Prometheus & Grafana Monitoring Setup on EC2 using Minikube

Overview

This setup involves running Prometheus and Grafana on an Ubuntu EC2 instance using Minikube. Additionally, a Node.js application exposing Prometheus metrics is deployed and monitored via Grafana.

Tools & Versions

- **EC2 OS:** Ubuntu 24.04
- **Kubernetes:** Minikube (latest)
- **Prometheus & Grafana:** Deployed via Helm
- **App Container:** pradeepaanandh/node-prom-app

Steps Performed

1. Prometheus & Grafana Installation

- Helm charts used to deploy Prometheus and Grafana:

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

2. Expose Services via NodePort

- Edited Grafana and Prometheus services:

```
type: NodePort
ports:
  - port: 80
    nodePort: <custom-port>
```

- Final NodePorts:

- Grafana: `30877`

- Prometheus: `32683`

3. Node.js Application Deployment

- **Deployment.yaml:**

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: node-prom-app
  labels:
    app: node-prom
spec:
  replicas: 1
  selector:
    matchLabels:
      app: node-prom
  template:
    metadata:
      labels:
        app: node-prom
      annotations:
        prometheus.io/scrape: "true"
        prometheus.io/port: "3000"
    spec:
      containers:
        - name: node-prom
          image: pradeepaanandh/node-prom-app
          ports:
            - containerPort: 3000
```

- **Service.yaml:**

```
apiVersion: v1
kind: Service
metadata:
  name: node-prom-svc
spec:
  type: NodePort
  selector:
    app: node-prom
  ports:
    - port: 3000
      targetPort: 3000
      nodePort: 30081
```

4. Grafana Dashboard Setup

- Access Grafana: `http://<EC2-IP>:30877`
- Login default: `admin/admin` (then reset password)
- Add Prometheus data source:
- URL: `http://prometheus-server.monitoring.svc.cluster.local`
- Create new dashboard → Panel → Query: `up`

Errors & Troubleshooting

❌Node.js app not showing in Prometheus UI

Fix:

- Ensure proper annotations are set in Deployment.yaml:

```
annotations:
  prometheus.io/scrape: "true"
  prometheus.io/port: "3000"
```

- Validate the service and pod are labeled `app: node-prom`

❌NodePort mismatch

Fix:

- Verified with `kubectl get svc -n monitoring`
- Ensure correct `nodePort` value assigned in `Service.yaml`

❌Service not found error

Fix:

- Ran:

```
kubectl apply -f service.yaml -n monitoring
```

to make sure service is created in right namespace.

❌Prometheus label match parse error

```
Error: parse error: unexpected identifier "node" in label matching
```

Fix:

- Incorrect query syntax.
- Correct usage:

```
up{job="node-prom"}
```

or simply start with `up` to validate target status.

❌No data in Grafana panel

Fix:

- Wait few seconds after panel creation
- Ensure Prometheus is added as Data Source
- Cross-check Prometheus targets at: `http://<EC2-IP>:32683/targets`

Result

- Node.js app is successfully deployed and monitored.
- Metrics are being scraped by Prometheus.
- Grafana panel displays metric graphs.

Next Steps (Planned)

- Automate this setup using Helm and Terraform.
- Create custom alerts in Prometheus.
- Set up dashboards with multiple metrics.
- Integrate Slack alert channel.

Author: pradeepaanandh\ **Date:** 2025-07-16