Atelier 9 : Supervision et Logs

Objectifs de l'atelier :

1. Mettre en place **Prometheus** et **Grafana** via Docker Compose pour surveiller les systèmes.
2. Utiliser **Node Exporter** pour superviser les métriques système de la VM **Vagrant**.
3. Configurer une **stack ELK (Elasticsearch, Logstash, Kibana)** pour la gestion centralisée des logs.
4. Configurer **Filebeat** pour envoyer les logs Nginx de la VM vers **Logstash** et **Elasticsearch**.

Exercice 1 : Mettre en place Prometheus et Grafana via Docker Compose (45 min)

Objectif : Déployer Prometheus et Grafana en utilisant Docker Compose pour collecter et visualiser les métriques système.

Étapes :

1. **Créer un répertoire pour le projet** :

```
mkdir monitoring-stack
cd monitoring-stack
```

2. **Créer le fichier `docker-compose.yml` pour Prometheus et Grafana** :

```
version: '3'

services:
  prometheus:
    image: prom/prometheus
    container_name: prometheus
    ports:
      - "9090:9090"
    volumes:
      - ./prometheus.yml:/etc/prometheus/prometheus.yml
    networks:
      - monitoring

  grafana:
    image: grafana/grafana
    container_name: grafana
    ports:
      - "3000:3000"
    networks:
      - monitoring
    environment:
      GF_SECURITY_ADMIN_PASSWORD: 'admin'
```

```
networks:
  monitoring:
    driver: bridge
```

3. **Créer le fichier `prometheus.yml` pour la configuration de Prometheus** :
   Ce fichier permettra à Prometheus de collecter des métriques depuis **Node Exporter** (qui sera déployé dans un autre exercice).

```
global:
  scrape_interval: 15s

scrape_configs:
  - job_name: 'node'
    static_configs:
      - targets: ['node_exporter:9100']
```

4. **Démarrer Docker Compose** :

```
docker-compose up -d
```

5. **Accéder à Prometheus et Grafana** :
   - Ouvrez votre navigateur et accédez à **Prometheus** sur http://localhost:9090.
   - Accédez à **Grafana** sur http://localhost:3000 avec le mot de passe **admin**.
6. **Configurer Grafana pour se connecter à Prometheus** :
   - Allez dans **Configuration > Data Sources**.
   - Sélectionnez **Prometheus**, puis configurez l'URL : `http://prometheus:9090`.
7. **Vérifier les métriques sur Grafana** :
   - Créez un dashboard dans Grafana et ajoutez un graphique en utilisant des requêtes comme `node_cpu_seconds_total` pour visualiser les métriques système.

Exercice 2 : Utiliser le Node Exporter pour superviser les métriques système de la VM Vagrant (45 min)

Objectif : Déployer le **Node Exporter** sur une VM Vagrant pour collecter des métriques système et les envoyer à Prometheus.

Étapes :

1. **Vérifier que la VM Vagrant est en place** (si ce n'est pas déjà fait, créer une VM avec une distribution Linux comme Ubuntu).
2. **Installer le Node Exporter sur la VM Vagrant** :
   - Connectez-vous à la VM via SSH :

     ```
     vagrant ssh
     ```

   - Téléchargez et installez le **Node Exporter** :

```
wget
https://github.com/prometheus/node_exporter/releases/download/v1.2.2
/node_exporter-1.2.2.linux-amd64.tar.gz
tar -xvzf node_exporter-1.2.2.linux-amd64.tar.gz
cd node_exporter-1.2.2.linux-amd64
sudo ./node_exporter &
```

3. **Configurer Prometheus pour collecter les métriques depuis la VM** :
   - Modifiez le fichier `prometheus.yml` pour ajouter l'adresse IP de la VM Vagrant (ex. : `192.168.33.10:9100`).

```
scrape_configs:
  - job_name: 'node'
    static_configs:
      - targets: ['192.168.33.10:9100']
```

   - Relancez Prometheus pour prendre en compte la modification :

```
docker-compose restart prometheus
```

4. **Vérifier dans Grafana** :
   - Sur Grafana, ajoutez un **panel** pour afficher les métriques envoyées par **Node Exporter**, telles que `node_memory_MemTotal_bytes` ou `node_cpu_seconds_total`.

Exercice 3 : Mettre en place une stack ELK simple via Docker Compose (45 min)

Objectif : Déployer une stack **ELK (Elasticsearch, Logstash, Kibana)** avec Docker Compose pour collecter, analyser et visualiser les logs.

Étapes :

1. **Créer le fichier `docker-compose.yml` pour la stack ELK** :

```
version: '3'

services:
  elasticsearch:
    image: docker.elastic.co/elasticsearch/elasticsearch:7.10.2
    container_name: elasticsearch
    environment:
      - discovery.type=single-node
    ports:
      - "9200:9200"
    networks:
      - elk

  logstash:
    image: docker.elastic.co/logstash/logstash:7.10.2
    container_name: logstash
    ports:
```

```
      - "5044:5044"
    volumes:
      - ./logstash.conf:/usr/share/logstash/pipeline/logstash.conf
    networks:
      - elk

  kibana:
    image: docker.elastic.co/kibana/kibana:7.10.2
    container_name: kibana
    ports:
      - "5601:5601"
    networks:
      - elk

networks:
  elk:
    driver: bridge
```

2. **Créer le fichier `logstash.conf` pour Logstash** :
   Ce fichier définit comment Logstash traite les logs et les envoie à Elasticsearch.

```
input {
  beats {
    port => 5044
  }
}

filter {
  grok {
    match => { "message" => "%{COMMONAPACHELOG}" }
  }
}

output {
  elasticsearch {
    hosts => ["elasticsearch:9200"]
    index => "logs-%{+YYYY.MM.dd}"
  }
}
```

3. **Démarrer la stack ELK** :

```
docker-compose up -d
```

4. **Accéder à Kibana** :
   - Ouvrez votre navigateur et allez sur http://localhost:5601.
   - Créez un index pattern pour visualiser les logs dans Kibana (par exemple, `logs-*`).

Exercice 4 : Configurer Filebeat pour envoyer les logs Nginx vers Logstash/Elasticsearch (45 min)

Objectif : Utiliser **Filebeat** pour envoyer les logs Nginx vers Logstash, puis vers Elasticsearch.

Étapes :

1. **Installer Filebeat sur la VM Vagrant** :
   - Connectez-vous à la VM via SSH :

```
vagrant ssh
```

   - Installez **Filebeat** (en fonction de votre système d'exploitation) :

```
sudo apt-get install filebeat
```

2. **Configurer Filebeat pour envoyer les logs vers Logstash** :
   - Modifiez le fichier de configuration de Filebeat (`filebeat.yml`) pour envoyer les logs Nginx vers **Logstash** :

```
filebeat.inputs:
  - type: log
    enabled: true
    paths:
      - /var/log/nginx/*.log

output.logstash:
  hosts: ["localhost:5044"]
```

3. **Démarrer Filebeat** :

```
sudo systemctl start filebeat
sudo systemctl enable filebeat
```

4. **Vérifier dans Kibana** :
   - Accédez à Kibana et recherchez les logs Nginx envoyés via Filebeat. Vous pouvez utiliser **Discover** dans Kibana pour voir les logs en temps réel.

Conclusion de l'atelier :

- Vous avez mis en place une solution de **supervision** avec **Prometheus** et **Grafana** pour surveiller les métriques système.
- Vous avez créé une **stack ELK** pour gérer et visualiser les logs.
- Vous avez utilisé **Filebeat** pour collecter les logs Nginx et les envoyer vers **Logstash** et **Elasticsearch**.