Steps To install Prometheus and Grafana on an Amazon EKS (Elastic Kubernetes Service) cluster and configure Grafana to use Prometheus as a data source for collecting metrics, follow these step-by-step instructions.

To copy code : https://chatgpt.com/share/67c30ba2-3af4-8006-a925-248a5ff047be .

Deployment of Prometheus & Grafana

Step 1: Deploy Prometheus & Grafana Using Helm

We will use the **Prometheus Community Helm Chart** to deploy both Prometheus and Grafana.

1. Add the Prometheus Helm Repository

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

helm repo update
```

2. Create a Namespace for Monitoring

```
kubectl create namespace monitoring
```

3. Install Prometheus Stack

Use the following Helm command to install **Prometheus and Grafana**:

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

This installs **Prometheus**, **Grafana**, and the necessary exporters to collect Kubernetes metrics.

Step 2: Expose Prometheus & Grafana Services

By default, these services are available inside the cluster. You can access them externally using **kubectl port-forward** or an **ELB/ALB Ingress Controller**.

1. Port-forward Prometheus

```
kubectl port-forward -n monitoring svc/prometheus-kube-prometheus-prometheus 9090:9090
```

```
after this it will start forwarding the traffic from local host to the pod
, for next execution process use new terminal , u cant use the same one
beacouse it is in process of forwarding .
```

You can now access Prometheus at http://localhost:9090.

2. Port-forward Grafana

```
kubectl port-forward -n monitoring svc/prometheus-grafana 3000:80 .

after this it will start forwarding the traffic from local host to the pod
, for next execution process use new terminal , u cant use the same one
beacouse it is in process of forwarding .
```

Grafana will be available at http://localhost:3000.

Step 3: Get Grafana Admin Credentials

The default username is `admin`. To get the password, run:

```
kubectl get secret -n monitoring prometheus-grafana -o
jsonpath="{.data.admin-password}" | base64 --decode ; echo
```

Use these credentials to log in to Grafana.

Step 4: Configure Prometheus as a Data Source in Grafana

1. **Log in to Grafana** at `http://localhost:3000`
2. Navigate to **Configuration → Data Sources**.
3. Click **"Add Data Source"**.
4. Select **Prometheus**.
5. In the **URL** field, enter:

   ```
   http://prometheus-kube-prometheus-
   prometheus.monitoring.svc.cluster.local:9090
   ```

6. Click **Save & Test**.

Step 5: Import Kubernetes Dashboards

1. In Grafana, go to **Dashboards → Import**.

2. Use an existing **Kubernetes Dashboard** from Grafana's repository (e.g., ID `3119` for Kubernetes cluster monitoring).
3. Select **Prometheus** as the data source.
4. Click **Import**.

Now, you should see Kubernetes metrics visualized in Grafana!

Step 6: (Optional) Expose Services Using LoadBalancer or Ingress

To access **Grafana** externally:

```
kubectl patch svc prometheus-grafana -n monitoring -p '{"spec": {"type": "LoadBalancer"}}'
```

Get the external IP:

```
kubectl get svc -n monitoring prometheus-grafana
```

Use the external IP to access Grafana.

But After this if you try to access this external IP it will not open this is beacouse u haven't opened the inbound rule,

If u are using aks , Go to the aks vms agent pool ,and go to the networking and add the inbound rules and the port which have to open and accessed . After it will connect using with node ip and port .

If u are using eks go to the security group attached to the vm's and add inbound rules.

Final Notes

- **Prometheus** collects Kubernetes metrics automatically.
- **Grafana** visualizes the data from Prometheus.
- Use **Helm values.yaml** if you need custom configurations.

Let me know if you need any clarifications! 🚀

To expose Prometheus and Grafana in EKS, you can use NodePort, LoadBalancer, or Ingress. Below are detailed steps for each method.

1□ Expose Services Using NodePort

A **NodePort** exposes services on a static port (30000–32767) on all cluster nodes.

Modify Prometheus Service to Use NodePort

```
kubectl edit svc -n monitoring prometheus-kube-prometheus-prometheus
```

Find the `spec.type` field and change it to `NodePort`:

```
spec:
  type: NodePort
```

Find the **Prometheus port section** and add a `nodePort` value:

```
ports:
  - name: web
    port: 9090
    targetPort: 9090
    nodePort: 30090
```

Save and exit.

Modify Grafana Service to Use NodePort

```
kubectl edit svc -n monitoring prometheus-grafana
```

Change the `spec.type` to `NodePort` and set a `nodePort`:

```
spec:
  type: NodePort
ports:
  - name: service
    port: 80
    targetPort: 3000
    nodePort: 30030
```

Save and exit.

Access Prometheus & Grafana

Find a worker node's public IP:

```
kubectl get nodes -o wide
```

Then access:

- **Prometheus**: `http://<NODE_PUBLIC_IP>:30090`
- **Grafana**: `http://<NODE_PUBLIC_IP>:30030`

- But After this if you try to access this external IP it will not open this is beacouse u haven't opened the inbound rule,

- If u are using aks , Go to the aks vms agent pool ,and go to the networking and add the inbound rules and the port which have to open and accessed . After it will connect using with node ip and port .

- If u are using eks go to the security group attached to the vm's and add inbound rules.

2□ Expose Services Using LoadBalancer

A **LoadBalancer** exposes the service externally via an AWS ELB.

Modify Prometheus Service to Use LoadBalancer

```
kubectl patch svc prometheus-kube-prometheus-prometheus -n monitoring -p
'{"spec": {"type": "LoadBalancer"}}'
```

Modify Grafana Service to Use LoadBalancer

```
kubectl patch svc prometheus-grafana -n monitoring -p '{"spec": {"type":
"LoadBalancer"}}'
```

Get External Access URLs

```
kubectl get svc -n monitoring prometheus-kube-prometheus-prometheus
prometheus-grafana
```

Look for the **EXTERNAL-IP** field.

Access:

- **Prometheus**: `http://<EXTERNAL-IP>:9090`
- **Grafana**: `http://<EXTERNAL-IP>`

- But After this if you try to access this external IP it will not open this is beacouse u haven't opened the inbound rule,

- If u are using aks , Go to the aks vms agent pool ,and go to the networking and add the inbound rules and the port which have to open and accessed . After it will connect using with node ip and port .

- If u are using eks go to the security group attached to the vm's and add inbound rules.

3□ Expose Services Using Ingress

Ingress provides a single entry point using **AWS ALB Ingress Controller**.

1. Install AWS ALB Ingress Controller

```
helm repo add eks https://aws.github.io/eks-charts
helm repo update
helm install aws-load-balancer-controller eks/aws-load-balancer-controller
\
  --set clusterName=my-eks-cluster \
  --set serviceAccount.create=false \
  --set region=us-east-1 \
  --set vpcId=<VPC_ID> \
  -n kube-system
```

2. Create an Ingress Resource

Create a file `grafana-ingress.yaml`:

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: grafana-ingress
  namespace: monitoring
  annotations:
    alb.ingress.kubernetes.io/scheme: internet-facing
spec:
  ingressClassName: alb
  rules:
  - host: grafana.mydomain.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
```

```
            name: prometheus-grafana
            port:
              number: 80
```

Apply it:

```
kubectl apply -f grafana-ingress.yaml
```

3. Get the Ingress URL

```
kubectl get ingress -n monitoring .
```

Once the ALB is provisioned, you can access Grafana at http://grafana.mydomain.com.

- But After this if you try to access this external IP it will not open this is beacouse u haven't opened the inbound rule,

- If u are using aks , Go to the aks vms agent pool ,and go to the networking and add the inbound rules and the port which have to open and accessed . After it will connect using with node ip and port .
- If u are using eks go to the security group attached to the vm's and add inbound rules.

Conclusion

Method	Access Type	Pros	Cons
NodePort	Local only via Node IP	Easy to set up	Requires manually finding node IP
LoadBalancer	Public AWS ELB	Direct external access	Creates a new ELB per service
Ingress	Single domain-based access	Centralized entry point, supports HTTPS	Requires ALB Ingress Controller

Let me know if you need more details! 🚀