:443 --docker-username=rahul --docker-*

*password=3efvA2ehRFng\_u2yU2gQ*

* To verify secrets

*# kubectl get secrets*

*# kubectl get secrets –n keycloak-gatekeeper*

* To delete secret

*# kubectl delete secretname*

*# kubectl delete secretname –n keycloak-gatekeeper*

* **Applying aiops-k8-ingress**
* Go inside aiops-k8-ingress directory.

/home/automation/K8-LAB-AIOPS/aiops-k8-ingress/

*# cd /home/automation/K8-LAB-AIOPS/aiops-k8-ingress/*

* Check the files present inside that directory

*# ls*

There are 3 k8-ingress files.

No changes required here for any files.

* Apply Ingress

Make sure you are inside the aiops-k8-ingress directory, perform below

command:

*# kubectl apply -f .*

* **Applying aiops-services**
* Go inside aiops-services directory.

/home/automation/K8-LAB-AIOPS/aiops-services/

*# cd /home/automation/K8-LAB-AIOPS/aiops-services/*

* Check the files present inside that directory

*# ls*

There are 11 service files.

No changes required here for any files.

* To Apply Services

Make sure you are inside the aiops-services directory, perform below

command:

*# kubectl apply -f .*

* **Modifying and applying aiops-keycloak**
* Go inside aiops-keycloak directory.

/home/automation/K8-LAB-AIOPS/aiops-keycloak/

*# cd /home/automation/K8-LAB-AIOPS/aiops-keycloak*/

* Check the files present inside the directory

*# ls*

There are 2 keycloak directories

1.Keycloak-gatekeeper.

2. User management.

* Go inside keycloak-gatekeeper directory.

/home/automation/K8-LAB-AIOPS/aiops-keycloak/keycloak-gatekeeper/

*# cd /home/automation/K8-LAB-AIOPS/aiops-keycloak/keycloak-*

*gatekeeper/*

* Check the files present inside the directory

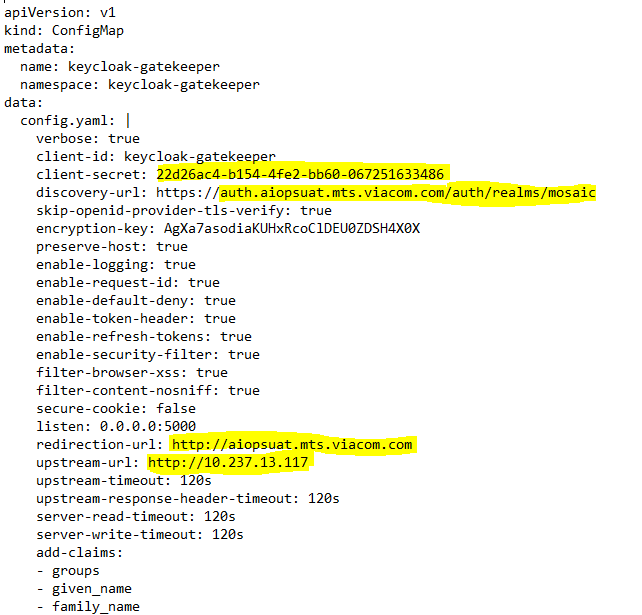
*# ls*

There are 6 keycloak-gatekeeper files.

* Configmap

go inside the file and edit

*# sudo vi configMap.yaml*



DNS name: **http://<<DNS>>**

Discovery-url: [<<Public IP of Database VM>>:8080](file:///C:/Users/10662070/Documents/15.206.192.255:8080)/auth/realms/mosaic

nginx ingress service/svc IP: <http://10.111.89.248>

# kubectl get svc -n ingress-nginx

Secret regenerated from Keycloak

Changes in above file:

enable-token-header: false

enable-security-filter: false

filter-browser-xss: false

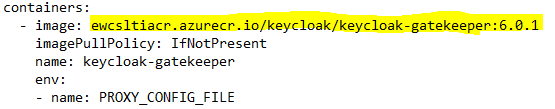
filter-content-nosniff: false

Save changes and proceed to next file.

* gatekeeper-deployment

go inside the file and edit

*# sudo vi gatekeeper-deployment.yaml*



New image link given by Rahul.

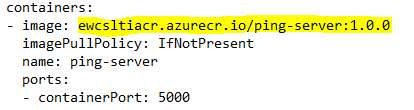
* Save changes and proceed to next file.
* ping-ing

No changes required here for this file.

* ping-server-deployment

go inside the file and edit

*# sudo vi ping-server-deployment.yaml*



New image link given by Rahul.

* Save changes and proceed to next file.
* Ping-server-service

No changes required here for this file.

* Service

go inside the file and edit

*# sudo vi service.yaml*

---

apiVersion: v1

kind: Service

metadata:

namespace: keycloak-gatekeeper

name: keycloak-gatekeeper

spec:

ports:

- nodePort: 30799

port: 80

protocol: TCP

targetPort: 5000

selector:

app: keycloak-gatekeeper

sessionAffinity: None

Newly added lines

type: NodePort

* Save changes and proceed to next steps.
* Apply keycloak-gatekeeper

Make sure you are inside the keycloak-gatekeeper directory, perform

below command:

*# kubectl apply -f .*

* To check all services in keycloak-gatekeeper namespace

Once above module finished, perform below command to map Keycloak-

gatekeeper svc port from cluster nodes to created LB.

*# kubectl get svc -n keycloak-gatekeeper*



* Provide highlighted port number to infra team.
* You will receive New DNS attached to Load balancer name. Note down DNS

attached to LB name and perform required steps further.

* Go inside user management folder.

/home/automation/K8-LAB-AIOPS/aiops-keycloak/usermanagement/

*# cd /home/automation/K8-LAB-AIOPS/aiops-keycloak/usermanagement/*

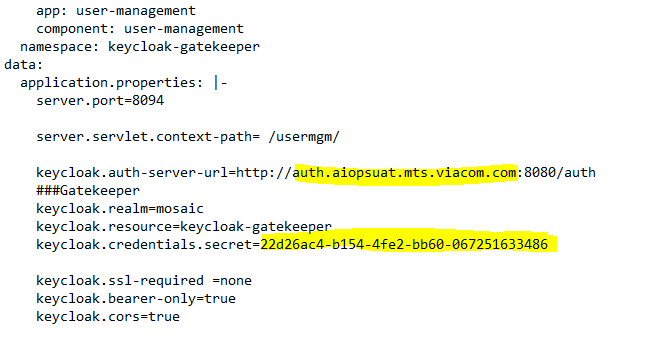
* Check the files present inside that directory

*# ls*

There is only one file i.e., iam.yaml file.

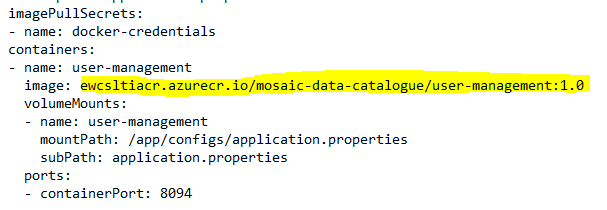
* Follow the below instructions for making required changes in file.

*# sudo vi iam.yaml*

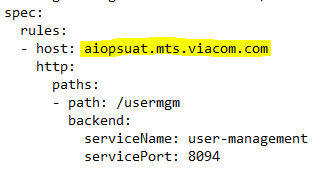


DB server public IP

Secret regenerated in keycloak



New user management image link provided



<<DNS>> provided providedprovided

Save changes.

Once done with change you can apply that file.

* To apply user management

Make sure you are inside the user management directory, perform below

*# kubectl apply -f iam.yaml*

* **Modifying and Applying aiops-configmaps**
* Go inside aiops-configmaps directory.

/home/automation/K8-LAB-AIOPS/aiops-configmaps/

# cd /home/automation/K8-LAB-AIOPS/aiops-configmaps/

* Check the files present inside that directory

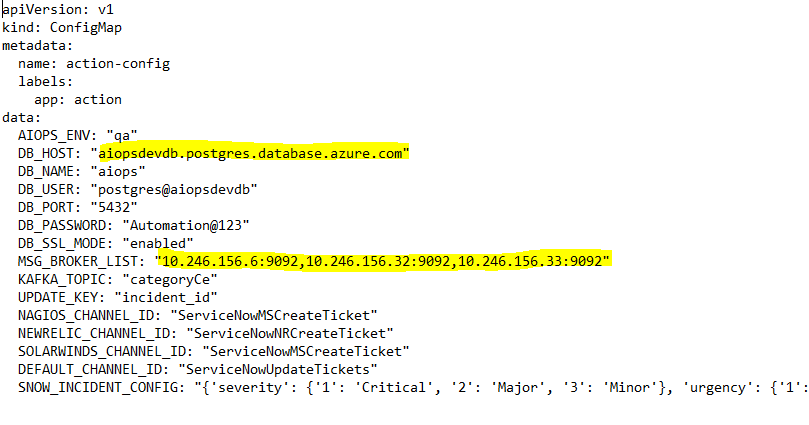
*# ls*

There are 10 configmap files.

* action\_config

To go inside the file and edit

*# sudo vi action\_config.*

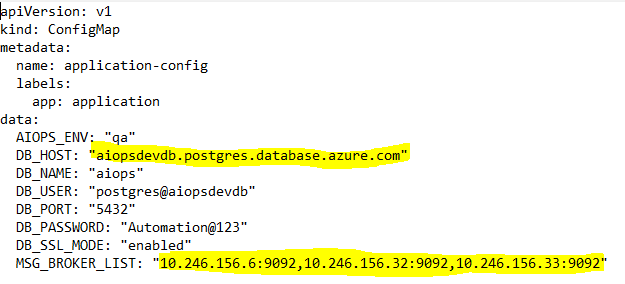


Private IP’s of Kafka VM’s

DB Server private IP

* Also change DB name, DB user and DB password if needed .
* Save changes and proceed to next file.
* application-config

*# sudo vi application-config.yaml*



Private IP’s of Kafka VM’s

DB Server private IP

* Same changes need to be done in the below files (3, 4, 5, 6, 7, 8)
* cmdb\_config
* cognitive-config

*# sudo vi cognitive-config.yaml*

apiVersion: v1

kind: ConfigMap

metadata:

name: cognitive-config

labels:

app: cognitive-service

data:

AIOPS\_ENV: "qa"

DB\_HOST: "10.10.1.253"

DB\_NAME: "aiops"

DB\_USER: "postgres"

DB\_PORT: "5432"

DB\_PASSWORD: "Automation@123"

DB\_SSL\_MODE: "enabled"

MSG\_BROKER\_LIST: "10.10.1.198:9092,10.10.1.11:9092,10.10.1.5:9092"

KAFKA\_PARTITIONS: "1"

Newly added variables

NUM\_WORKERS: "17"

PYTHONHASHSEED: "

* parser-config

*# sudo vi parser-config.yaml*

apiVersion: v1

kind: ConfigMap

metadata:

name: parser-config

labels:

app: parser

data:

AIOPS\_ENV: "qa"

DB\_HOST: "10.10.1.253"

DB\_NAME: "aiops"

DB\_USER: "postgres"

DB\_PORT: "5432"

DB\_PASSWORD: "Automation@123"

DB\_SSL\_MODE: "enabled"

MSG\_BROKER\_LIST: "10.10.1.198:9092,10.10.1.11:9092,10.10.1.5:9092"

nagios\_timezone: "EST"

solarwinds\_timezone: "EST"

newrelic\_timezone: "UTC"

snow\_timezone: "UTC"

nagios\_format: "%H:%M:%S-%m/%d/%Y"

solarwinds\_format: "%m/%d/%Y %I:%M %p"

Newly added variables

snow\_format: "%Y-%m-%d %H:%M:%S"

new\_relic\_format: "%Y-%m-%d %H:%M:%S.%f %Z%z"

alert\_staging\_flag: "False"

alert\_rejected\_flag: "True"

TOPIC\_DAM: "ch\_dam"

NUM\_WORKERS: "17"

* re\_config

*# sudo vi re\_config.yaml*

apiVersion: v1

kind: ConfigMap

metadata:

name: ruleengine-config

labels:

app: ruleengine

data:

AIOPS\_ENV: "qa"

DB\_HOST: "10.10.1.253"

DB\_NAME: "aiops"

DB\_USER: "postgres"

DB\_PORT: "5432"

DB\_PASSWORD: "Automation@123"

DB\_SSL\_MODE: "enabled"

MSG\_BROKER\_LIST: "10.10.1.198:9092,10.10.1.11:9092,10.10.1.5:9092"

THREADS\_PARTITION: "5"

Newly added variable

NUM\_WORKERS: "17"

* database\_service\_config
* email-config
* connector-config

*# sudo vi connector-config.yaml*



Private IP’s of Kafka VM’s

DB Server private IP

SNOW\_UPDATE\_POST: "true"

Newly added variable

NUM\_WORKERS: "17"

* Save changes and proceed to next file.
* frontend-config

# sudo vi frontend-config.yaml



DNS name: **<<DNS>>**

DNS attached to LB: **<<DNS>>**

DNS name: **<<DNS>>**



<<Private IP address of DB server>>:8080

Secret regenerated from keycloak

* Save changes and proceed to next file.
* To apply configmaps

Make sure you are inside the aiops-configMaps directory, perform below

command:

*# kubectl apply -f .*

* **Modifying and Applying aiops-deployments**
* Go inside aiops-deployments directory.

/home/automation/K8-LAB-AIOPS/aiops-deployments/

*# cd /home/automation/K8-LAB-AIOPS/aiops-deployments/*

* Check the files present inside that directory

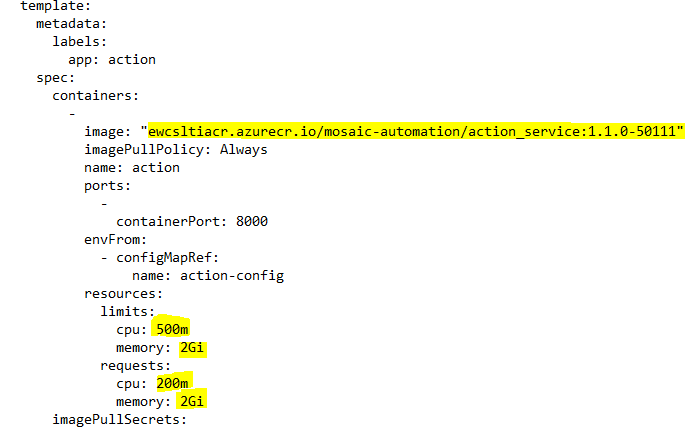
*# ls*

There are 10 deployment files.

* Follow the below instructions for making required changes in each file.
* action-deployment

go inside the file and edit

*# sudo vi action-deployment.yaml*



Provide the maximum resources for limits,

Minimum resources for requests

New Link of image provided

Save changes and proceed to next file.

Same changes need to be done in all the other files mentioned below.

* Application-deployment
* Cmdb-deployment
* Cognitive-deployment
* Connector-deployment
* Database-deployment
* Email-parser-deployment
* Frontend-deployment
* Parser-deployment
* Rule-engine-deployment

* To apply deployments

Make sure you are inside the aiops-deployments directory, perform be;low command

*# kubectl apply -f .*

* **Modifying and Applying aiops-svc-ingress**
* go inside aiops-ingress directory.

/home/automation/K8-LAB-AIOPS/aiops-ingress/

*# cd /home/automation/K8-LAB-AIOPS/aiops-ingress/*

* To check the files present inside that directory

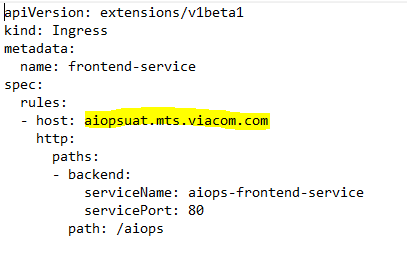
*# ls*

There are 5 ingress files.

* Aiops-ing

To go inside the file and edit

*# sudo vi aiops-ing.yaml*



DNS name: **<<DNS>>**

Save changes and proceed to next file.

Same changes need to be done in all the other files mentioned below.

* Application-ing
* Cmdb-ing
* Connector-ing
* Rewrite
* To apply ingress

Make sure you are inside the aiops-ingress directory, perform below

command:

*# kubectl apply -f .*

1. **Post installation Checks**

****

# **Backup and Restore**

# **MySQL Backup**

* **Backup**

***#*** *mysqldump ---user <<username>> ---password=<<password>> keycloak > keycloak.dump or keycloak.sql or keycloak.gz*

* **Restoring**

***#*** *mysql -u <<username>> -p keycloak < keycloak.dump or keycloak.sql or keycloak.gz*

## **PostgreSQL Backup**

* **Backup**

*# pg\_dump aiops > aiops.bak*

* **Restoring**

*#**psql aiops < aiops.bak*

## **Kafka Backup**

* **Backup:** The Zookeeper state data will be stored in /tmp/zookeeper

*# tar -czf /home/kafka/zookeeper-backup.tar.gz /tmp/zookeeper/\**

The Topics and the messages are stored in /app/data/kafka-logs

*# tar -czf /home/kafka/kafka-backup.tar.gz /app/data/kafka-logs/\**

* **Restoring:** To restore the zookeeper state

*# tar -C /tmp/zookeeper -xzf /home/kafka/zookeeper-backup.tar.gz --strip-components*

To restore the topics and messages

*# tar -C /app/data/kafka-logs -xzf /home/kafka/kafka-backup.tar.gz --strip-components*

# **Removal of the existing deployment**

## **Remove Java In all the VMs**

*# sudo apt remove java\**

## **Remove MySQL**

*# Mysql=> drop database keycloak;*

*# sudo apt remove mysql\**

*# sudo rm -rf /var/lib/mysql/*

*# sudo rm -rf /var/log/mysql/*

*# sudo rm -rf /etc/mysql/*

## **Remove keycloak**

*# sudo rm -rf /app*

*#* sudo vi .bashrc content is removed(only keycloak env data)

## **Remove Postgresql**

*# psql-> drop database aiops;*

*# sudo apt remove postgresql*

*# sudo rm -rf /var/lib/postgresql/*

*# sudo rm -rf /var/log/postgresql/*

*# sudo rm -rf /etc/postgresql/*

## 

## **Remove kafka**

*# cd /app/kafka*

*# ./bin/kafka-topics.sh --delete --zookeeper 10.10.0.66:2181 –topic <topic name>*

*# sudo rm –rf /app/data*

*# sudo rm -rf /app*

## **Remove Kubernetes configurations**

*# kubectl delete ns ingress-nginx*

*# kubectl delete ns keycloak-gatekeeper*

*# kubectl delete deployments –all*

*# kubectl delete svc –all*

*# kubectl delete ingress –all*

*# kubectl delete configmaps –all*

## **Remove Docker, Kubernetes**

*# sudo rm –rf .kube*

*# sudo apt remove kubectl*

*# sudo apt remove docker*

*# sudo apt remove kubeadm*

*# sudo apt remove kubelet*

# **Conclusion**

This document summarizes the overall implementation steps of AIOPS IAAS deployment(ubuntu) one could refer in future. Along with the implementation plan, pre checks and post check it also consists a full-fledged backup plan for the same.