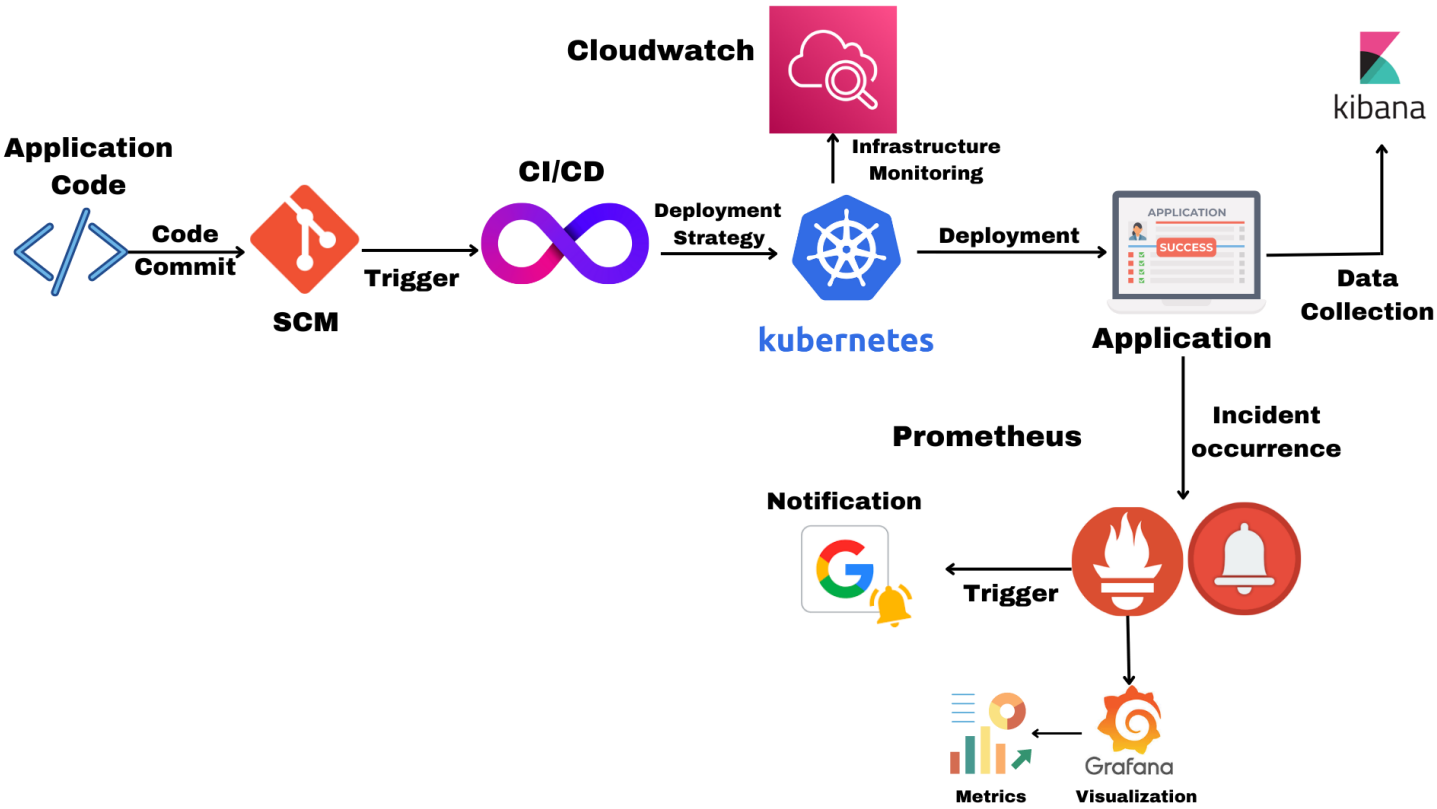
DevOps Shack

High level Monitoring in Devops

Monitoring is a crucial aspect of the DevOps lifecycle, ensuring that the infrastructure, applications, and services perform optimally and that any issues are quickly identified and resolved. Effective monitoring allows teams to gain insights into system performance, application health, and user experience, ultimately leading to more reliable and efficient software delivery.



In this document, we will explore how to set up and configure some of the most widely used monitoring tools in DevOps: AWS CloudWatch, Prometheus, Alertmanager, Grafana, and Kibana. Each tool serves a specific purpose in the monitoring ecosystem, from collecting and aggregating metrics to visualizing data and managing alerts.

Tools Used

1. **AWS CloudWatch**

Purpose: Monitor and manage AWS resources and applications.

Key Features:

Provides real-time monitoring of AWS services.

Allows setting alarms and automating responses based on metric thresholds.

Integrates seamlessly with other AWS services.

2. **Prometheus**

Purpose: An open-source system monitoring and alerting toolkit.

Key Features:

Efficient at collecting and storing time-series data.

Offers a powerful query language (PromQL) for analyzing metrics.

Ideal for monitoring cloud-native applications, especially those using microservices.

3. **Alertmanager**

Purpose: Handles alerts sent by Prometheus and manages notifications.

Key Features:

Routes alerts to the correct receiver based on defined rules.

Supports silencing, inhibition, and grouping of alerts.

Integrates with various notification platforms like email, Slack, and PagerDuty.

4. **Grafana**

Purpose: An open-source platform for monitoring and observability.

Key Features:

Allows creating customizable and interactive dashboards.

Supports a wide range of data sources, including Prometheus, Elasticsearch, and more.

Provides alerting features with the ability to visualize alerts on dashboards.

5. **Kibana**

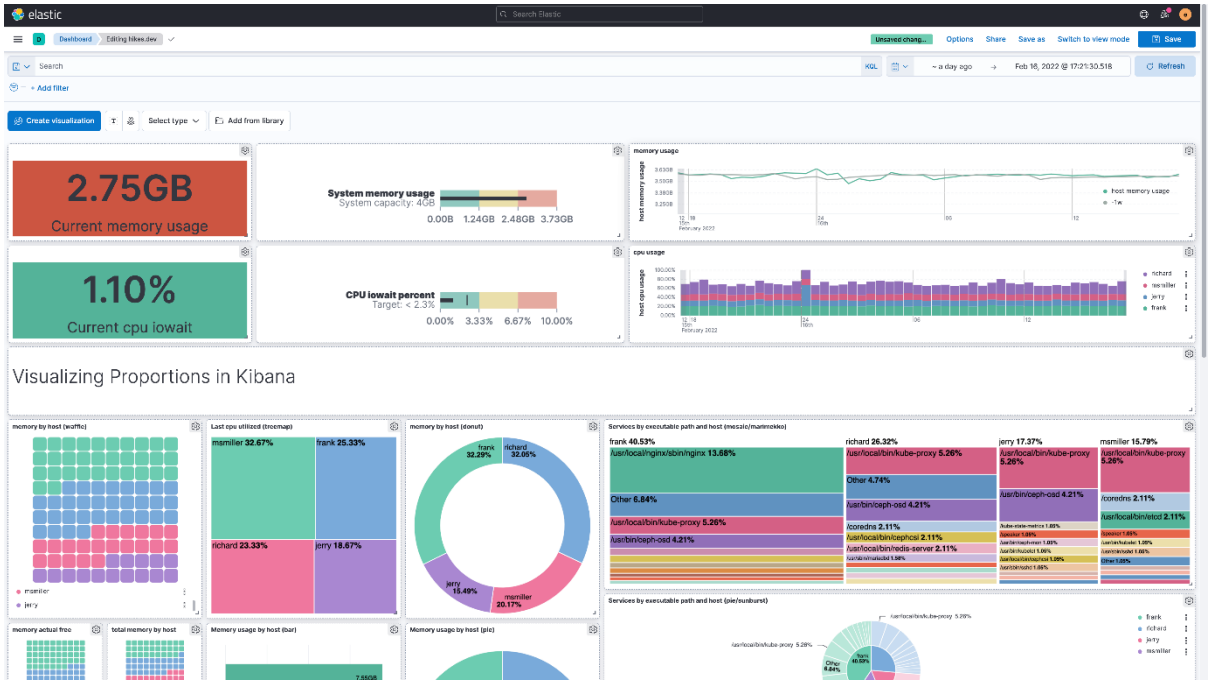
Purpose: A visualization and Data collection tool to explore tool for Elasticsearch.

Key Features:

Specializes in log and time-series analytics.

Enables users to create visualizations and dashboards for Elasticsearch data.

Includes features for searching, viewing, and interacting with data stored in Elasticsearch.



Kibana Dashboard view

Setup Instructions

1. AWS CloudWatch

Setup Steps:

Create an IAM Role for CloudWatch:

```
aws iam create-role --role-name CloudWatchRole --assume-role-policy-document file://trust-policy.json
```

attach CloudWatch Policy:

```
aws iam attach-role-policy --role-name CloudWatchRole --policy-arn arn:aws:iam::aws:policy/CloudWatchFullAccess
```

Create a CloudWatch Log Group:

```
aws logs create-log-group --log-group-name MyLogGroup
```

Create a CloudWatch Alarm:

```
aws cloudwatch put-metric-alarm --alarm-name "High CPU Usage" --metric-name CPUUtilization --namespace AWS/EC2 --statistic Average --period 300 --threshold 80 --comparison-operator GreaterThanOrEqualToThreshold --evaluation-periods 2 --alarm-actions arn:aws:sns:us-east-1:123456789012:MyTopic
```

2. Prometheus

Setup Steps:

Install Prometheus:

```
wget https://github.com/prometheus/prometheus/releases/download/v2.32.1/prometheus-2.32.1.linux-amd64.tar.gz
```

```
tar -xvf prometheus-2.32.1.linux-amd64.tar.gz
```

```
cd prometheus-2.32.1.linux-amd64
```

Configure Prometheus:

Edit the prometheus.yml file to configure Prometheus:

```
global:
  scrape_interval: 15s
scrape_configs:
  - job_name: 'prometheus'
    static_configs:
      - targets: ['localhost:9090']
```

Start Prometheus:

```
./prometheus --config.file=prometheus.yml
```

3. Alertmanager

Setup Steps:

Install Alertmanager:

```
wget https://github.com/prometheus/alertmanager/releases/download/v0.23.0/alertmanager-0.23.0.linux-amd64.tar.gz
tar -xvf alertmanager-0.23.0.linux-amd64.tar.gz
cd alertmanager-0.23.0.linux-amd64
```

Configure Alertmanager:

Edit the alertmanager.yml file to set up alerting rules:

```
global:
```

```
  resolve_timeout: 5m
route:
  receiver: 'email-alert'
receivers:
  - name: 'email-alert'
    email_configs:
      - to: 'your-email@example.com'
        from: 'alertmanager@example.com'
        smarthost: 'smtp.example.com:587'
        auth_username: 'your-username'
        auth_password: 'your-password'
```

Start Alertmanager:

```
./alertmanager --config.file=alertmanager.yml
```

4. Grafana

Setup Steps:

Install Grafana:

```
wget https://dl.grafana.com/oss/release/grafana-8.3.3.linux-amd64.tar.gz
tar -zxvf grafana-8.3.3.linux-amd64.tar.gz
cd grafana-8.3.3
```

Start Grafana Server:

```
./bin/grafana-server
```

Access Grafana:

Open your browser and go to http://localhost:3000. The default login is admin/admin.

Add Prometheus as a Data Source:

Go to Configuration > Data Sources.

Select Prometheus.

Enter the URL of your Prometheus server, e.g., http://localhost:9090.

5. Kibana

Setup Steps:

Install Kibana:

```
wget https://artifacts.elastic.co/downloads/kibana/kibana-7.16.2-linux-x86_64.tar.gz
tar -xzf kibana-7.16.2-linux-x86_64.tar.gz
cd kibana-7.16.2-linux-x86_64
```

Configure Kibana:

Edit the kibana.yml file to connect Kibana to your Elasticsearch instance:

```
server.port: 5601
elasticsearch.hosts: ["http://localhost:9200"]
```

Start Kibana:

./bin/kibana

Access Kibana:

Open your browser and go to http://localhost:5601.

Conclusion

By setting up these tools, you've created a robust monitoring environment that allows you to track and visualize the performance of your applications and infrastructure in real-time. Each tool plays a vital role in ensuring the reliability and scalability of your systems, making it easier to detect issues early and respond swiftly.