How to Visualize Kubernetes Cluster Events in real-time

Last updated on Nov 26,2019 *7.2K Views*


Aayushi Johari
A technophile who likes writing about different technologies and spreading knowledge.

In this article, you will learn how to publish Kubernetes cluster events data to Amazon Elastic Search using Fluentd logging agent. The data will then be viewed using Kibana, an open-source visualization tool for Elasticsearch. Amazon ES consists of integrated Kibana integration.

We will walk you through with the following process:

- Creating a Kubernetes Cluster
- Creating an Amazon ES cluster
- Deploy Fluentd logging agent on Kubernetes cluster
- Visualize kubernetes date in Kibana

Step 1: Creating a Kubernetes Cluster

Kubernetes is an open source platform created by Google to manage containerized applications. it enables you to manage, scale and deploy your containerized apps in a clustered environment. We can orchestrate our containers across various hosts with Kubernetes, scale the containerized apps with all resources on the fly, and have a centralized container management environment.

We will start with creating Kubernetes cluster and I'll demonstrate you step by step, on how to install and configure Kubernetes on CentOS 7.

1.Configure Hosts

- vi /etc/hosts
- make changes according to your host details in the hosts file

```
172.31.7.47 k8s-master
172.31.0.4 k8s-worker1
172.31.3.232    k8s-worker2
```

2. Disable SELinux by executing below commands

- setenforce 0
- sed -i –follow-symlinks 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/sysconfig/selinux

3. Enable br_netfilter Kernel Module

The br_netfilter module is required for kubernetes installation.
Run the command below to enable the br_netfilter kernel module.

- modprobe br_netfilter
- echo '1' > /proc/sys/net/bridge/bridge-nf-call-iptables

4. Disable SWAP by running below commands.

- swapoff -a
- Then edit /etc/fstab and comment the swap line

```
UUID=0f790447-ebef-4ca0-b229-d0aa1985d57f       /       xfs     defaults        1 1
#/root/swap     swap    swap    sw      0 0
```

5. Install the latest version of Docker CE. Install the package dependencies for docker-ce by running below commands.

- yum install -y yum-utils device-mapper-persistent-data lvm2

Add the docker repository to the system and install docker-ce using the yum command.

- yum-config-manager –add-repo https://download.docker.com/linux/centos/docker-ce.repo
- yum install -y docker-ce

6. Install Kubernetes

```
[kubernetes]

name=Kubernetes
baseurl=<a href="https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64">https://packages.cl
enabled=1
gpgcheck=1
repo_gpgcheck=1
gpgkey=<a href="https://packages.cloud.google.com/yum/doc/yum-key.gpg">https://packages.cloud.google.cc
       <a href="https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg">https://packages.cloud.g
EOF
```

Install the kubernetes packages kubeadm, kubelet, and kubectl using by running yum command below.

- systemctl start docker && systemctl enable docker

After the installation is complete, restart all those servers. After restart start the services docker and kubelet

- systemctl start docker && systemctl enable docker
- systemctl start kubelet && systemctl enable kubelet

7. Kubernetes Cluster Initialization
Login to master server and run the below command

- systemctl start kubelet && systemctl enable kubelet

Once Kubernetes initialization is complete, you will get the results. Copy the commands from the results you got and Execute it to start using the cluster.

Make a note of the kubeadm join command from results. The command will be used to register new nodes to the kubernetes cluster.

8. Deploy the flannel network to the kubernetes cluster
kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml

The flannel network has been deployed to the Kubernetes cluster.
Wait for some time and then check kubernetes node and pods using commands below.

- kubectl get nodes
- kubectl get pods –all-namespaces

And you will get the 'k8s-master' node is running as a 'master' cluster with status 'ready', and you will get all pods that are needed for the cluster, including the 'kube-flannel-ds' for network pod configuration.
9. Adding Nodes to the clusterConnect to the node01 server and run the kubeadm join command

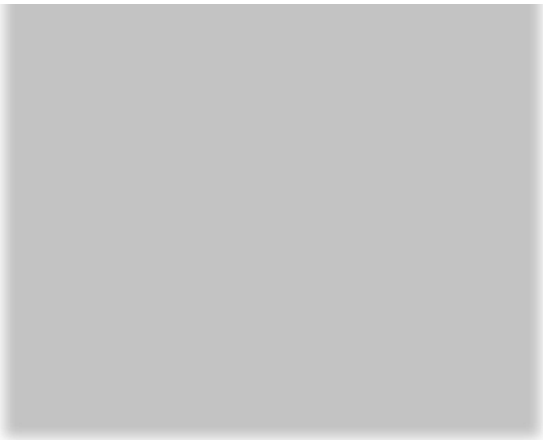
- kubeadm join 172.31.7.47:6443 –token at03m9.iinkh5ps9q12sh2i –discovery-token-ca-cert-hash sha256:3f6c1824796ef1ff3d9427c883bde915d5bc13331d74891d831f29a8c4a0c5ab

Connect to the node02 server and run the kubeadm join command

- kubeadm join 172.31.7.47:6443 –token at03m9.iinkh5ps9q12sh2i –discovery-token-ca-cert-hash sha256:3f6c1824796ef1ff3d9427c883bde915d5bc13331d74891d831f29a8c4a0c5ab

Wait for some time and Validate the 'k8s-master' master cluster server, check the nodes and pods using the following command.

- kubectl get nodes



Now you will get worker1 and worker2 has been added to the cluster with status 'ready'.

- kubectl get pods –all-namespaces

Kubernetes cluster master initialization and configuration has been completed.

Step 2: Creating an Amazon ES cluster

Elasticsearch is an open source search and analytics engine which is used for log analysis and real-time monitoring of applications. Amazon Elasticsearch Service (Amazon ES) is an AWS service that allows the deployment, operation, and scale of Elasticsearch in the AWS cloud. You can use Amazon ES to analyze email sending events from your Amazon SES

We will create an Amazon ES cluster and then Deploy Fluentd logging agent to Kubernetes cluster which will collect logs and send to Amazon ES cluster

This section shows how to use the Amazon ES console to create an Amazon ES cluster.

To create an Amazon ES cluster

1. Sign in to the AWS Management Console and open the Amazon Elasticsearch Service console at https://console.aws.amazon.com/es/
2. Select **Create a New Domain and choose Deployment type** in the Amazon ES console.

3. Under Version, leave the default value of the Elasticsearch version field.
4. Select Next
5. Type a name for your Elastic search domain on the **configure cluster** page under **Configure Domain.**
6. On the Configure cluster page, select the following options under data Instances
    - **Instance type –** Choose t2.micro.elasticsearch (Free tier eligible).
    - **Number of Instance –** 1
7. Under Dedicated Master Instances
    - **Enable dedicated master –** Do not enable this option.
    - **Enable zone awareness –** Do not enable this option.
8. Under Storage configuration, choose the following options.
    - **Storage type –** Choose EBS. For the EBS settings, choose EBS volume type of General Purpose (SSD) and EBS volume size of 10.
9. Under encryption **– Do not enable this option**
10. Under snapshot configuration
    - **Automated snapshot start hour –** Choose Automated snapshots start hour 00:00 UTC (default).
11. Choose Next
12. Under Network configuration select VPC Access and select details as per your VPC is shown below.

Under Kibana authentication: – Do not enable this option.

13. To set the access policy, select Allow open access to the domain.Note:- In production you should restrict access to specific IPaddress or Ranges.

14. Choose Next.
15. On the Review page, review your settings, and then choose Confirm and Create.

Note: The cluster will take up to ten minutes to deploy. Take note of your Kibana URL once you click the elastic search domain created.

Step 3: Deploy Fluentd logging agent on Kubernetes cluster

Fluentd is an open source data collector, which lets you unify the data collection and consumption for better use and understanding of data. In this case, we will deploy Fluentd logging on Kubernetes cluster, which will collect the log files and send to the Amazon Elastic Search.

We will create a ClusterRole which provides permissions on pods and namespace objects to make get, list and watch request to cluster.

First, we need to configure RBAC (role-based access control) permissions so that Fluentd can access the appropriate components.

1.fluentd-rbac.yaml:

```
  namespace: kube-system

---

apiVersion: rbac.authorization.k8s.io/v1beta1
kind: ClusterRole
metadata:
  name: fluentd
  namespace: kube-system
rules:
- apiGroups:
  - ""
  resources:
  - pods
  - namespaces
  verbs:
  - get
  - list
  - watch

---

kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1beta1
metadata:
  name: fluentd
roleRef:
  kind: ClusterRole
  name: fluentd
  apiGroup: rbac.authorization.k8s.io
subjects:
- kind: ServiceAccount
  name: fluentd
  namespace: kube-system
```

Create: $ kubectl create -f kubernetes/fluentd-rbac.yaml

Now, we can create the DaemonSet.

2. fluentd-daemonset.yaml

```
  namespace: kube-system
  labels:
    k8s-app: fluentd-logging
    version: v1
    kubernetes.io/cluster-service: "true"
spec:
  template:
    metadata:
      labels:
        k8s-app: fluentd-logging
        version: v1
        kubernetes.io/cluster-service: "true"
    spec:
      serviceAccount: fluentd
      serviceAccountName: fluentd
      tolerations:
      - key: node-role.kubernetes.io/master
        effect: NoSchedule
      containers:
      - name: fluentd
        image: fluent/fluentd-kubernetes-daemonset:v1.3-debian-elasticsearch
        env:
          - name:  FLUENT_ELASTICSEARCH_HOST
            value: "elasticsearch.logging"
          - name:  FLUENT_ELASTICSEARCH_PORT
            value: "9200"
          - name: FLUENT_ELASTICSEARCH_SCHEME
            value: "http"
          - name: FLUENT_UID
            value: "0"
        resources:
          limits:
            memory: 200Mi
          requests:
            cpu: 100m
            memory: 200Mi
        volumeMounts:
        - name: varlog
          mountPath: /var/log
        - name: varlibdockercontainers
          mountPath: /var/lib/docker/containers
          readOnly: true
      terminationGracePeriodSeconds: 30
      volumes:
      - name: varlog
        hostPath:
          path: /var/log
      - name: varlibdockercontainers
        hostPath:
          path: /var/lib/docker/containers
```

Make sure to define FLUENT_ELASTICSEARCH_HOST & FLUENT_ELASTICSEARCH_PORT according to your elastic search environment

Deploy:

$ kubectl create -f kubernetes/fluentd-daemonset.yaml

Validate the logs

$ kubectl logs fluentd-lwbt6 -n kube-system | grep Connection

You should see that Fluentd connect to Elasticsearch within the logs:

Step 4: Visualize kubernetes data in Kibana

1. Connect to the kibana dashboard URL to get from Amazon ES console
2. To see the logs collected by Fluentd in Kibana, click “Management” and then select “Index Patterns” under “Kibana”
3. choose the default Index pattern (logstash-*)

4. Click Next Step and set the “Time filter field Name” (@timestamp) and choose Create index pattern

5. Click Discover to view your application logs

6. Click Visualize and select create a visualization and choose Pie. Fill up the following fields as shown below.

- Select Logstash-* index and click split slices
- Aggregation – Significant terms
- Field = Kubernetes.pod_name.keyword
- Size – 10

7. And Apply Changes

That's it! This is how you can visualize the Kubernetes Pod created in Kibana.

Summary:

Monitoring by log analysis is a critical component of any application deployment. You can gather and consolidate logs across your cluster in Kubernetes to monitor the whole cluster from one single dashboard. In our example, we have seen fluentd act as a mediator between kubernetes cluster and Amazon ES. Fluentd combines log collection and aggregation and sends logs to Amazon ES for log analytics and data visualization with kibana.

The above example shows how to add AWS Elastic search logging and kibana monitoring to kubernetes cluster using fluentd.

*If you found this Kubernetes blog relevant, check out the **Kubernetes Certification Training** by Edureka, a trusted online learning company with a network of more than 250,000 satisfied learners spread across the globe.*

Got a question for us? Please mention it in the comments section and we will get back to you.