EKS & NGINX Load Balancer Monitoring with Prometheus, Grafana, and Alerts

Introduction

Monitoring is crucial for maintaining a healthy infrastructure. This guide will walk you through setting up monitoring for an NGINX Load Balancer deployed in an Amazon EKS cluster, using Prometheus for data collection, Grafana for visualization, and alerts for proactive incident management.

Prerequisites

Before starting, make sure you have the following in place:

- ***Amazon EKS Cluster** with Kubernetes deployed*
- ***NGINX Ingress Controller** running within the EKS cluster*
- ***kubectl** configured to communicate with your EKS cluster*
- ***Helm** installed for deploying Kubernetes applications*
- ***Prometheus Operator** installed on the cluster*
- ***Grafana** for visualization*

Step 1: Install Prometheus on EKS

Prometheus collects metrics from different services and stores them for querying and alerting. We'll deploy Prometheus using Helm, which simplifies the process of installing complex applications.

Add Prometheus Helm chart repository:

```
helm repo add prometheus-community
https://prometheus-community.github.io/helm-charts
helm repo update
```

1.

Install Prometheus using Helm:

```
helm install prometheus
prometheus-community/kube-prometheus-stack --namespace
monitoring --create-namespace
```

2.

Verify Prometheus Installation: *Run the following command to check the pods running in the monitoring namespace:*

```
kubectl get pods -n monitoring
```

3. *You should see pods for Prometheus and its components like* `Prometheus-server` *and* `Prometheus-alertmanager`*.*

Step 2: Install Grafana on EKS

Grafana will visualize the metrics collected by Prometheus. We'll install Grafana using Helm as well.

Install Grafana:

```
helm install grafana prometheus-community/grafana
--namespace monitoring
```

1.

Verify Grafana Installation: *Check if the Grafana pod is running:*

```
kubectl get pods -n monitoring
```

2.

Access Grafana Dashboard: *To access Grafana, port-forward the Grafana pod to your local machine:*

```
kubectl port-forward service/grafana 3000:80 -n monitoring
```

3. *Visit* `http://localhost:3000` *in your browser and log in using the default credentials:*
    - ***Username:*** *admin*
    - ***Password:*** *prom-operator*
4. ***Add Prometheus as a Data Source in Grafana:***
    - *In the Grafana UI, go to* ***Settings > Data Sources > Add Data Source****.*
    - *Select* ***Prometheus*** *as the data source and enter* `http://prometheus-server.monitoring.svc:80` *as the URL.*
    - *Click* ***Save & Test*** *to ensure Grafana can query Prometheus.*

Step 3: Monitor NGINX Ingress Controller Metrics

To monitor the NGINX Load Balancer, you need to collect NGINX Ingress Controller metrics via Prometheus. These metrics are exposed by the NGINX ingress controller itself.

Install the NGINX Ingress Controller: *Use the Helm chart to install NGINX Ingress Controller:*

arduino
CopyEdit

```
helm install nginx-ingress ingress-nginx/ingress-nginx
--namespace ingress-nginx --create-namespace
```

1.

Expose NGINX Metrics: *Ensure that the NGINX Ingress Controller is exposing metrics in a format Prometheus can scrape. If it's not, you can configure the ingress controller to expose these metrics by modifying the deployment:*
CopyEdit

```
kubectl edit deployment nginx-ingress-controller -n
ingress-nginx
```

Add or modify the arguments to expose metrics:
yaml
CopyEdit

```
args:
  - /nginx-ingress-controller
  - --metrics-server
  - --nginx-metrics-port=9913
```

2.
3. ***Configure Prometheus to Scrape NGINX Metrics:*** *You need to inform Prometheus to scrape metrics from the NGINX Ingress Controller.*

Create a `ServiceMonitor` *object for Prometheus to scrape metrics:*
yaml
CopyEdit

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
```

```
  name: nginx-ingress-controller
  namespace: monitoring
spec:
  selector:
    matchLabels:
      app: nginx-ingress-controller
  endpoints:
    - port: metrics
      interval: 15s
```

-

Apply the `ServiceMonitor`:
CopyEdit

```
kubectl apply -f nginx-service-monitor.yaml
```

-

Step 4: Set Up Grafana Dashboards

1. ***Import the Prometheus Default Dashboards:*** *Grafana comes with some default dashboards, but you can import more specific ones. Here's how to import the NGINX Ingress Controller Dashboard:*
   - *Go to **Dashboard** > **Import** in Grafana.*
   - *Use the dashboard ID `9614` (NGINX Ingress Controller Dashboard) and click **Load**.*
   - *Select the Prometheus data source and click **Import**.*
2. ***Customize Dashboards:*** *Customize your dashboard to include NGINX metrics such as:*
   - *Requests per second*
   - *Response codes (2xx, 4xx, 5xx)*

- *Latencies*
- *Traffic volume*

Step 5: Set Up Alerts in Prometheus

Alerts help you get notified about issues before they impact users.

Configure Alerting Rules in Prometheus: *To set up alert rules, create an* `alerting_rules.yaml` *file:*
yaml
CopyEdit

```
groups:
  - name: nginx-ingress
    rules:
      - alert: NginxHighErrorRate
        expr:
rate(nginx_ingress_controller_requests{status=~"4..|5.."
}[5m]) > 10
        for: 5m
        labels:
          severity: critical
        annotations:
          summary: "NGINX has high error rate"
          description: "NGINX is returning more than 10
errors per minute."
```

1.

Apply the Alert Rules: *Apply the alerting rules:*
CopyEdit

```
kubectl apply -f alerting_rules.yaml
```

2.
3. ***Configure Alertmanager:*** *Alertmanager will send notifications when an alert is triggered. Ensure that Alertmanager is configured to send alerts to your preferred communication channel (Slack, email, etc.).*

Edit the Alertmanager configuration:
CopyEdit

```
kubectl edit configmap
alertmanager-prometheus-alertmanager -n monitoring
```

-
- *Add your alert notification settings (e.g., Slack webhook URL).*

Step 6: Monitor Alerts in Grafana

1. ***Create an Alert Panel*** *in Grafana: Add an alert panel to your Grafana dashboard to visualize and monitor Prometheus alerts.*
2. ***Configure Alert Notifications in Grafana:***
   - *Go to* ***Alerting*** *>* ***Notification Channels*** *in Grafana.*
   - *Add a new notification channel (e.g., Slack) with the necessary credentials.*