

# Let's talk about Kubernetes Cluster Monitoring

Using Prometheus & Grafana

Twinkl Sisodia  
Software Engineer  
Red Hat





"I am Twinkl Sisodia, Software Engineer at Red Hat. I work with organizations and partners to construct and control powerful metrics driven cloud native architectures."

—  
**Twinkl Sisodia**

Software Engineer at Red Hat

# Topics

- ▶ Monitoring and its importance
- ▶ Prometheus and its usage
- ▶ Grafana dashboards
- ▶ AlertManager to Slack
- ▶ Demo



**kubernetes**



Prometheus

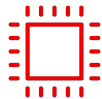


**Grafana**



**slack**

# IMPORTANCE OF MONITORING K8s



CPU

Resource Utilization approaching critical limits



MEMORY, STORAGE

Memory/Storage Utilization approaching critical limits



K8S Resources

Desired number of Nodes/Pods not running



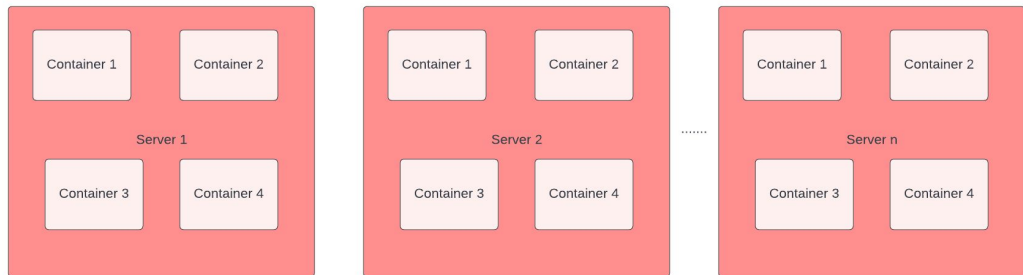
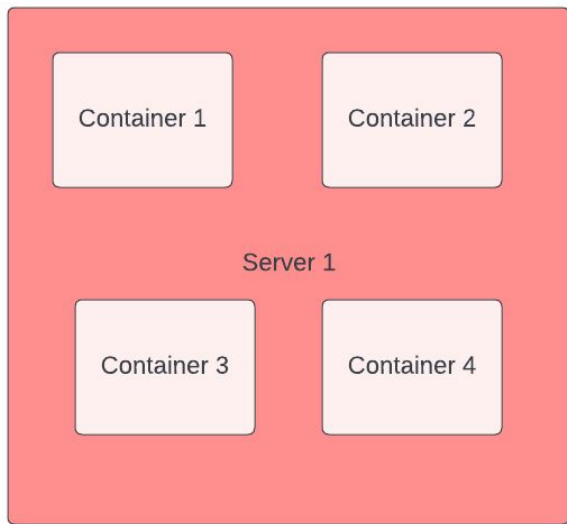
# Prometheus



## What is Prometheus?

**Prometheus** is an open-source systems monitoring and alerting toolkit originally built at [SoundCloud](https://soundcloud.com). Prometheus collects and stores its metrics as time series data, i.e. metrics information is stored with the timestamp at which it was recorded, alongside optional key-value pairs called labels.

# Infrastructure Example



# Prometheus Components

## PrometheusRules

Defines a desired set of Prometheus alerting and/or recording rules. The operator generates a rule file, which can be used by Prometheus instances

2

## ServiceMonitors

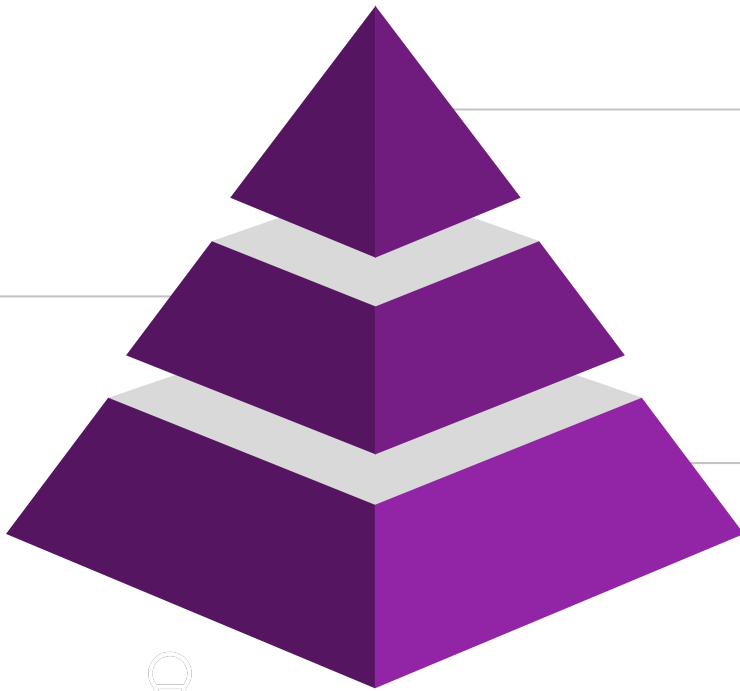
1

Specifies how groups of Kubernetes services should be monitored

## Alertmanager

3

Specifies subsections of the Alertmanager configurations, allowing routing of alerts to custom receivers, and setting inhibit rules



# ServiceMonitor

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: blue
  namespace: monitor
  labels:
    app: blue
spec:
  selector:
    matchLabels:
      app: blue
  namespaceSelector:
    any: true
    matchNames:
      - blue
  endpoints:
    - port: http
```



# PrometheusRules

CONFIDENTIAL designator

```
spec:
  groups:
    - name: recording_rules
      interval: 2s
      rules:
        - record: blue_requests_per_minute
          expr: increase(http_requests_total{container="blue"}[1m])
    - name: LoadRules
      rules:
        - alert: HighLoadBlue
          expr: blue_requests_per_minute >= 30
          labels:
            severity: page # or critical
          annotations:
            summary: "high load average"
            description: "high load average"
        - alert: MediumLoadBlue
          expr: blue_requests_per_minute >= 25
          labels:
            severity: warn
          annotations:
            summary: "medium load average"
            description: "medium load average"
        - alert: LowLoadBlue
          expr: blue_requests_per_minute >= 20
          labels:
            severity: acknowledged
          annotations:
            summary: "low load average"
            description: "low load average"
```

# AlertManagerConfig Secret

CONFIDENTIAL designator

devconf > ! alertmanager.yaml

```
1 route:
2   group_by: [alertname,cluster,service,job]
3   receiver: "slack"
4   repeat_interval: 1m
5   group_interval: 1m
6   group_wait: 10s
7   routes:
8   - match:
9     | severity: 'warn'
10    | receiver: "slack"
11  - match:
12    | severity: 'acknowledged'
13    | receiver: "slack"
14  - match:
15    | severity: 'page'
16    | receiver: "slack"
17 receivers:
18 - name: "slack"
19   slack_configs:
20   - api_url: 'https://hooks.slack.com/services/[REDACTED]'
21     channel: '#monitoring-alerts'
22     send_resolved: true
23 templates: []
```

V0000000



# Grafana



## What is Grafana?

**Grafana open source software** enables you to query, visualize, alert on, and explore your metrics, logs, and traces wherever they are stored. Grafana OSS provides you with tools to turn your time-series database (TSDB) data into insightful graphs and visualizations.



# Grafana Operator



## Grafana instance

Installs grafana using the default configuration and an Ingress or Route



## Datasource

Prometheus datasource



## Dashboard

Visualizations dashboard



# Grafana Datasource

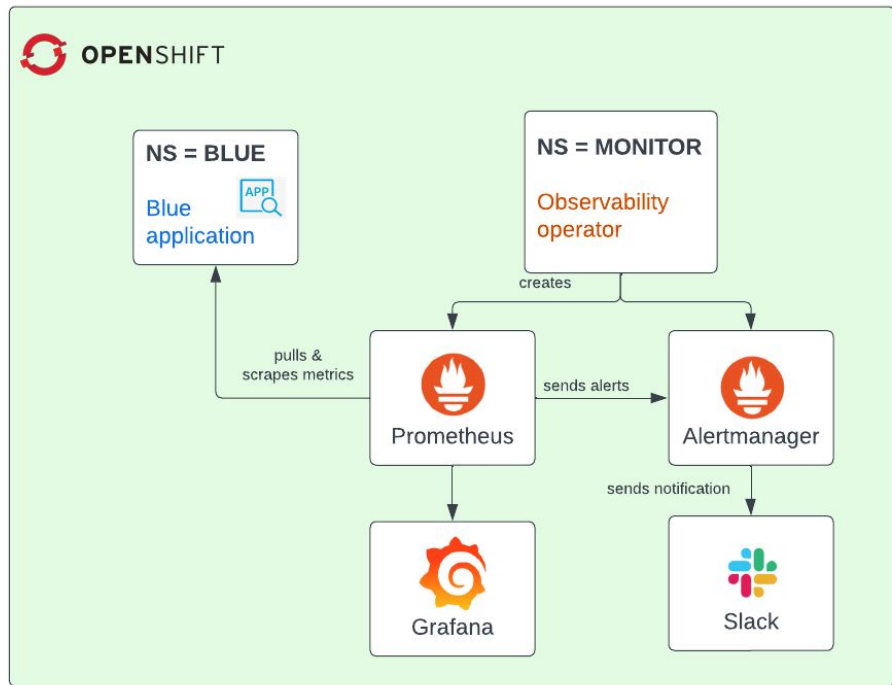
CONFIDENTIAL designator

```
devconf > ! grafana-ds.yaml
```

```
1  apiVersion: integreatly.org/v1alpha1
2  kind: GrafanaDataSource
3  metadata:
4    name: blue-grafanadatasource
5  spec:
6    name: middleware.yaml
7    datasources:
8      - name: Prometheus
9        type: prometheus
10       access: proxy
11       url: http://blue-prometheus:9090
12       isDefault: true
13       version: 1
14       editable: true
15       jsonData:
16         tlsSkipVerify: true
17         timeInterval: "5s"
```



# Architecture



# DEMO



## Monitoring Kubernetes

# Summary

- We talked about importance of monitoring
- Discussed about prometheus, grafana and its components
- In the demo, deployed an app
- Deployed observability operator and prometheus components and sent alerts to slack
- Deployed grafana operator and its components
- Lastly, custom dashboard to see insightful graphs



# Thank you



[www.linkedin.com/in/twinklisisodia](https://www.linkedin.com/in/twinklisisodia)



<https://github.com/tsisodia10>



[twinklisisodia10@gmail.com](mailto:twinklisisodia10@gmail.com)