++++++++++++++++++++

Kubernetes Monitoring

++++++++++++++++++++++

-> Prometheus is an open-source systems monitoring and alerting toolkit

-> Prometheus collects and stores its metrics as time series data

-> It provides out-of-the-box monitoring capabilities for the k8s container orchestration platform.

-> Grafana is a database analysis and monitoring tool

-> Grafana is a multi-platform open source analytics and interactive visualization web application.

-> It provides charts, graphs, and alerts for to web when connected to supported data sources.

-> Grafana allows you to query, visualize, alert on and understand your metrics no matter where they are stored. Create, explore and share dashboards.

Note: Graphana will connect with Prometheus for data source.

++++++++++++++++++++++++++++++++++++

How to deploy premetous and Grafana in K8s

++++++++++++++++++++++++++++++++++++

-> Most Efficient way is using Helm Chart to deploy Prometheus Operator

++

Install Prometheus & Grafana

++

Add the latest helm repository in Kubernetes

$ helm repo add stable https://charts.helm.sh/stable

Add prometheus repo to helm

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

Update Helm Repo

$ helm repo update

Search Repo

$ helm search repo prometheus-community

install prometheus

$ helm install stable prometheus-community/kube-prometheus-stack

Get all pods

$ kubectl get pods

Node: You should see prometheus pods running

#Check the services

$ kubectl get svc

#By default prometheus and grafana service is available within the cluster using ClusterIP, to access them outside lets change it either NodePort or Loadbalancer.

$ kubectl edit svc stable-kube-prometheus-sta-prometheus

```
  ports:
  - name: http-web
    port: 9090
    protocol: TCP
    targetPort: 9090
  selector:
    app.kubernetes.io/name: prometheus
    prometheus: stable-kube-prometheus-sta-prometheus
  sessionAffinity: None
  type: ClusterIP
```

Change

```
    port: 9090
    protocol: TCP
    targetPort: 9090
  selector:
    app.kubernetes.io/name: prometheus
    prometheus: stable-kube-prometheus-sta-prometh
  sessionAffinity: None
  type: NodePort
status:
  loadBalancer: {}
```

$ kubectl edit svc stable-kube-prometheus-sta-prometheus

Now edit the grafana service

$ kubectl edit svc stable-grafana

```
    nodePort: 31903
    port: 80
    protocol: TCP
    targetPort: 3000
  selector:
    app.kubernetes.io/instance: stable
    app.kubernetes.io/name: grafana
  sessionAffinity: None
  type: NodePort
status:
  loadBalancer: {}
```

Verify the service if changed to **NodePort**

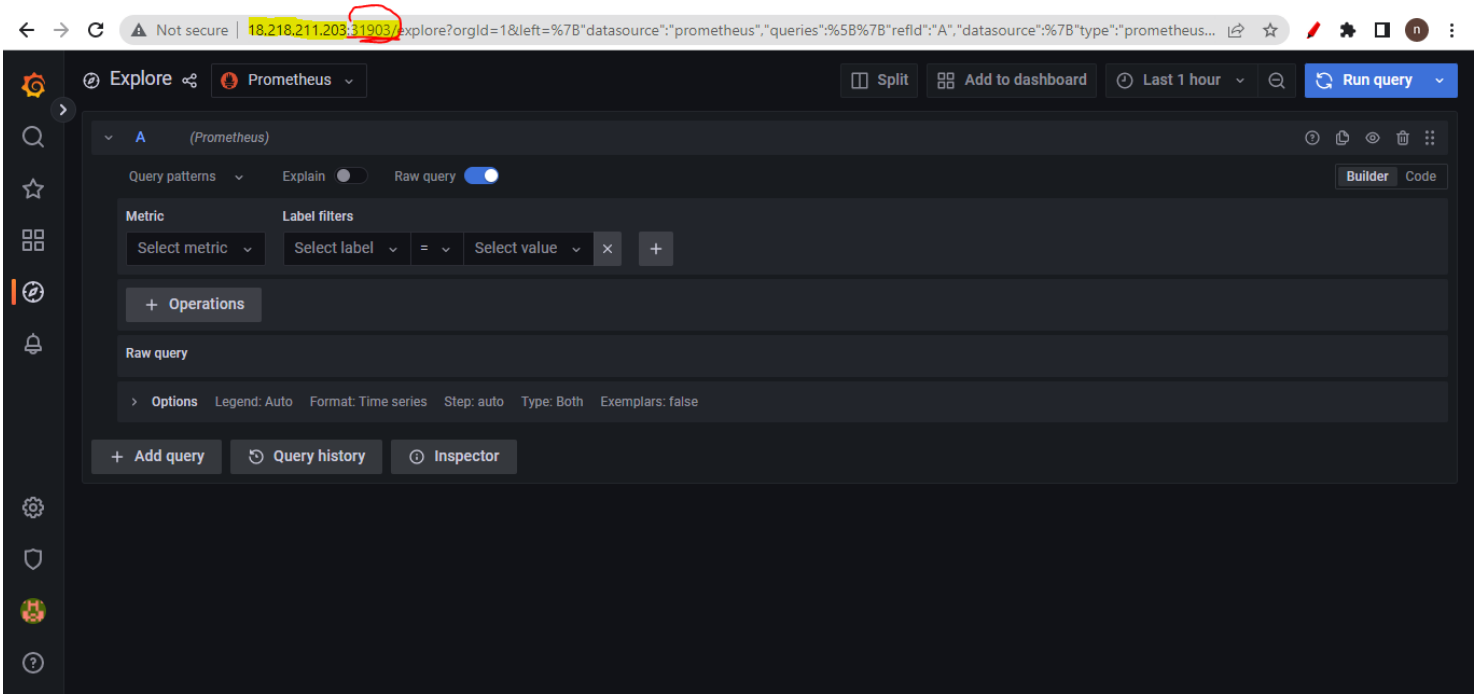
$ kubectl get svc

```
root@ip-172-31-22-198:~/K8s_7PM# kubectl get svc
E0305 13:05:27.655228   11090 memcache.go:255] couldn't get resource list for metrics.k8s.io/v1beta1: th
e server is currently unable to handle the request
E0305 13:05:27.675846   11090 memcache.go:106] couldn't get resource list for metrics.k8s.io/v1beta1: th
e server is currently unable to handle the request
E0305 13:05:27.678079   11090 memcache.go:106] couldn't get resource list for metrics.k8s.io/v1beta1: th
e server is currently unable to handle the request
E0305 13:05:27.680542   11090 memcache.go:106] couldn't get resource list for metrics.k8s.io/v1beta1: th
e server is currently unable to handle the request
NAME                                     TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)
          AGE
alertmanager-operated                    ClusterIP   None             <none>        9093/TCP,9094/TCP,9
094/UDP   21m
kubernetes                               ClusterIP   10.96.0.1        <none>        443/TCP
          44m
metrics-server                           ClusterIP   10.98.47.30      <none>        443/TCP
          24m
prometheus-operated                      ClusterIP   None             <none>        9090/TCP
          21m
stable-grafana                           NodePort    10.101.109.251   <none>        80:31903/TCP
          21m
stable-kube-prometheus-sta-alertmanager  ClusterIP   10.101.22.23     <none>        9093/TCP
          21m
stable-kube-prometheus-sta-operator      ClusterIP   10.108.40.141    <none>        443/TCP
          21m
stable-kube-prometheus-sta-prometheus    NodePort    10.100.18.16     <none>        9090:32677/TCP
          21m
stable-kube-state-metrics                ClusterIP   10.98.253.179    <none>        8080/TCP
          21m
stable-prometheus-node-exporter          ClusterIP   10.103.90.251    <none>        9100/TCP
          21m
vertex-svc                               NodePort    10.100.190.181   <none>        8080:30001/TCP
          35m
```

To access Prometheus web interface copy Loadbalancer URL and port number 9090 To access Grafana web interface copy Loadbalancer URL and port number 80

UserName: **admin**

Password: **prom-operator**



When we use kubeadm we must use Nodeport (Donot Expose outside Cluster) or Ingress Controller

290

It looks like you are using a custom Kubernetes Cluster (using `minikube`, `kubeadm` or the like). In this case, there is no LoadBalancer integrated (unlike AWS or Google Cloud). With this default setup, you can only use `NodePort` or an Ingress Controller.

With the Ingress Controller you can setup a domain name which maps to your pod; you don't need to give your Service the `LoadBalancer` type if you use an Ingress Controller.

But when we use AWS EKS then we can use LoadBalancer (Expose outside Cluster)