# Prometheus & Grafana: Practical 🚀





#### What are Prometheus and Grafana?

**Prometheus** (1) is an open-source **monitoring and alerting toolkit** designed for cloud-native applications. It collects, stores, and gueries **metrics**—numerical data that represents system performance. Using a pull-based model, Prometheus retrieves data from targets (such as servers, containers, or applications) at regular intervals. This data is stored in its time-series database, which can be queried using **PromQL** to analyze performance trends or detect anomalies.

**Grafana** complements Prometheus by providing **customizable dashboards** for data visualization. While Prometheus handles data collection and querying, Grafana focuses on presenting the data in a clear, visually appealing way. It also enables users to set alerts based on metrics, helping teams respond proactively to potential issues.

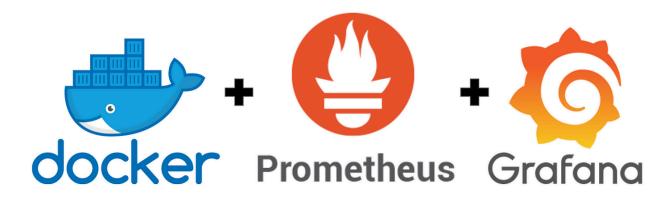
# Why Use Prometheus and Grafana for Alerting? 🤔



- \* Real-Time Insights: Monitor critical metrics in real-time, catching issues before they escalate.
- **Custom Alerts**: Configure alerts for specific conditions, ensuring no anomaly goes unnoticed.
- Seamless Integration: Prometheus and Grafana integrate effortlessly, creating a unified monitoring and alerting solution.
- Scalability: Both tools are highly scalable, suitable for small setups and enterprise-level deployments.

# **Practical: Monitoring Vulnerabilities on Docker Containers**

We will now set up Prometheus and Grafana in **Docker containers** to monitor a vulnerable application. This setup will demonstrate how vulnerabilities in a containerized application can be detected and visualized.



# Step 1: Create a Docker Network

Create a Docker network to ensure communication between Prometheus. Grafana, and the vulnerable container:

## What's Happening?

- This creates an isolated **Docker network** called monitoring-net.
- Containers added to this network can communicate with each other while remaining isolated from other networks.

# Step 2: Set Up Prometheus in Docker 🕒

## 1. Create Prometheus Configuration File

Create a file named prometheus.yml in your working directory with the following content:

```
global:
    scrape_interval: 5s

scrape_configs:
    - job_name: 'docker-target'
    static_configs:
    - targets: ['vulnerable-app:8080']
```

## **+** Explanation:

- scrape\_interval: Defines how frequently Prometheus collects metrics (5 seconds here).
- static\_configs: Specifies the endpoint (vulnerable-app:8080) where metrics will be scraped.

#### 2. Run Prometheus in Docker

```
docker run -d \
```

```
--name=prometheus \
--network=monitoring-net \
-p 9090:9090 \
-v $(pwd)/prometheus.yml:/etc/prometheus/prometheus.yml \
prom/prometheus
```

## What's Happening?

- --network=monitoring-net: Connects Prometheus to the monitoring-net network.
- -v: Mounts the configuration file to Prometheus' configuration directory.
- -p 9090:9090: Exposes Prometheus on port 9090.

# Step 3: Set Up Grafana in Docker 📊

Run Grafana as a Docker container:

```
docker run -d \
    --name=grafana \
    --network=monitoring-net \
    -p 3000:3000 \
    grafana/grafana
```

## What's Happening?

- This starts Grafana and connects it to the same Docker network as Prometheus.
- Grafana is accessible at http://localhost:3000.

# Step 4: Deploy a Vulnerable Application 🐞

Let's deploy a vulnerable container (e.g., vulnweb) for monitoring:

```
docker run -d \
    --name=vulnerable-app \
    --network=monitoring-net \
    -p 8080:8080 \
    vulnerables/web-dvwa
```

## What's Happening?

- This starts the **Damn Vulnerable Web Application (DVWA)** on port 8080.
- The container is added to the monitoring-net network, making it accessible to Prometheus.

# Step 5: Configure Prometheus as a Data Source in Grafana

- 1. Login to Grafana:
  - o Open http://localhost:3000 in your browser.
  - Default credentials:
    - Username: admin
    - Password: admin.
- 2. Add Prometheus as a Data Source:
  - Navigate to Configuration > Data Sources in Grafana.
  - Select Prometheus from the list.
  - Provide the Prometheus URL (e.g., http://prometheus:9090).

## Step 6: Set Up Alerts for Vulnerabilities 🚨

#### 1. Create a Dashboard:

- In Grafana, click Create > Dashboard.
- Add a panel and use PromQL queries (e.g., rate(http\_requests\_total[5m])) to visualize traffic spikes or anomalies.

#### 2. Define Alerts:

- In the panel editor, go to the **Alert** tab.
- Create a condition (e.g., **abnormally high error rates**).
- Configure notification channels like email or Slack.

## What's Happening?

 Alerts monitor container behavior for unusual patterns or excessive requests, which could indicate exploitation attempts.

#### **Conclusion**

You've set up **Prometheus and Grafana on Docker to monitor a vulnerable application.** This configuration not only demonstrates **real-time alerting** but also highlights the importance of monitoring containerized environments for potential risks. Use this setup to explore further metrics or **secure vulnerabilities** in your applications.

# **Follow ZORAIZ for More:)**