

# Déploiement d'une infrastructure web NGINX supervisée avec Prometheus et Grafana.

# SOMMAIRE

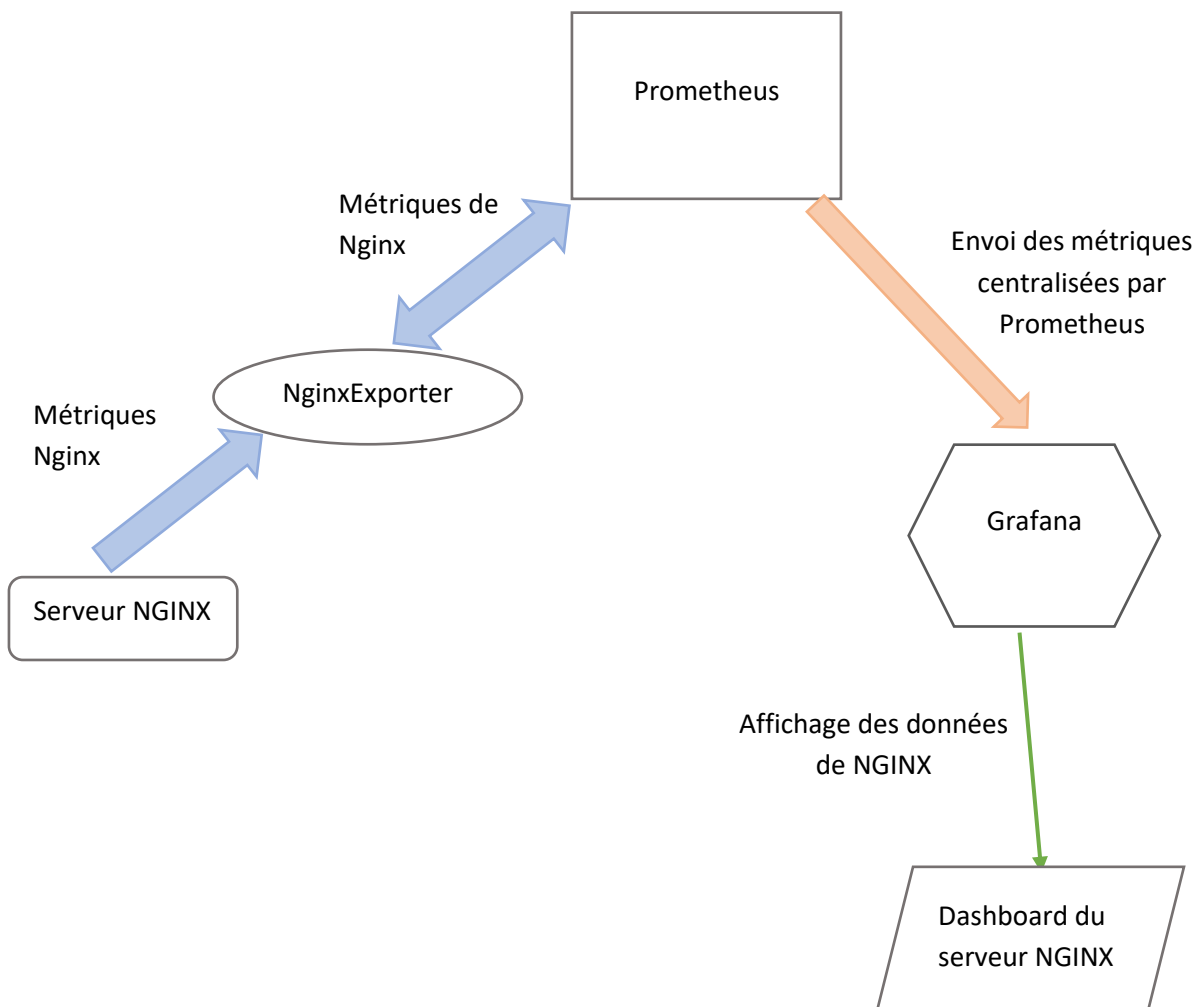
---

- I. Introduction
- II. Le service Prometheus
- III. Le service Grafana
- IV. Le service web Nginx
- V. Le service Exporter : Nginx Exporter
- VI. Quelques captures pour le résultat

# Introduction

---

Le monitoring est aussi appelé surveillance informatique. Son but est de s'assurer de la disponibilité et de la performance des infrastructures, des équipements, des logiciels et des processus supportant les données. Cette infrastructure est mise en place avec des conteneurs Docker.



# Le service Prometheus

Prometheus est un logiciel libre de surveillance informatique et générateur (un outil de monitoring polyvalent) essentiel pour surveiller la santé de ses systèmes et applications.

Pour le déploiement du service Prometheus sous Docker, la méthode retenue est l'utilisation d'un fichier docker-compose. Dans ce fichier (ci-dessous), on récupère l'image officielle de Prometheus depuis la bibliothèque Docker et on nomme le conteneur « prometheus ».

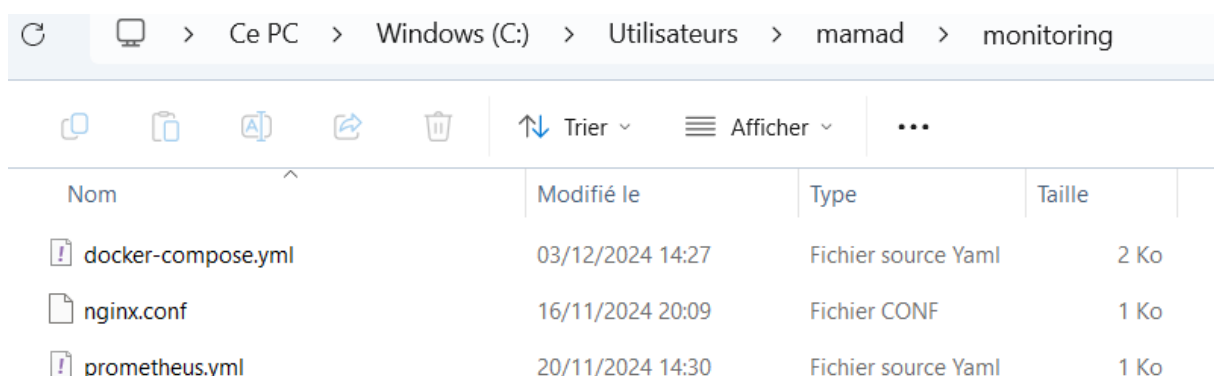
Puis, dans le fichier docker-compose, on peut ajouter d'autres éléments, comme par exemple, dans notre cas, une catégorie « volumes » qui crée un montage de répertoire entre le stockage de la machine hôte et celui du conteneur.




Pour terminer, on indique le port sur lequel écoutera le conteneur Prometheus (9090 dans notre cas) en « mappant » un port du conteneur sur un port de la machine hôte.

## Fichier docker-compose :

```
2  services:
3      prometheus:
4          image: prom/prometheus:latest
5          container_name: 'prometheus'
6          volumes:
7              - ./prometheus.yml:/etc/prometheus/prometheus.yml
8          ports:
9              - '9090:9090' # Port pour accéder à Prometheus
10
```

**Remarque :** il faut créer un dossier où on va mettre le fichier **docker-compose.yml**, le fichier **nginx.conf** et le fichier **prometheus.yml** par exemple dans mon cas, c'est :



Nom	Modifié le	Type	Taille
 docker-compose.yml	03/12/2024 14:27	Fichier source Yaml	2 Ko
 nginx.conf	16/11/2024 20:09	Fichier CONF	1 Ko
 prometheus.yml	20/11/2024 14:30	Fichier source Yaml	1 Ko

Le fichier **prometheus.yml** est le **fichier de configuration principal** de **Prometheus**. Il définit comment Prometheus collecte les métriques, depuis quelles cibles, et d'autres paramètres comme le scraping, l'alerte et la gestion des job.

#### Fichier de configuration de prometheus :

```
prometheus.yml
1  global:
2    scrape_interval: '15s'
3    evaluation_interval: '15s'
4
5  scrape_configs:
6    - job_name: 'Prometheus'
7      static_configs:
8        - targets: ['prometheus:9090']
9
10   - job_name: 'Grafana'
11     static_configs:
12       - targets: ['grafana:3000']
13
14   - job_name: 'nginx'
15     static_configs:
16       - targets: ['nginx-exporter:9113']
```

Dans le fichier de configuration, il était essentiel de collecter les métriques des différents services déployés, notamment Grafana, Prometheus et NGINX. Cependant, pour certains services comme NGINX, l'utilisation d'un exporter est nécessaire afin d'exposer leurs métriques, car Prometheus ne peut pas interagir directement avec ces services. Cela justifie la présence d'un **NginxExporter** dans la liste des cibles configurées pour Prometheus.

# Le service Grafana

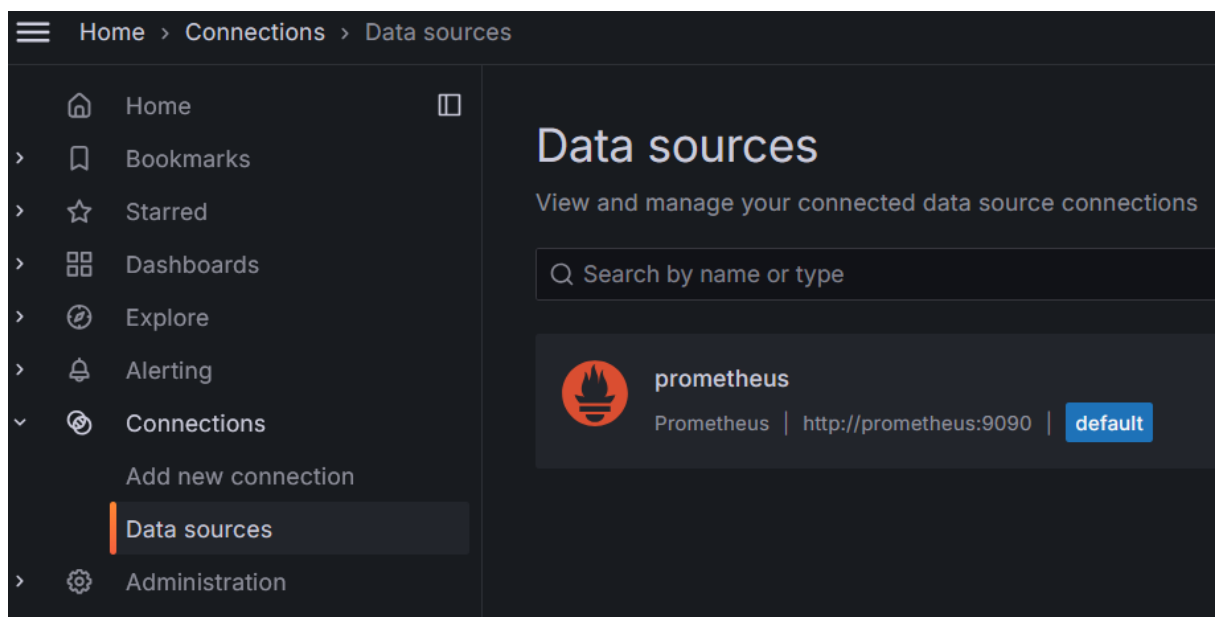
Grafana est une plateforme de visualisation de données open source développée par Grafana Labs, qui permet aux utilisateurs de consulter leurs données via des graphiques unifiés dans un ou plusieurs tableaux de bord afin de mieux les interpréter et les comprendre.

Pour le déploiement de Grafana, on utilise aussi le fichier docker-compose.yml et dans ce fichier, on récupère l'image officielle de grafana depuis la bibliothèque de Docker.

## Fichier docker-compose de grafana :

```
grafana:
  image: grafana/grafana:latest
  container_name: 'grafana'
  depends_on:
    - prometheus
  environment:
    - GF_SECURITY_USER_PASSWORD=admin
    - GF_SECURITY_ADMIN_PASSWORD=admin # Mot de passe admin
  ports:
    - '3000:3000' # Port pour accéder à Grafana
```

On peut ajouter le fichier de configuration pour le provisioning qui permet de renseigner une datasource par défaut au lancement de Grafana. Mais dans mon cas, je l'ai fait directement sur la plateforme de grafana pour renseigner notre datasource qui est Prometheus.



# Le service web NGINX

---

Nginx est un serveur web robuste et capable de gérer un grand nombre de connexions simultanées avec une utilisation maximale des ressources. C'est un logiciel libre de serveur web ainsi qu'un proxy inverse.

## Fichier docker-compose :

```
nginx:
  image: nginx:latest
  container_name: nginx
  volumes:
    - ./nginx.conf:/etc/nginx/nginx.conf:ro
  ports:
    - "8080:80"
```

## Fichier de configuration de NGINX :

```
N nginx.conf
1  events {}
2
3  http {
4      server {
5          listen 80;
6
7          location / {
8              root /usr/share/nginx/html; #définir la racine du serveur
9              index index.html index.htm; #fichiers d'index par défaut
10         }
11
12         location /status {
13             stub_status on;
14             #allow 127.0.0.1; # Limite l'accès à localhost
15             #deny all;
16             allow all;
17         }
18     }
19 }
```

# Le service Exporter : Nginx Exporter

---

Le service NginxExporter est un élément important pour la récupération des métriques d'un serveur web NGINX. En effet, cet exporter va jouer le rôle d'intermédiaire entre le serveur web et le service Prometheus, en récupérant les métriques de NGINX, puis en les transmettant à Prometheus. Ce service ne demande pas de configuration spécifique.

## Fichier docker-compose :

```
nginx-exporter:
  image: nginx/nginx-prometheus-exporter:latest
  container_name: 'nginx-exporter'
  command: -nginx.scrape-uri=http://nginx/status
  environment:
    - NGINX_STATUS=http://nginx:80/status
  depends_on:
    - nginx
  ports:
    - '9113:9113' # Port pour exposer les métriques
```

Une fois que la configuration terminée, on exécute la commande docker-compose up -d pour démarrer les différents services (conteneurs) sur l'invite de commande.

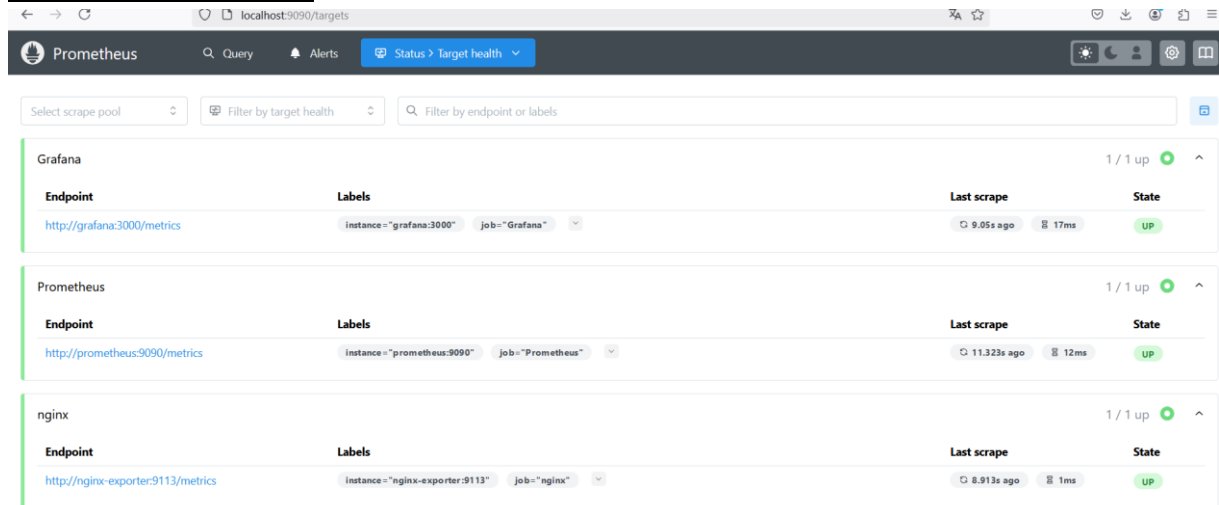
```
C:\Users\mamad\monitoring>docker-compose up -d
[+] Running 5/5
✓Container nginx           Started
✓Container node_exporter   Started
✓Container prometheus      Started
✓Container nginx-exporter  Started
✓Container grafana         Started

C:\Users\mamad\monitoring>|
```



# Le résultat

## Interface Prometheus :



## Interface Grafana avec le tableau de bord du serveur NGINX :

