**Prometheus-Grafana-Alertmanager Installation guide on k8s Cluster**

1. Installation and Setup

Install the kube-prometheus stack, a collection of Kubernetes manifests, Grafana dashboards, and Prometheus rules, Alert manager, Alert manager rules, Node-exporter combined with documentation and scripts to provide easy to operate end-to-end Kubernetes cluster monitoring with Prometheus using the Prometheus Operator. See the [kube-prometheus](https://github.com/prometheus-operator/kube-prometheus) for details about components, dashboards, and alerts.

**Repository Info:**

1. Add Repo:

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

helm repo update

1. Download values file :

helm show values prometheus-community/kube-prometheus-stack > prometheus.yaml

Note: Once it downloads , we need to configured it according to our use case

**Configuring values file (prometheus.yaml) of Thanos with S3:**

1. To enable thanos sidecar, we need to edit thanos configuration in prometheusSpec section

And also add thanos objectstoreconfig as a secret as follows:

thanos:

objectStorageConfig:

key: "thanos.yaml"

name: "thanos-objstore-config"

Note : Need to create kubernetes secret for [Example objectStoreConfig](https://thanos.io/tip/thanos/storage.md/#s3)

thanos.yaml:

type: S3

config:

bucket: "bucket-name"

endpoint: "s3.amazonaws.com"

access\_key: ""

secret\_key: ""

region: "us-east-1"

insecure: false

signature\_version2: false

put\_user\_metadata: {}

http\_config:

idle\_conn\_timeout: 0s

response\_header\_timeout: 0s

insecure\_skip\_verify: false

trace:

enable: false

part\_size: 0

Write the command in terminal to create secret:

kubectl create secret generic thanos-objstore-config –from-file=thanos.yaml

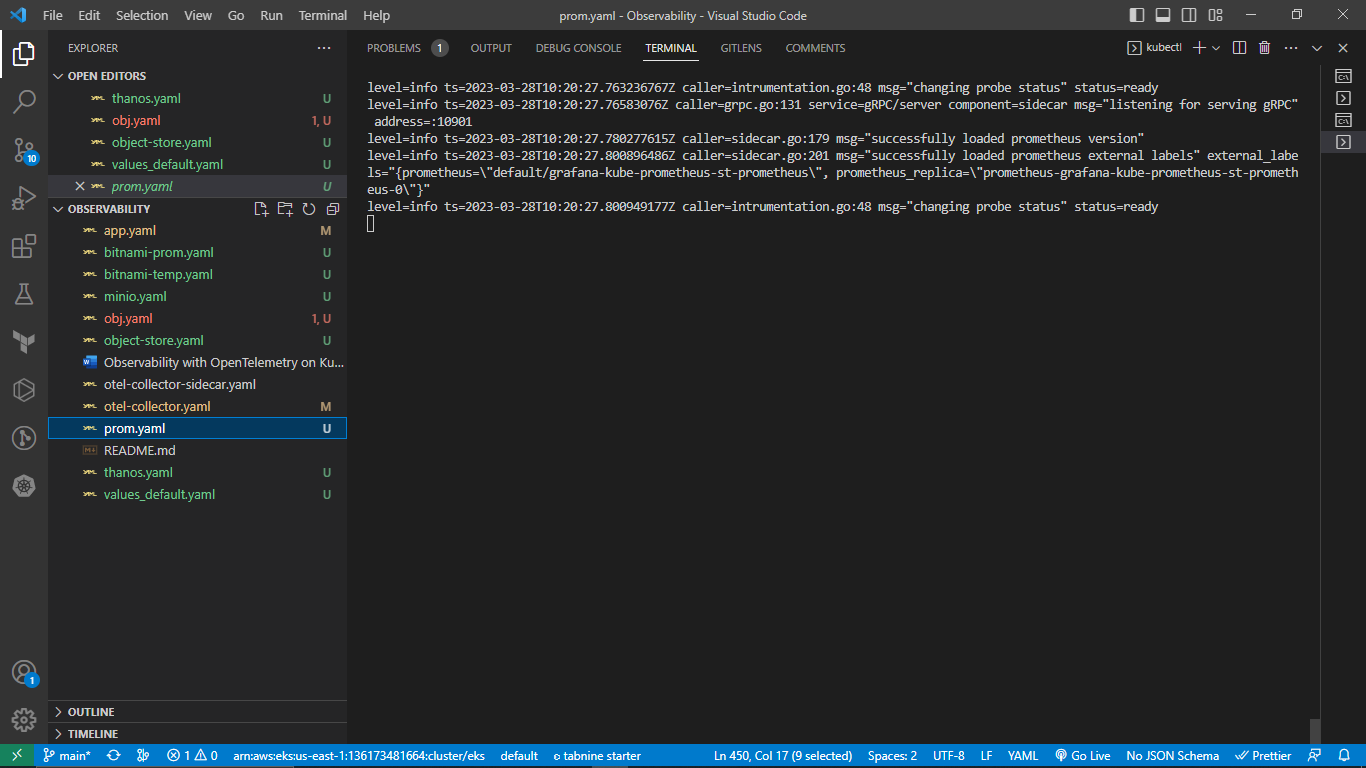
**Install Helm chart :**

helm install [RELEASE\_NAME] prometheus-community/kube-prometheus-stack –f prometheus.yaml

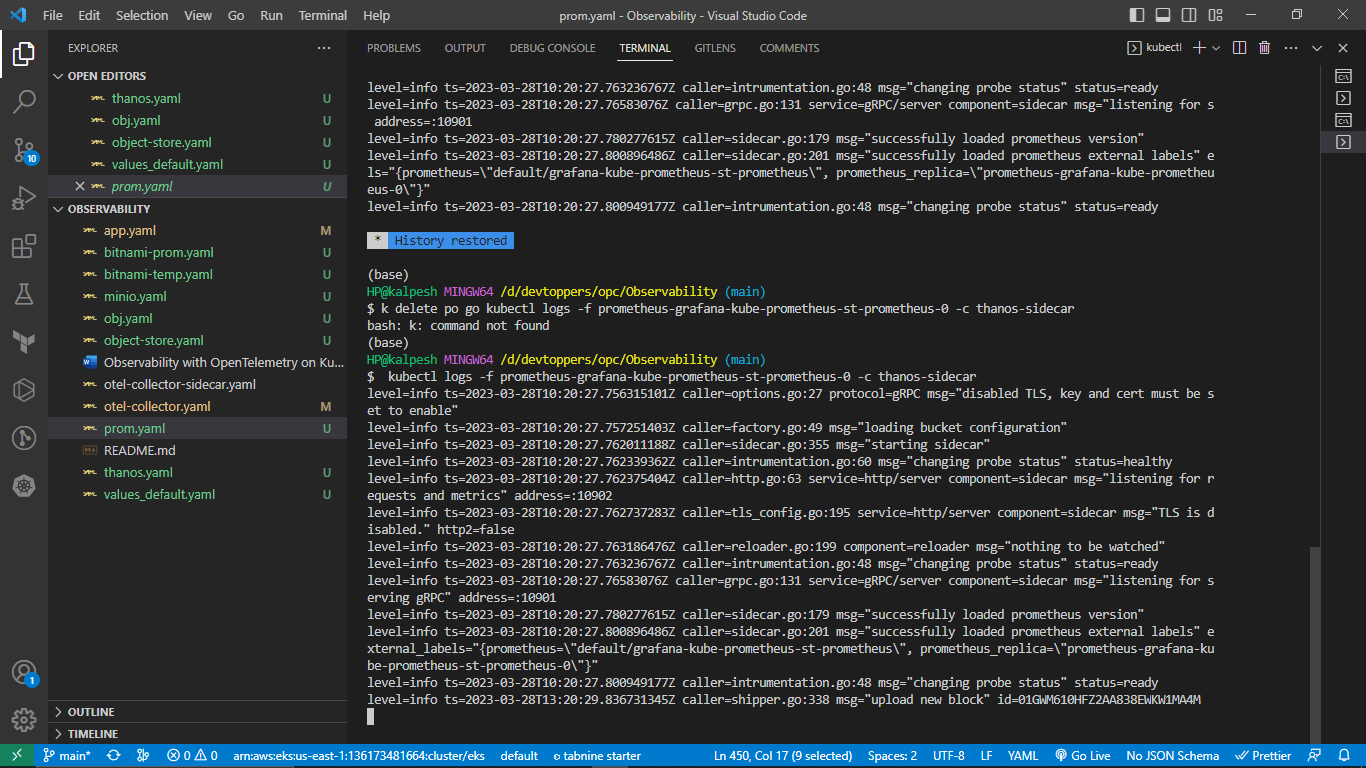
**Check logs of prometheus stack:**

kubectl logs –f [Prometheus stack pod\_name] –c thanos-sidecar

After applying above command will get result like below:

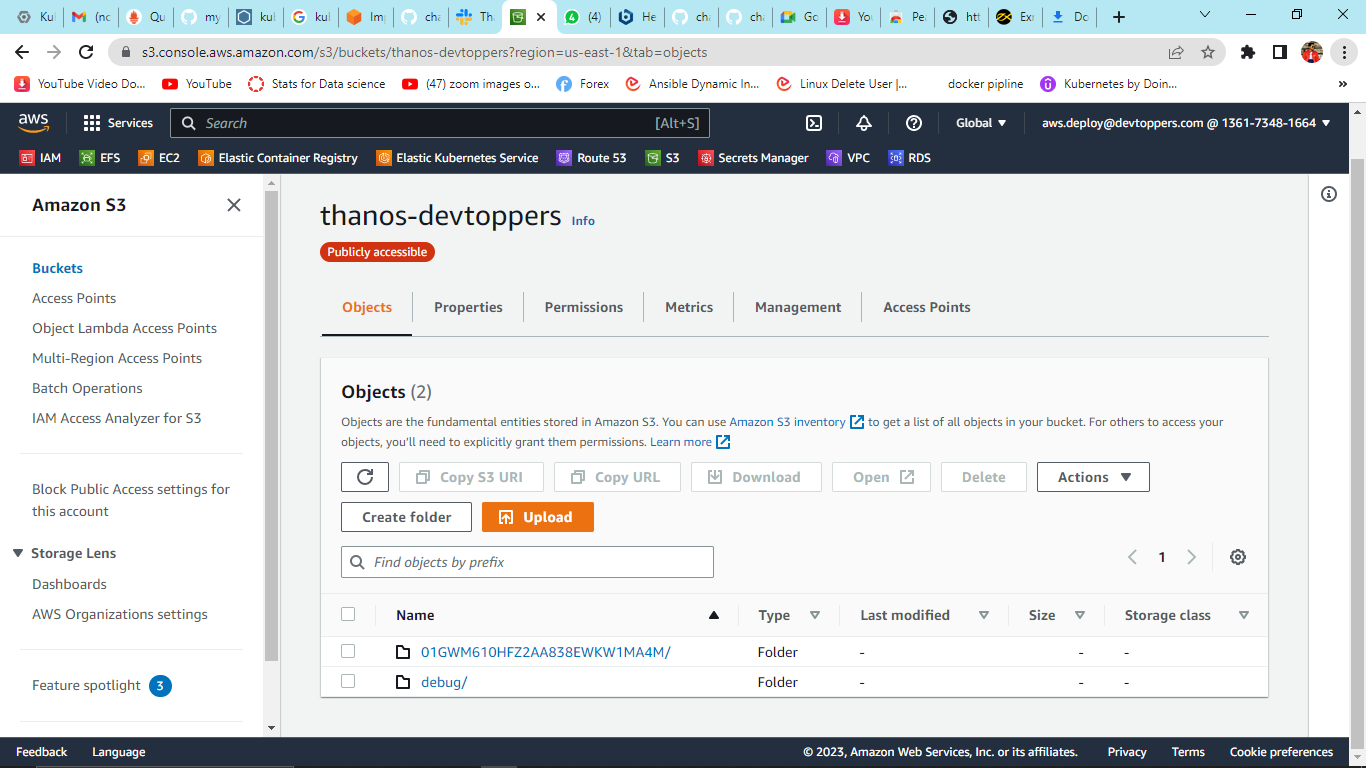


The maximum number of bytes of storage blocks to retain. The oldest data will be removed first. Defaults to 0 or disabled. Units supported: B, KB, MB, GB, TB, PB, EB. Ex: "512MB". Based on powers-of-2, so 1KB is 1024B. Only the persistent blocks are deleted to honor this retention although WAL and m-mapped chunks are counted in the total size. So the minimum requirement for the disk is the peak space taken by the wal (the WAL and Checkpoint) and chunks\_head (m-mapped Head chunks) directory combined (peaks every 2 hours).

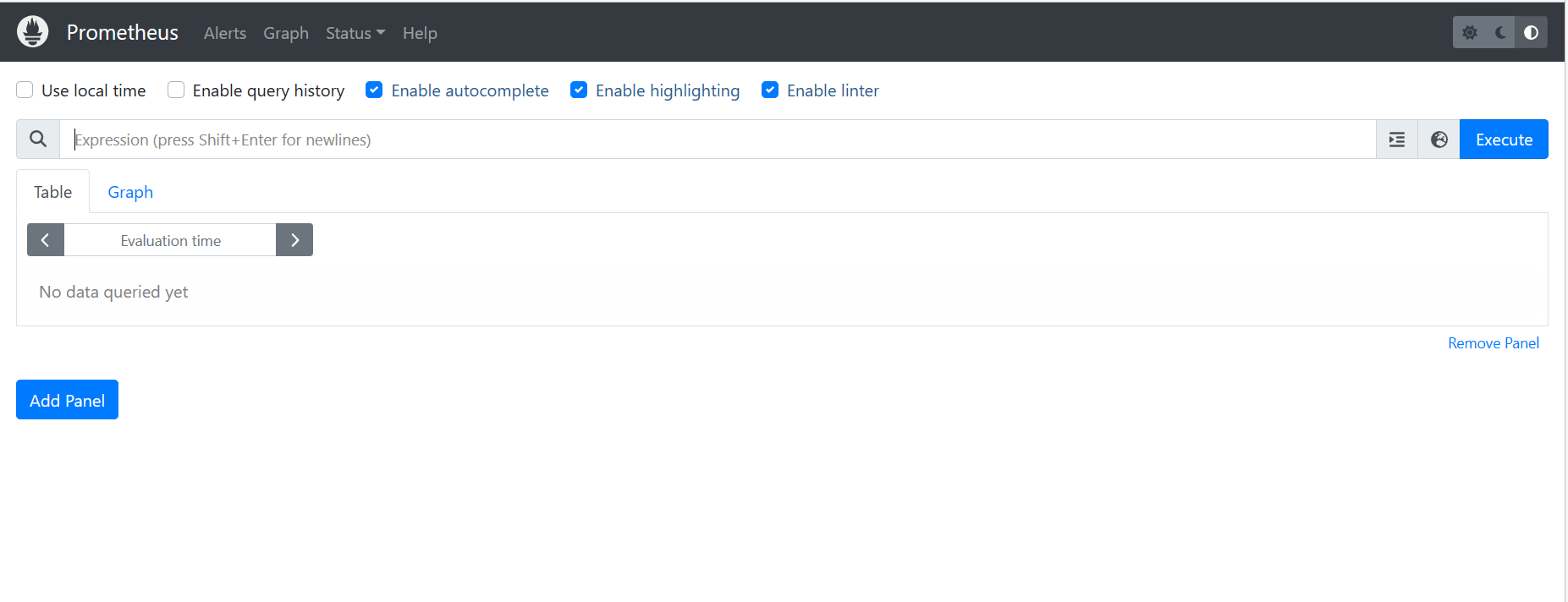


From the above image, thanos-sidecar will upload the data after every 3 hours and store the data in S3 bucket

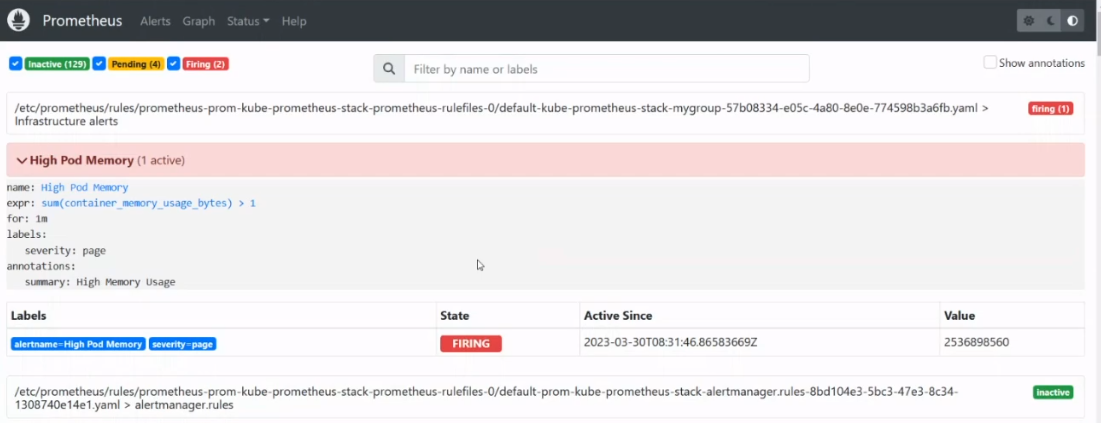
( See the timings from the above image ) and data will get upload in S3 bucket (See below image)



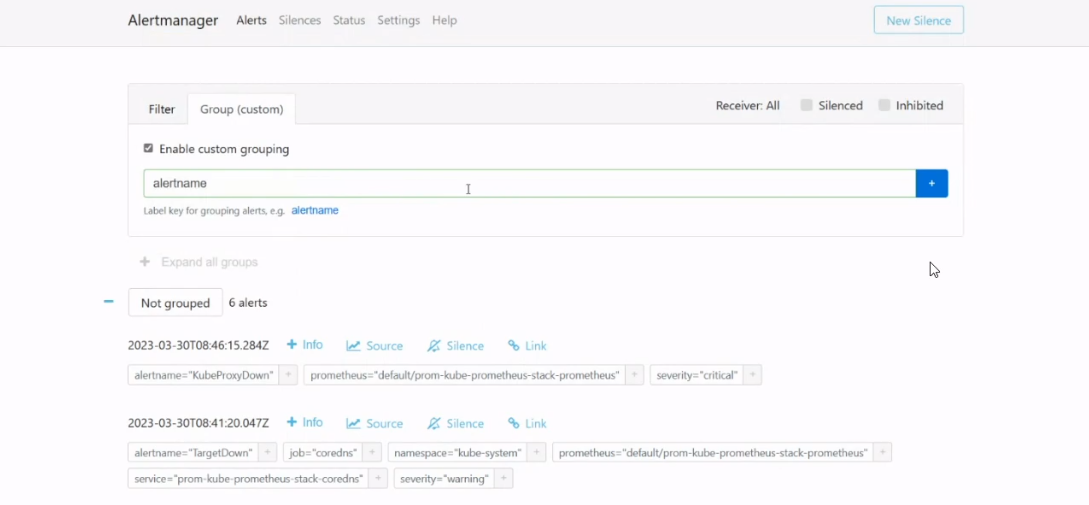
After successful chart deployment we will get the Prometheus dashboard like below by accesing its service

****

Click on Alerts and get the alertsrules that we have set in prometheus.yaml file

****

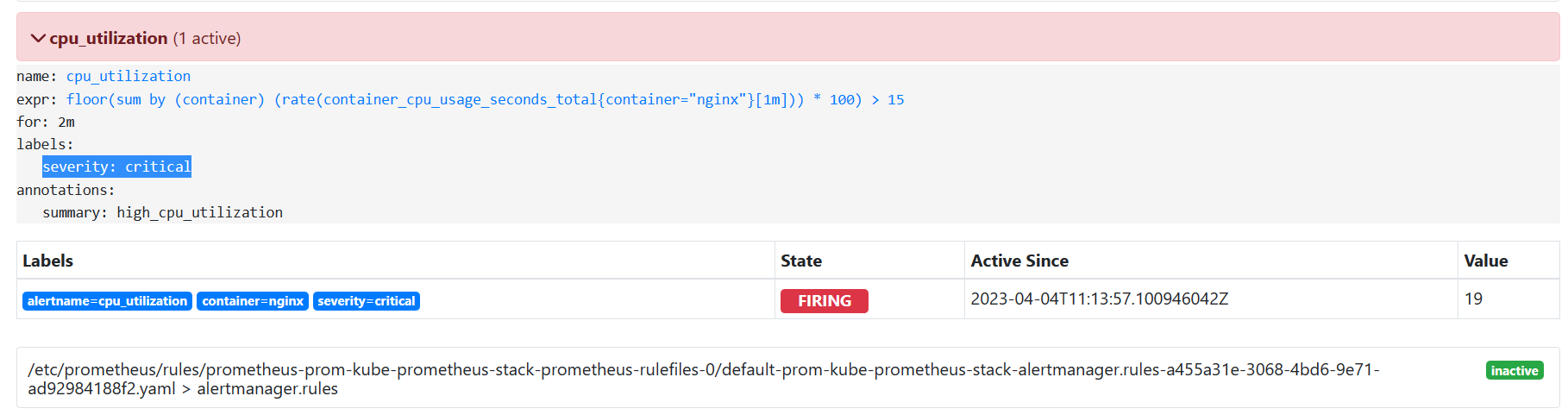
**OR** Access the Alertmanger dashboard by accessing its service to see the alerts in alerts section

****

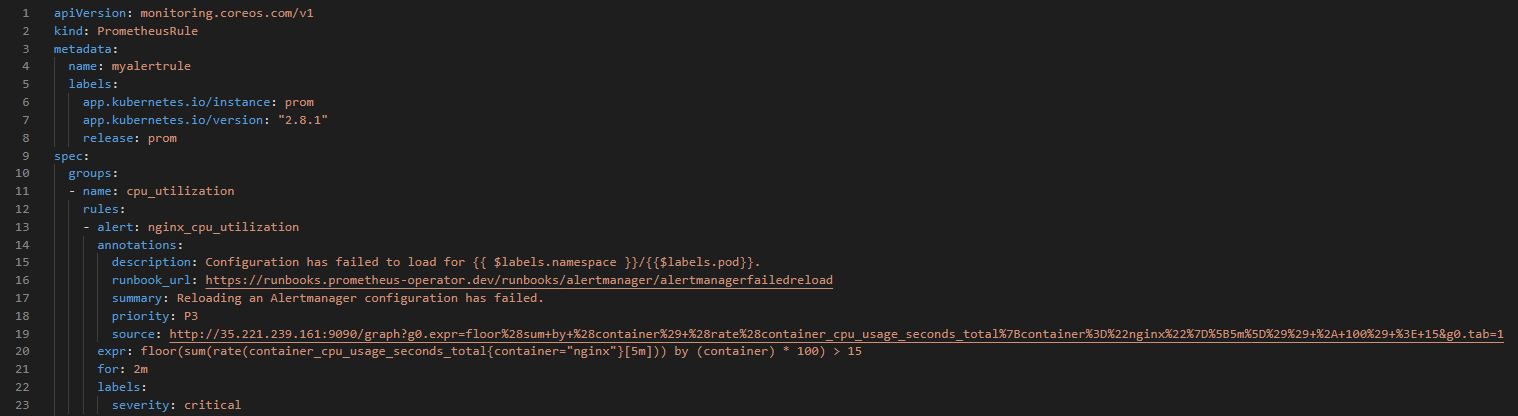
**Configuration of Alertmanager with Slack and Opsgenie:**

1. **To add custom Prometheus rules, In order to add the additional Prometheus rules section,**

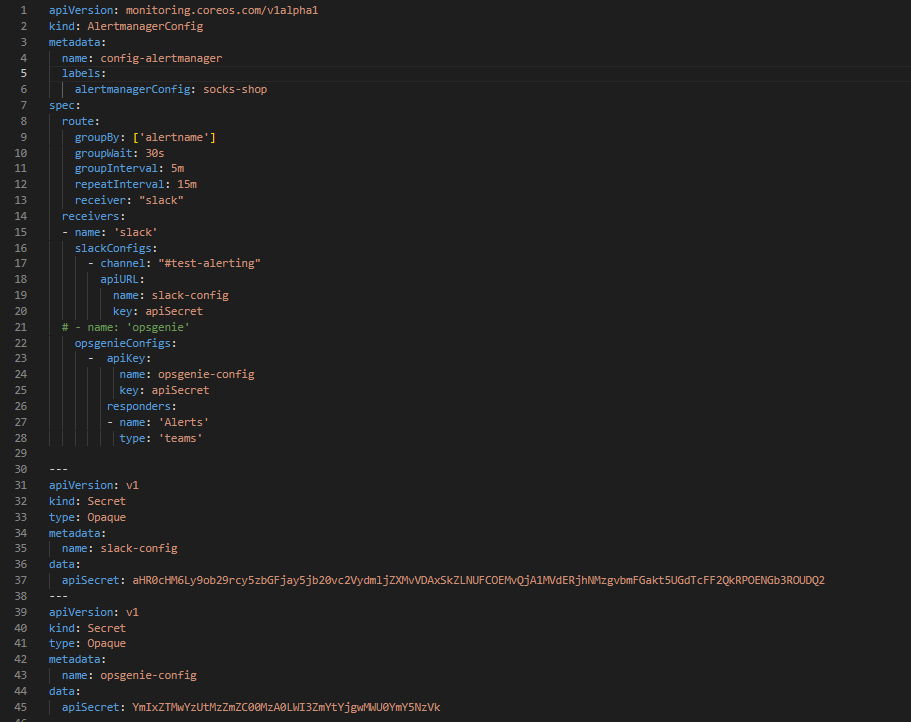
**set custom Prometheus rules**

****

**OR to make our own additional custom alert rules ,we can create CRD’s which is also a better option. For example,**

****

**To override the default alertmanager config we need to create an alertmanager config CRD in such a way that it will integrate alertmanager with slack and opsgenie as follows**

****

For **slack**,

1. Create a channel on slack

2. Create an app from **Manage Apps** > Click on **Create App** > Name the App > Mention created channel name > Copy the **Webhook URL** > Encode the URL from **base64** > Copy the the encoded value and paste in **apisecret.**

For **opsgenie**,

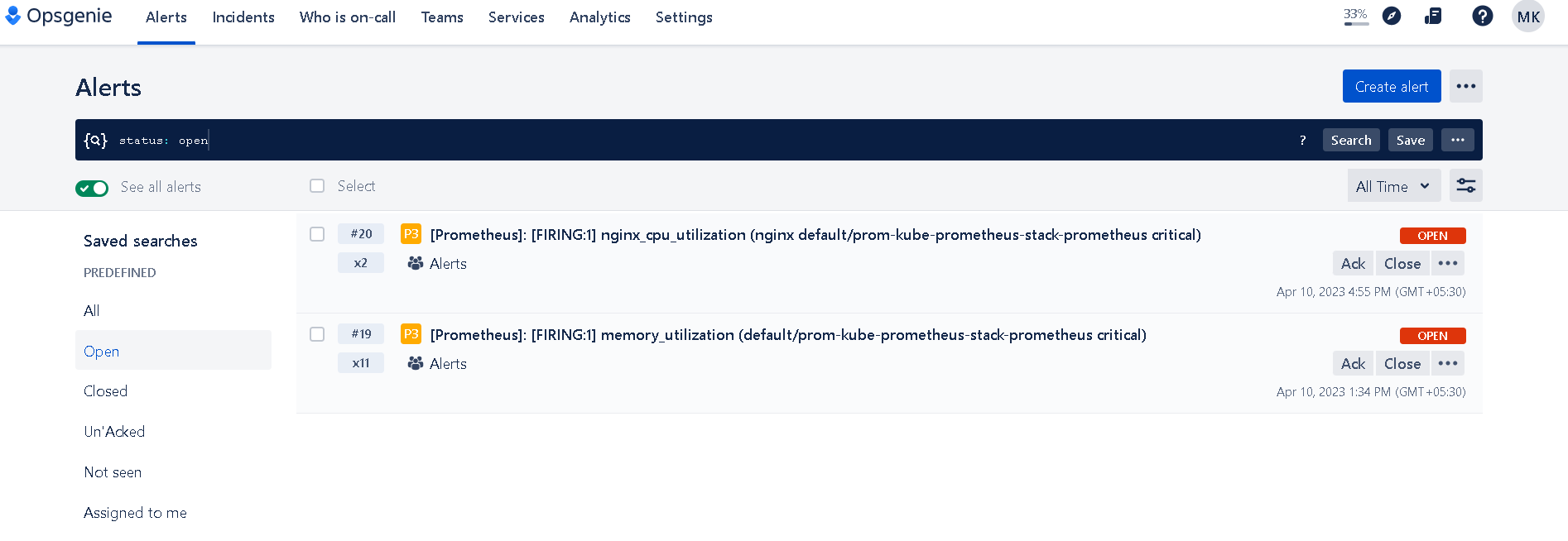
1. Create an account on Opsgenie

2. Select the product as **opsgenie cloud** > Click on **Teams** > Add teams>Give a team name >

Add the users for responder roles.

3. Click on your team > Click on **Add Integration** > Search and Select **Prometheus** > Copy the **api key** > Encode the api key from **base64** > Paste it in **apisecret**.

After sometime of groupinterval will receive the alerts in Alerts section



After applying above alertmanager crd config, we need mention above CRD’s name in alertmanager.spec.alertmanagerConfiguration Section of Kube-prometheus-stack helm chart.

1. For Demo purpose we have deployed nginx pod and for cpu\_utilization we have apply stress of 1 core cpu .
2. Once threshold limit reaches , it will fire an alerts and we can see in prometheus alerts section .