Let’s talk about Kubernetes Cluster Monitoring

Using Prometheus & Grafana

Twinkll Sisodia
Software Engineer
Red Hat

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

Twinkll Sisodia
Software Engineer
Red Hat

LinkedIn : www.linkedin.com/in/twinkllsisodia
Email : twinkllsisodia10@gmail.com

Topics

- Monitoring and its importance
- Prometheus & its components
- Grafana & its components
- Demo

kubernetes

Prometheus

Grafana

slack

IMPORTANCE OF MONITORING

CPU

Resource Utilization approaching critical limits

MEMORY, STORAGE

Memory/Storage Utilization approaching critical limits

K8S Resources

Desired number of Nodes/Pods not running

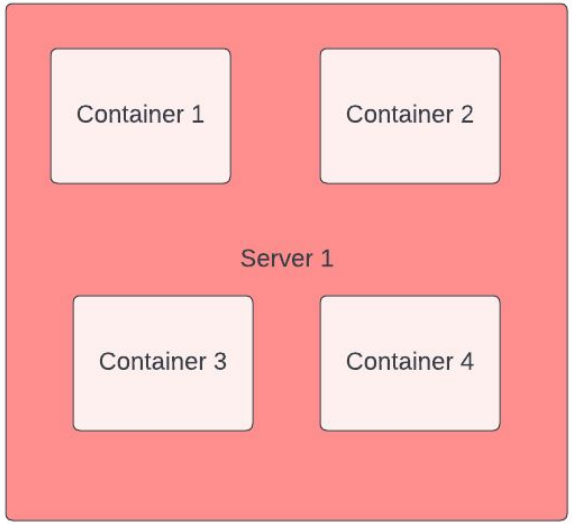
Prometheus

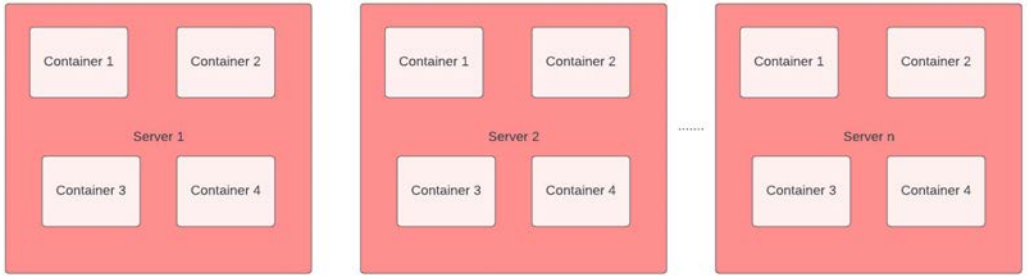
What is Prometheus?

Prometheus is an open-source systems monitoring and alerting toolkit originally built at SoundCloud. 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.

Source:
https://prometheus.io/docs/introduction/overview/

Infrastructure Example





Source:
https://prometheus.io/docs/introduction/overview/

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

ServiceMonitors

Specifies how groups of Kubernetes services should be monitored

Alertmanager

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

ServiceMonitor


```yaml
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


```yaml
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

devconf > ! alertmanager.yaml

```yaml
route:
  group_by: [alertname,cluster,service,job]
  receiver: "slack"
  repeat_interval: 1m
  group_interval: 1m
  group_wait: 10s
  routes:
  - match:
      severity: 'warn'
    receiver: "slack"
  - match:
      severity: 'acknowledged'
    receiver: "slack"
  - match:
      severity: 'page'
    receiver: "slack"
receivers:
- name: "slack"
  slack_configs:
  - api_url: 'https://hooks.slack.com/services/[redacted]'
    channel: '#monitoring-alerts'
    send_resolved: true
templates: []
```

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

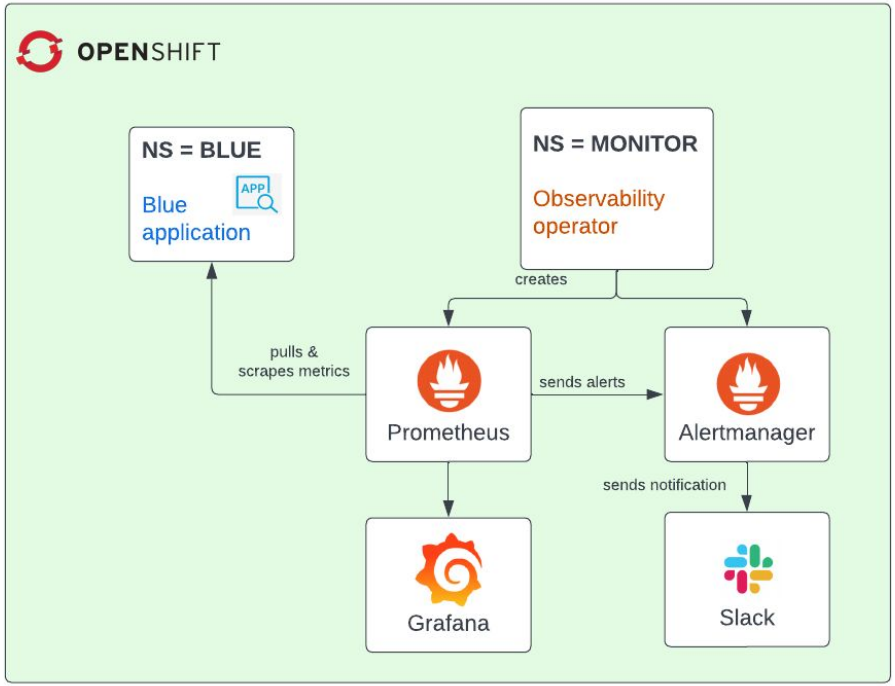
Source:
https://github.com/grafana-operator/grafana-operator/tree/master/documentation

Grafana Datasource

devconf > ! grafana-ds.yaml

```yaml
apiVersion: integreatly.org/v1alpha1
kind: GrafanaDataSource
metadata:
  name: blue-grafanadatasource
spec:
  name: middleware.yaml
  datasources:
    - name: Prometheus
      type: prometheus
      access: proxy
      url: http://blue-prometheus:9090
      isDefault: true
      version: 1
      editable: true
      jsonData:
        tlsSkipVerify: true
        timeInterval: "5s"
```

Architecture



DEMO

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

Scan for LinkedIn

Scan for GitHub repo

Red Hat