**Playbook for Setting up EFK Stack for Kubernetes Monitoring**

The high-level steps that are required to setup a Linode cluster and then install an application to validate the Kubernetes security monitoring is given below .

1. Set up a free docker hub private repository to push the test java and node application images
2. Create the Kubernetes cluster on Linode Kubernetes engine
3. This step is optional. Deploy a Kubernetes dashboard in the cluster
4. Deploy the test applications node and java app images from docker registry into the cluster
5. Deploy the ElasticSearch using helm
6. Deploy the Kibana using helm
7. Deploy the FluentD with helm
8. Configure the FluentD log collector to process container logs and send them to elastic

The detailed steps are given below

**Setting up docker hub registry :**

Setup your free Docker registry using the documentation from Dcker

<https://docs.docker.com/registry/configuration/>

Once the registry is setup , use the following commands on your local machine to connect to your registry

*DCKR\_REG\_SRVR=docker.io*

*DCKR\_USR=your dockerID, same ID as the user’s `docker login`*

*DCKR\_EMAIL=your dockerhub email, same password as user’s `docker login email `*

*DCKR\_PASSWD=your dockerhub pwd, same as for `docker login`*

*kubectl create secret docker-registry myregistrysecret \*

*--docker-server=$ DCKR\_REG\_SRVR \*

*--docker-username=$ DCKR\_USR \*

*--docker-password=$ DCKR\_PASSWD \*

*--docker-email=$ DCKR\_EMAIL*

**Installing Kubernetes Cluster in Linode Kubernetes Engine**

The Kubernetes cluster can be installed in the Linode from the GUI with a series of self-explanatory steps . Once cluster is installed , install the kubectl dashboard with the steps.

Use the linode-values.yaml file for the linode Kubernetes cluster setup

*kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.0.0/aio/deploy/recommended.yaml*

*kubectl proxy*

verify that the proxy is working on local machine with

*http://localhost:8001/api/v1/namespaces/kubernetes-dashboard/services/https:kubernetes-dashboard:/proxy/.*

Once the installation is complete, download the kubeconfig.yaml file from the cluster , which can be used later for connection .

Perform the command to export the config file to current working directory

*export KUBECONFIG=/path/to/config.yaml*

**Building Test Applications**

Build any application that can be pushed onto the Docker registry. This helps to download applications to any environment for validation . For this course we have used a node app and a java app

Download and save the node-app.zip file and unzip it to the home or root directory

Build the nide app using the command

*docker build -t node-app .*

Push the node app docker image to the docker registry

*docker tag node-app demo-app:node-1.0*

for the Java application used in this project , download and unzip the files to the root directory

Build the gradle project using

*./gradlew build*

By executing from the root ID build the Java app Docker image

*docker build -t java-app .*

Push the Java app docker image to docker registry

*docker tag java-app demo-app:java-1.0*

**Installing ES Helm Chart**

Use the previously downloaded ‘values-linode’ file for this step and then, Use the following commands from kube terminal to install elasticsearch

*helm repo add elastic https://Helm.elastic.co*

*helm install elasticsearch elastic/elasticsearch -f values-linode.yaml*

**Installing the Kibana Chart**

*helm install kibana elastic/kibana*

To access kibana locally , perform port forwarding to a local port

*kubectl port-forward deployment/kibana-kibana 5601*

*access: localhost:5601*

**Installing FluentD using Help charts**

*helm repo add bitnami https://charts.bitnami.com/bitnami*

*helm install fluentd bitnami/fluentd*