

## EFK Setup Using HELM – Kubernetes

-> EFK is a suite of tools combining Elasticsearch, Fluentd and Kibana to manage logs

-> Fluentd will collect the logs and send them to Elasticsearch. This latter will receive the logs and save it on its database

-> Kibana will fetch the logs from Elasticsearch and display it on a nice web app

**Note:** All three components are available as binaries or as Docker containers




-> We are going to download and install below 3 components in our Kubernetes Cluster

1) Elastic Search

2) Fluent Bit

3) Kibana



# ASHOK IT

*Learn Here.. Lead Anywhere..!!*

```
[ec2-user@ip-172-31-6-228 efk-demo]$ ls -l
total 64
drwxr-xr-x. 4 ec2-user ec2-user 188 Nov 12 10:08 elasticsearch
-rw-r--r--. 1 ec2-user ec2-user 27894 Mar  8 2022 elasticsearch-7.17.1.tgz
drwxr-xr-x. 5 ec2-user ec2-user 124 Nov 12 10:28 fluent-bit
-rw-r--r--. 1 ec2-user ec2-user 12765 Nov  9 21:52 fluent-bit-0.21.0.tgz
drwxr-xr-x. 4 ec2-user ec2-user 128 Nov 12 10:46 kibana
-rw-r--r--. 1 ec2-user ec2-user 10364 Apr 21 2022 kibana-7.17.3.tgz
-rw-r--r--. 1 ec2-user ec2-user 314 Nov 12 09:57 pv.yml
-rw-r--r--. 1 ec2-user ec2-user 142 Nov 12 12:20 storage-class.yml
[ec2-user@ip-172-31-6-228 efk-demo]$
[ec2-user@ip-172-31-6-228 efk-demo]$
[ec2-user@ip-172-31-6-228 efk-demo]$
```

## Elastic Search Installation

**Step-1: Create manifest file to create Storage Class with either host path or NFS Server (I am using hostpath)**

```
1  apiVersion: storage.k8s.io/v1
2  kind: StorageClass
3  metadata:
4    name: dev-storage-class
5    namespace: logging
6  provisioner: kubernetes.io/hostpath
```

**Step-2 : Create Storage Class using above manifest file**

```
total 64
drwxr-xr-x. 4 ec2-user ec2-user 128 Nov 12 13:28 elasticsearch
-rw-r--r--. 1 ec2-user ec2-user 27894 Mar  8 2022 elasticsearch-7.17.1.tgz
drwxr-xr-x. 5 ec2-user ec2-user 124 Nov 12 10:28 fluent-bit
-rw-r--r--. 1 ec2-user ec2-user 12765 Nov  9 21:52 fluent-bit-0.21.0.tgz
drwxr-xr-x. 4 ec2-user ec2-user 128 Nov 12 10:46 kibana
-rw-r--r--. 1 ec2-user ec2-user 10364 Apr 21 2022 kibana-7.17.3.tgz
-rw-r--r--. 1 ec2-user ec2-user 314 Nov 12 09:57 pv.yml
-rw-r--r--. 1 ec2-user ec2-user 142 Nov 12 12:20 storage-class.yml
[ec2-user@ip-172-31-6-228 efk-demo]$
[ec2-user@ip-172-31-6-228 efk-demo]$
[ec2-user@ip-172-31-6-228 efk-demo]$ kubectl apply -f storage-class.yml
```

**Step-3: check the storage class is created or not**

```
[ec2-user@ip-172-31-6-228 efk-demo]$ kubectl get sc
NAME                PROVISIONER             RECLAIMPOLICY   VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION   AGE
dev-storage-class   kubernetes.io/hostpath   Delete          Immediate            false                   3h36m
gp2 (default)       kubernetes.io/aws-efs     Delete          WaitForFirstConsumer false                   5h13m
[ec2-user@ip-172-31-6-228 efk-demo]$
[ec2-user@ip-172-31-6-228 efk-demo]$
[ec2-user@ip-172-31-6-228 efk-demo]$
```

**Step-4: Create Namespace ( Namespace name : logging)**

*Command To Create Namespace : \$ kubectl create namespace logging*

*Command To check Namespace : \$ kubectl get ns*

```
[ec2-user@ip-172-31-6-228 efk-demo]$
[ec2-user@ip-172-31-6-228 efk-demo]$
[ec2-user@ip-172-31-6-228 efk-demo]$
[ec2-user@ip-172-31-6-228 efk-demo]$ kubectl get ns
NAME                STATUS    AGE
default             Active    5h16m
kube-node-lease     Active    5h16m
kube-public          Active    5h16m
kube-system          Active    5h16m
logging              Active    3h39m
medilab-app-ns       Active    5h40m
medilab-db-ns        Active    3h58m
[ec2-user@ip-172-31-6-228 efk-demo]$
```

**Step-5: Create Persistent Volume Manifest File using above Storage Class and namespace and they apply it**

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: elasticsearch-master
  namespace: logging
  labels:
    app: elasticsearch-master
spec:
  capacity:
    storage: 20Gi
  accessModes:
    - ReadWriteOnce
  storageClassName: dev-storage-class
  persistentVolumeReclaimPolicy: Retain
  hostPath:
    path: /tmp/
```

```
[ec2-user@ip-172-31-6-228 efk-demo]$
[ec2-user@ip-172-31-6-228 efk-demo]$
[ec2-user@ip-172-31-6-228 efk-demo]$
[ec2-user@ip-172-31-6-228 efk-demo]$ kubectl apply -f pv.yml
persistentvolume/elasticsearch-master unchanged
[ec2-user@ip-172-31-6-228 efk-demo]$
```

**Step-6: check the PV created**

```
root@ip-172-31-5-221:~/elasticsearch# kubectl get pv -n logging
```

NAME	CAPACITY	ACCESS MODES	RECLAIM POLICY	STATUS	CLAIM
elasticsearch-master	20Gi	RWO	Retain	Available	
	dev-storage-class	27s			

**Step-7: Download and extract the elastic search using the below commands**

Command To Download : `$ wget https://helm.elastic.co/helm/elasticsearch/elasticsearch-7.17.1.tgz`

Command To Extract : `$ tar -xvf elasticsearch-7.17.1.tgz`

```
drwxr-xr-x  4 root root    4096 Nov 11 16:37 elasticsearch
```

**Step-8:** Go inside the elasticsearch directory and edit the values.yml file to include the storage class and also to verify the volume is within the range of volume we created in above step.

**Note:** By default we don't get this storageClassName attribute , we need to add it with the storage class we created.

```
root@ip-172-31-5-221:~/elasticsearch# ls -l
total 92
-rw-r--r-- 1 root root 341 Mar 8 2022 Chart.yaml
-rw-r--r-- 1 root root 29 Mar 8 2022 Makefile
-rw-r--r-- 1 root root 49860 Mar 8 2022 README.md
-rw-r--r-- 1 root root 194 Nov 11 16:31 custom-storage-class.yml
drwxr-xr-x 14 root root 4096 Nov 11 14:58 examples
-rw-r--r-- 1 root root 315 Nov 11 16:37 pv.yml
-rw-r--r-- 1 root root 222 Nov 11 16:22 pvc.yml
drwxr-xr-x 3 root root 4096 Nov 11 14:58 templates
-rw-r--r-- 1 root root 9542 Nov 12 09:23 values.yaml
root@ip-172-31-5-221:~/elasticsearch#
```

```
volumeClaimTemplate:
  accessModes: ["ReadWriteOnce"]
  storageClassName: "dev-storage-class"
  resources:
    requests:
      storage: 5Gi
rbac:
  create: false
  serviceAccountAnnotations: {}
  serviceAccountName: ""
  automountToken: true
```

**Step-3:** After we modify the values.yml file, install elasticsearch using below command (here we are giving modified values.yml and our custom namespace)

*\$ helm install elasticsearch-repo . -f values.yaml -n logging*

```
root@ip-172-31-5-221:~/elasticsearch# helm install elasticsearch-repo . -f values.yaml -n logging
NAME: elasticsearch-repo
LAST DEPLOYED: Sat Nov 12 09:30:50 2022
NAMESPACE: logging
STATUS: deployed
REVISION: 1
NOTES:
1. Watch all cluster members come up.
   $ kubectl get pods --namespace=logging -l app=elasticsearch-master -w2. Test cluster health using Helm test.
   $ helm --namespace=logging test elasticsearch-repo
root@ip-172-31-5-221:~/elasticsearch#
```

## Fluent – Bit Installation

**Step-1:** Download fluent-bit from below URL & Extract it using below commands

```
$ wget https://github.com/fluent/helm-charts/releases/download/fluent-bit-0.21.0/fluent-bit-0.21.0.tgz
```

```
$ tar -xvf fluent-bit-0.21.0.tgz
```

**Step-2:** Go inside fluent-bit extracted directory and Install fluent-d using helm with below command

```
$ helm install fluent-bit . -f values.yaml -n logging
```

```
[ec2-user@ip-172-31-6-228 fluent-bit]$  
[ec2-user@ip-172-31-6-228 fluent-bit]$  
[ec2-user@ip-172-31-6-228 fluent-bit]$ helm install fluent-bit . -f values.yaml -n logging
```

## Kibana Setup

**Step-1:** Download Kibana from below URL & Extract it

```
$ wget https://helm.elastic.co/helm/kibana/kibana-7.17.3.tgz
```

```
$ tar -xvf kibana-7.17.3.tgz
```

**Step-2 :** Go inside kibana directory and edit values.yml file to change Service Type & Node Port

Default Service Type : Cluster IP

Changed To : Node Port

Note: We have given NodePort as "30002" (Enable this port in Security Group) (If you don't give nodeport then it will generate random node port)

```
updateStrategy:  
  type: "Recreate"  
  
service:  
  type: NodePort  
  loadBalancerIP: ""  
  port: 5601  
  nodePort: 30002  
  labels: {}  
  annotations:  
    {}  
    # cloud.google.com/load-balancer-type: "Internal"  
    # service.beta.kubernetes.io/aws-load-balancer-internal: 0.0.0.0/0  
    # service.beta.kubernetes.io/azure-load-balancer-internal: "true"  
    # service.beta.kubernetes.io/openstack-internal-load-balancer: "true"  
    # service.beta.kubernetes.io/cce-load-balancer-internal-vpc: "true"  
  loadBalancerSourceRanges:  
    []
```

\$ helm install kibana . -f values.yaml -n logging

```
[ec2-user@ip-172-31-6-228 kibana]$
[ec2-user@ip-172-31-6-228 kibana]$
[ec2-user@ip-172-31-6-228 kibana]$ helm install kibana . -f values.yaml -n logging
```

### Step-3: Check Services created using below command

```
[ec2-user@ip-172-31-6-228 efk-demo]$
[ec2-user@ip-172-31-6-228 efk-demo]$
[ec2-user@ip-172-31-6-228 efk-demo]$ kubectl get svc -n logging
```

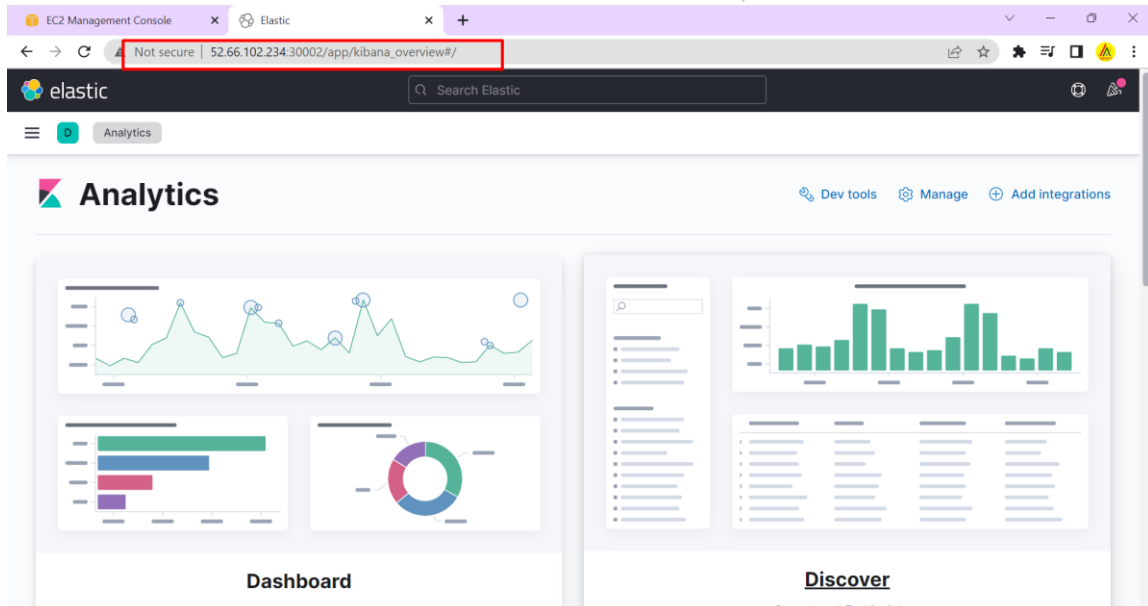
NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
elasticsearch-master	ClusterIP	10.100.210.186	<none>	9200/TCP,9300/TCP	46m
elasticsearch-master-headless	ClusterIP	None	<none>	9200/TCP,9300/TCP	46m
fluent-bit	ClusterIP	10.100.222.200	<none>	2020/TCP	18m
kibana-kibana	NodePort	10.100.202.64	<none>	5601:30002/TCP	15m

### Step-4: check the nodes in which Kibana pod is running

```
[ec2-user@ip-172-31-6-228 efk-demo]$ kubectl get pods -o wide -n logging
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINA
elasticsearch-master-0	1/1	Running	0	48m	192.168.178.149	ip-192-168-191-181.ap-south-1.compute.internal	<none>
elasticsearch-master-1	0/1	Pending	0	44m	<none>	<none>	<none>
fluent-bit-nhh72	1/1	Running	0	19m	192.168.66.64	ip-192-168-80-66.ap-south-1.compute.internal	<none>
fluent-bit-sljm6	1/1	Running	0	19m	192.168.183.41	ip-192-168-191-181.ap-south-1.compute.internal	<none>
kibana-kibana-864bcc7f5-ktllc	1/1	Running	0	16m	192.168.123.160	ip-192-168-80-66.ap-south-1.compute.internal	<none>

### Step-5: Access Kibana Dashboard using Node Public IP



The screenshot shows a web browser window with the URL `Not secure | 52.66.102.234:30002/app/kibana_overview/`. The page displays the Elastic Kibana Analytics dashboard. The dashboard includes a search bar, a sidebar with navigation options like 'Analytics', 'Dev tools', 'Manage', and 'Add integrations', and a main content area with various visualizations: a line chart, a bar chart, a horizontal bar chart, and a donut chart. The bottom of the dashboard is divided into two sections: 'Dashboard' and 'Discover'.

=== Learn Here.. Lead Anywhere..!! ===