

T-NSA-800  
LostOps

# Introduction

Ce document est une synthèse de notre projet T-NSA-800, le but de ce dernier est de monitorer une application web et ses différents services à l'aide de Prometheus et Grafana. Prometheus est un système de surveillance et d'alerte open-source. Grafana est une plateforme d'analyse et de visualisation de données. L'intégration de ces deux outils permet une surveillance efficace des systèmes et une visualisation claire des données collectées.

## Environnement de travail

- Trois VM Azure
  - Une VM Database
    - MySQL
  - Une VM App
    - PHP Laravel
    - Traefik
  - Une VM Docker
    - Container 1 : Grafana
    - Container 2 : Prometheus
    - Container 3 : Alert Manager
    - Container 4 : Kuma

## Installation et Configuration

### Étape 1 : Configuration de l'Environnement

Les VM Azure ont été fournies dans un Lab, nous allons omettre la partie création des VM.

### Étape 2 : Création des Containers Docker

- VM Database :
  - Installer MySQL et créer un user accessible depuis l'extérieur
- VM App : Deux containers Docker
  - L'application Laravel
  - Traefik qui fera office de Load Balancer

- VM Docker: Quatre containers Docker
  - Grafana
  - Prometheus
  - Alert Manager
  - Kuma

## Étape 3 : Configuration de Prometheus

La configuration de Prometheus se fait via un `prometheus.yml`, ce dernier regroupe toutes les sources sur lesquelles Prometheus va fetch de l'information.

`prometheus.yml`

```
global:
  scrape_interval: 15s
  evaluation_interval: 10s

scrape_configs:
  - job_name: 'web-app'
    static_configs:
      - targets: ['app-web912617.westeurope.cloudapp.azure.com:8082']

  - job_name: 'database'
    static_configs:
      - targets: ['app-database347477.westeurope.cloudapp.azure.com:9104']

  - job_name: 'node_exporter'
    static_configs:
      - targets: ['app-web912617.westeurope.cloudapp.azure.com:9100']

alerting:
  alertmanagers:
    - static_configs:
        - targets: ['app-docker355640.westeurope.cloudapp.azure.com:9093']

rule_files:
  - "rules.yml"
```

La configuration ci-dessus va permettre à Prometheus de scraper des metrics sur différents jobs, nous allons détailler les différentes sources de metrics.

## web-app

Il s'agit des metrics que Traefik nous fournit, cette source est accessible via l'endpoint /metrics qui est configurée dans le docker-compose.yaml de la VM App.

docker-compose.yaml

```
version: '3'
services:
  traefik:
    image: "traefik:v2.4"
    command:
      - "--providers.docker=true"
      - "--providers.docker.exposedbydefault=false"
      - "--entrypoints.web.address=:80"
      - "--entrypoints.metrics.address=:8082"
      - "--metrics.prometheus=true"
      - "--metrics.prometheus.entryPoint=metrics"
```

Les metrics accessibles sont l'uptime, le nombre, le temps, le return code des requêtes.

## database

Il s'agit des metrics de la BDD MySQL, nous avons suivi ce [tutoriel](#), cela permet d'exposer les metrics de la base de donnée tel que le nombre de requêtes, la quantité de données transférés, l'uptime.

## node\_exporter

Il s'agit des metrics de la VM Docker, nous avons suivi ce [tutoriel](#), cela permet d'exposer les metrics de la VM tel que l'usage CPU, la RAM disponible, l'espace restant sur le disque dur, l'uptime.

## Étape 4 : Configuration de Grafana

La configuration de Grafana a été faite via la GUI. Dans un premier temps il faut créer une Data Source, il s'agit d'un endpoint sur lequel Grafana va pouvoir récupérer de la donnée, cet endpoint est créé par Prometheus. Une fois la Data Source créée, Grafana va récupérer de manière périodique les informations de nos différents services via Prometheus. Il faut maintenant créer différents Dashboard. Nous avons créé un dashboard différent pour chaque source de metrics.

Une fois les dashboard créés, nous avons fait un export de notre Grafana qui permet de reconstruire un container Denovo sans devoir repasser par la GUI pour créer nos dashboard.

## Étape 5 : Configuration de Kuma

Uptime Kuma se configure directement via son interface web. On peut ajouter plusieurs types de monitoring (ping, http(s), tcp, base de données...) avec diverses options personnalisables, la fréquence de vérification, au bout de combien d'erreurs on estime que la cible est tombée et qu'il faut lancer une alerte.

Les monitorings qui ont été ajoutés à notre Uptime Kuma sont les suivants:

- L'accès à MariaDB avec le compte exporter
- Uptime Grafana
- Uptime Prometheus
- Le bon fonctionnement de Traefik
- L'accès au site web distribué par Traefik, ciblant directement le port de l'application
- Uptime VM Web et Database

Ces monitorings sont disponibles sur le dashboard de base de l'application.

Il existe également des Status Page, dans lesquels plusieurs monitorings sont regroupés. On a une première Status Page qui regroupe les containers de la même VM qui héberge Kuma.

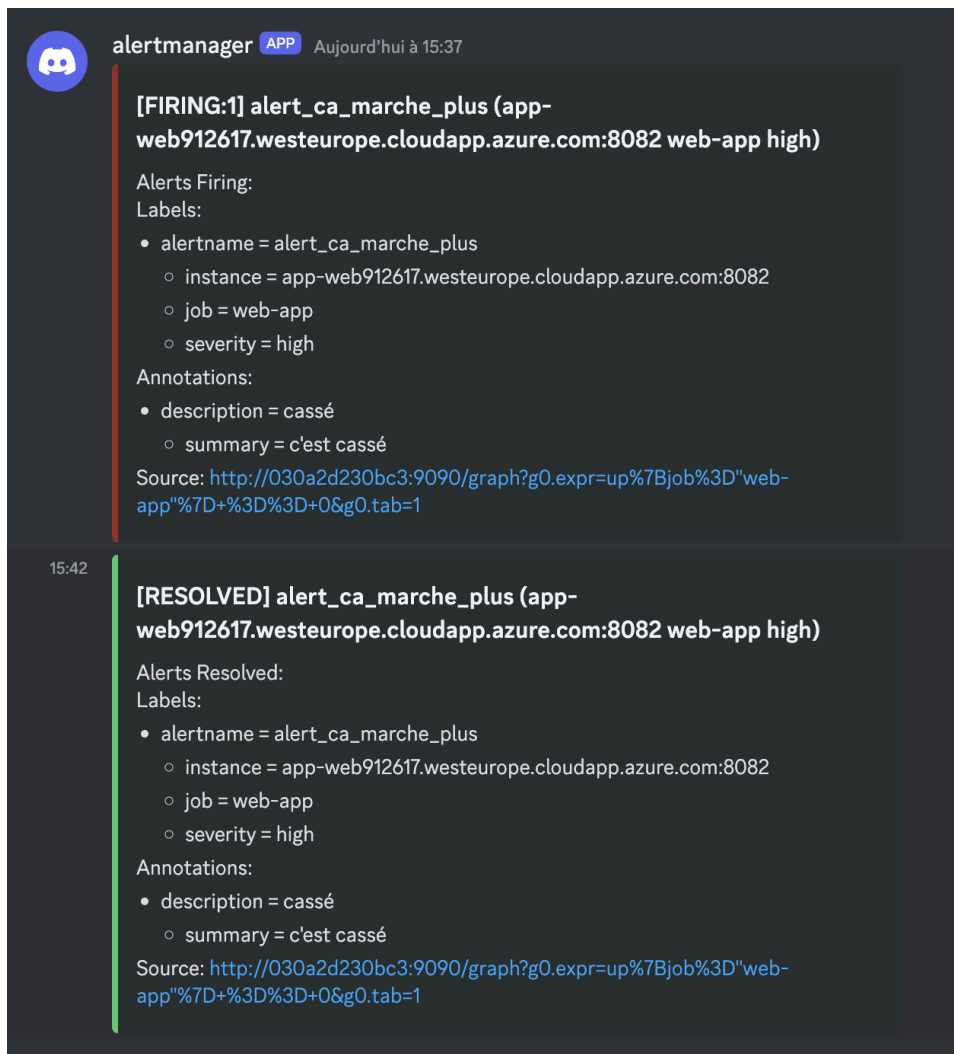
Une deuxième Status Page qui regroupe l'uptime général des VM web et database

## Étape 6 : Configuration d'Alert Manager

La configuration d'alert manager se fait via un alertmanager.yaml, ou un webhook discord est fourni pour l'envoi d'un message lors de la réception de l'alerte prometheus par alert manager, ainsi que la résolution de l'alerte.

```
route:
  group_by: ['alertname']
  group_wait: 10s
  group_interval: 5m
  repeat_interval: 3h
  receiver: 'discord'

receivers:
- name: discord
  discord_configs:
  - webhook_url: <WEBHOOK URL>
```



The screenshot shows a Discord chat window with a dark theme. At the top, the channel name is 'alertmanager' with a blue 'APP' badge and the text 'Aujourd'hui à 15:37'. The first message, timestamped '15:42', is a firing alert. It has a red vertical bar on the left and contains the following text: '[FIRING:1] alert\_ca\_marche\_plus (app-web912617.westeurope.cloudapp.azure.com:8082 web-app high)', 'Alerts Firing:', 'Labels:' followed by a bulleted list of labels (alertname, instance, job, severity), 'Annotations:' followed by a bulleted list of annotations (description, summary), and a 'Source:' link. The second message, also timestamped '15:42', is a resolved alert. It has a green vertical bar on the left and contains the following text: '[RESOLVED] alert\_ca\_marche\_plus (app-web912617.westeurope.cloudapp.azure.com:8082 web-app high)', 'Alerts Resolved:', 'Labels:' followed by a bulleted list of labels, 'Annotations:' followed by a bulleted list of annotations, and a 'Source:' link.

alertmanager **APP** Aujourd'hui à 15:37

**[FIRING:1] alert\_ca\_marche\_plus (app-web912617.westeurope.cloudapp.azure.com:8082 web-app high)**

Alerts Firing:

Labels:

- alertname = alert\_ca\_marche\_plus
  - instance = app-web912617.westeurope.cloudapp.azure.com:8082
  - job = web-app
  - severity = high

Annotations:

- description = cassé
  - summary = c'est cassé

Source: <http://030a2d230bc3:9090/graph?g0.expr=up%7Bjob%3D%3Dweb-app%7D+%3D%3D0&g0.tab=1>

15:42

**[RESOLVED] alert\_ca\_marche\_plus (app-web912617.westeurope.cloudapp.azure.com:8082 web-app high)**

Alerts Resolved:

Labels:

- alertname = alert\_ca\_marche\_plus
  - instance = app-web912617.westeurope.cloudapp.azure.com:8082
  - job = web-app
  - severity = high

Annotations:

- description = cassé
  - summary = c'est cassé

Source: <http://030a2d230bc3:9090/graph?g0.expr=up%7Bjob%3D%3Dweb-app%7D+%3D%3D0&g0.tab=1>

## Conclusion

Cet exercice avec Prometheus et Grafana offre une solution robuste de surveillance et de visualisation des données. Les documentations respectives des deux solutions sont très bien détaillées, ce qui facilite grandement la mise en place et la configuration de l'intégration. En suivant les étapes d'installation et les bonnes pratiques recommandées, il est possible de créer un système de surveillance et d'alerte robuste pour monitorer nos applications.

En conclusion, l'intégration de Prometheus et Grafana offre une solution complète et efficace pour la surveillance des systèmes et des applications. En suivant les meilleures pratiques et en exploitant pleinement les fonctionnalités offertes par ces outils, une équipe IT peut garantir une surveillance proactive, une détection rapide des incidents et une résolution efficace des problèmes, ce qui contribue à maintenir la santé et la stabilité de l'infrastructure informatique dans son ensemble.