

# Rapport de Projet : Mise en œuvre d'une infrastructure cloud de supervision centralisée sous AWS : Déploiement de Zabbix conteneurisé pour le monitoring d'un parc hybride (Linux & Windows)

Filière : Génie Informatique

**Réalisé par :**

Elimane Ndao

**Encadré par :**

Prof.Azeddine Khiat

**Année universitaire : 2024/2025**

## INTRODUCTION

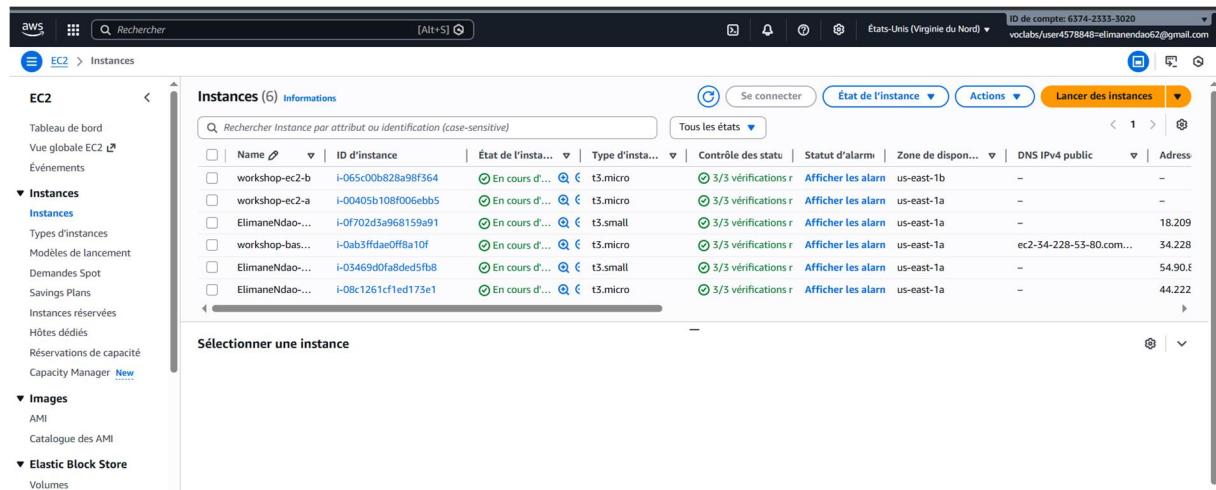
### 1. Contexte du projet

Avec l'essor du cloud computing, les entreprises migrent de plus en plus leurs infrastructures vers des plateformes cloud comme Amazon Web Services (AWS) afin de bénéficier de la flexibilité, de la scalabilité et de la haute disponibilité.

Cependant, cette migration implique un besoin crucial : le monitoring des ressources pour garantir la performance, la disponibilité et la sécurité des systèmes.

Dans ce contexte, le présent projet consiste à mettre en place un environnement cloud AWS comprenant plusieurs instances EC2, et à déployer une solution de supervision centralisée à l'aide de Zabbix, un outil open-source de monitoring largement utilisé en entreprise.

Le projet s'inscrit dans un cadre pédagogique visant à reproduire une architecture proche d'un environnement professionnel, incluant : un serveur de supervision Zabbix, un client Linux (Ubuntu), un client Windows, le tout interconnecté au sein d'un réseau sécurisé via des Security Groups AWS.



The screenshot shows the AWS Management Console with the EC2 service selected. The main pane displays a table of 6 EC2 instances. The columns include Name, ID d'instance, État de l'instance, Type d'insta..., Contrôle des statu..., Statut d'alarme, Zone de disponibilité, DNS IPv4 public, and Adress. The instances listed are workshop-ec2-b, workshop-ec2-a, ElimaneNda..., workshop-bas..., ElimaneNda..., and ElimaneNda... Each instance is currently running (En cours d'...).

Name	ID d'instance	État de l'insta...	Type d'insta...	Contrôle des statu...	Statut d'alarme	Zone de disponibilité	DNS IPv4 public	Adress
workshop-ec2-b	i-065c00b828a98f364	En cours d'...	t3.micro	3/3 vérifications r	Afficher les alarm	us-east-1b	–	–
workshop-ec2-a	i-00405b108f006eb5	En cours d'...	t3.micro	3/3 vérifications r	Afficher les alarm	us-east-1a	–	–
ElimaneNda...	i-0f702d3a968159a91	En cours d'...	t3.small	3/3 vérifications r	Afficher les alarm	us-east-1a	–	18.209
workshop-bas...	i-0ab5ffda0ff8a10f	En cours d'...	t3.micro	3/3 vérifications r	Afficher les alarm	us-east-1a	ec2-34-228-53-80.com...	34.228
ElimaneNda...	i-03469d0fa8ded5fb8	En cours d'...	t3.small	3/3 vérifications r	Afficher les alarm	us-east-1a	–	54.90.8
ElimaneNda...	i-08c1261cfed173e1	En cours d'...	t3.micro	3/3 vérifications r	Afficher les alarm	us-east-1a	–	44.222

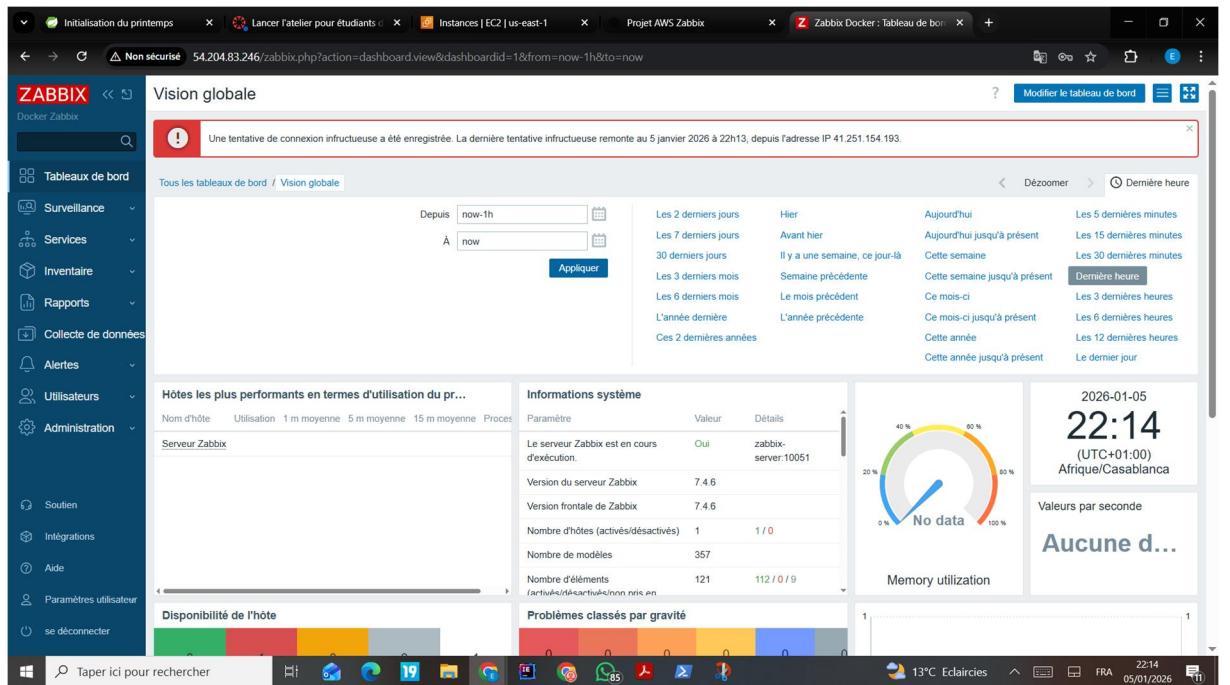
### 2. Objectifs du projet

L'objectif principal de ce projet est de concevoir et déployer une infrastructure de supervision complète sur AWS.

Les objectifs spécifiques sont les suivants :

- Déployer un serveur Zabbix sur une instance EC2 à l'aide de Docker et Docker Compose.
- Installer et configurer les agents Zabbix sur :
  - 1- une instance Linux (Ubuntu),
  - 2- une instance Windows.
- Assurer la communication réseau entre le serveur Zabbix et les agents via les ports nécessaires (10050, 10051).
- Ajouter les hôtes dans l'interface web Zabbix.
- Superviser les ressources système :

- 1- CPU
- 2- Mémoire
- 3- Disque
- 4- Disponibilité réseau
- Mettre en place des alertes en cas d'anomalies.
- Visualiser les performances à l'aide de graphiques dynamiques.



### 3. Outils utilisés

Pour la réalisation de ce projet, plusieurs outils et technologies ont été utilisés :

- Amazon Web Services (AWS) :
  - 1- EC2 pour l'hébergement des machines virtuelles
  - 2- Security Groups pour la gestion des règles de sécurité réseau
- Docker & Docker Compose :
  - 1- Déploiement rapide et reproductible du serveur Zabbix
- Zabbix :
  - 1- Zabbix Server
  - 2- Zabbix Web Interface
  - 3- Zabbix Agent (Linux & Windows)
- Systèmes d'exploitation :
  - 1- Ubuntu Server pour le serveur Zabbix et le client Linux
  - 2- Windows Server / Windows 10 pour le client Windows
- Outils complémentaires
  - 1- SSH
  - 2- PowerShell
  - 3- Navigateur web

## 4- draw.io pour les schémas réseau

### 4. Architecture Réseau

#### 4.1 Schéma VPC

Le projet consiste à mettre en place un environnement cloud AWS pour déployer un serveur Zabbix et deux clients (Linux et Windows). Le schéma ci-dessous illustre la topologie réseau utilisée :

- VPC principal avec un subnet public pour toutes les instances.
- Instances EC2 : Zabbix Server, Linux Client, Windows Client.
- Chaque instance est associée à un Security Group permettant l'accès SSH (22), HTTP (80), et les ports Zabbix (10050, 10051).
- Les adresses IP privées sont utilisées pour la communication interne entre le serveur Zabbix et les agents.

The screenshot shows the AWS VPC console interface. At the top, it displays the account ID (6374-2333-3020) and region (Etats-Unis (Virginie du Nord)). The main area is titled "Vos VPC" and shows three VPCs:

Name	ID de VPC	État	ID de contrôle d'accès	Mode de contrôle d'accès	Bloquer l'accès	CIDR IPv4
Zabbix-VPC	vpc-025c0823f1f23326c	Available	-	-	Désactivé	10.0.0.0/16
workshop-vpc	vpc-0e1c472270a0867b4	Available	-	-	Désactivé	10.0.0.0/16
-	vpc-0cf6fab6e9f73178a	Available	-	-	Désactivé	172.31.0.0/16

Below this, there is a section for "Groupes de sécurité (1/7) Informations" which lists seven security groups:

Name	ID du groupe de sécurité	Nom du groupe de sécurité	ID de VPC	Description
-	sg-0ab9634797c951d97	default	vpc-025c0823f1f23326c	default VPC security group
-	sg-0efe26172975fa3da	default	vpc-0cf6fab6e9f73178a	default VPC security group
<input checked="" type="checkbox"/>	sg-036e02ded7526e44a	Zabbix-Server-SG	vpc-025c0823f1f23326c	zabbix_server
-	sg-0ed131c50dbd4189c	Zabbix-Clients-SG	vpc-025c0823f1f23326c	client_server
-	sg-dadedd5fcfa813d50	workshop-sg-bastion	vpc-0e1c472270a0867b4	Administrateur SSH
-	sg-0a1d04a92e51a133f	workshop-sg-ec2	vpc-0e1c472270a0867b4	Groupe pour les instances applicatives.
-	sg-07a05c6820281a8fd	default	vpc-0e1c472270a0867b4	default VPC security group

Finally, there is a detailed view for the "Zabbix-Server-SG" security group:

Name	ID de règle de groupe	Type	Protocole	Plage de ports	Source
sgr-0b59071a5fa5b3e21	IPv4	HTTPS	TCP	443	0.0.0.0/0
sgr-0cc177e3c61d2d1ad	-	TCP personnalisé	TCP	10050	sg-036e02ded7526e44a
sgr-022c578df037bec57	IPv4	SSH	TCP	22	0.0.0.0/0
sgr-0fe94906e290ef11c	-	MySQL/Aurora	TCP	3306	sg-0ed131c50dbd4189c
sgr-0a56b3b6908c492c0	IPv4	HTTP	TCP	80	0.0.0.0/0

## 4.2 Security Groups

Les Security Groups définissent les règles de communication autorisées pour chaque instance.

### Security Group Zabbix-Server

- **Règles entrantes :**
  - SSH (22) depuis l'administrateur
  - HTTP (80) depuis le réseau interne et navigateur
  - Zabbix Server (10051) pour recevoir les données des agents
- **Règles sortantes :** Tout trafic autorisé

The screenshot shows the AWS CloudFormation console with the security group configuration for the Zabbix-Server-SG. The 'Règles sortantes' tab is selected, showing one rule: sgr-01e33e42b928823d5, IPv4, All traffic, All ports, All destination. There are tabs for Détails, Règles entrantes, Partage, Associations VPC, and Balises.

## 5. Instances EC2

The screenshot shows the AWS EC2 Instances page with six instances listed. The columns include Name, ID d'instance, État de l'instance, Type d'instance, Contrôle des statut, Statut d'alarme, Zone de disponibilité, DNS IPv4 public, and Adress. The instances are: workshop-ec2-b, workshop-ec2-a, ElimaneNda..., workshop-bas..., ElimaneNda..., and ElimaneNda... . The last two instances have incomplete names.

Name	ID d'instance	État de l'instance	Type d'instance	Contrôle des statut	Statut d'alarme	Zone de disponibilité	DNS IPv4 public	Adress
workshop-ec2-b	i-065c00b828a98f364	En cours d...	t3.micro	3/3 vérifications r	Afficher les alarm	us-east-1b	-	-
workshop-ec2-a	i-00405b108f006ebb5	En cours d...	t3.micro	3/3 vérifications r	Afficher les alarm	us-east-1a	-	-
ElimaneNda...	i-0f702d3a968159a91	En cours d...	t3.small	3/3 vérifications r	Afficher les alarm	us-east-1a	-	18.209
workshop-bas...	i-0ab5ffdaef0ff8a10f	En cours d...	t3.micro	3/3 vérifications r	Afficher les alarm	us-east-1a	ec2-34-228-53-80.com...	34.228
ElimaneNda...	i-03469df0fa8ded5fb8	En cours d...	t3.small	3/3 vérifications r	Afficher les alarm	us-east-1a	-	54.90.E
ElimaneNda...	i-08c1261cf1ed173e1	En cours d...	t3.micro	3/3 vérifications r	Afficher les alarm	us-east-1a	-	44.222

## 6. Déploiement Zabbix

Dans cette section, nous présentons l'installation et la vérification de Zabbix Server et des conteneurs associés via Docker.

### 6.1 Vérification des conteneurs Docker

Pour vérifier que les conteneurs Zabbix sont bien actifs, nous avons utilisé la commande :

```
docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
		NAMES		
f3c9445ce62d	zabbix/zabbix-web-nginx-mysql:latest	"docker-entrypoint.sh"	25 hours ago	Up 48 minutes (healthy)
	8443/tcp, 0.0.0.0:80->8080/tcp, [::]:80->8080/tcp	zabbix-web		
3819b3bd8704	zabbix/zabbix-server-mysql:latest	"/usr/bin/docker-ent..."	25 hours ago	Up 48 minutes
	0.0.0.0:10051->10051/tcp, [::]:10051->10051/tcp	zabbix-server		
6bfe5d285a69	mysql:8.0	"docker-entrypoint.s..."	25 hours ago	Up 48 minutes
	3306/tcp, 33060/tcp	mysql-server		

### Vérification des conteneurs Docker

Cette vérification permet de s'assurer que Zabbix Server, l'interface Web et la base de données MySQL fonctionnent correctement.

### Fichier docker-compose.yml

Le déploiement a été réalisé à l'aide d'un fichier **docker-compose.yml**, qui définit les services, images Docker et ports nécessaires pour Zabbix.

Pour visualiser le fichier, nous avons utilisé :

```
nano docker-compose.yml
```

Le fichier contient les services suivants :

- **mysql** : base de données pour Zabbix
- **zabbix-server** : serveur Zabbix
- **zabbix-web** : interface web Zabbix accessible sur le port 80

Les ports exposés permettent :

- **10051** : communication avec les agents Zabbix
- **80** : accès à l'interface web

```

GNU nano 7.2                                     docker-compose.yml

version: '3.8'

services:
  mysql-server:
    image: mysql:8.0
    container_name: mysql-server
    environment:
      MYSQL_DATABASE: zabbix
      MYSQL_USER: zabbix
      MYSQL_PASSWORD: zabbix_pwd
      MYSQL_ROOT_PASSWORD: root_pwd
    volumes:
      - mysql-data:/var/lib/mysql
    restart: unless-stopped

  zabbix-server:
    image: zabbix/zabbix-server-mysql:latest
    container_name: zabbix-server
    environment:
      DB_SERVER_HOST: mysql-server
      MYSQL_DATABASE: zabbix
      MYSQL_USER: zabbix
      MYSQL_PASSWORD: zabbix_pwd
      MYSQL_ROOT_PASSWORD: root_pwd
    ports:
      - "10051:10051"
    depends_on:
      - mysql-server
    restart: unless-stopped

  zabbix-web:
    image: zabbix/zabbix-web-nginx-mysql:latest
    container_name: zabbix-web
    environment:
      DB_SERVER_HOST: mysql-server
      MYSQL_DATABASE: zabbix
      MYSQL_USER: zabbix
      MYSQL_PASSWORD: zabbix_pwd
      ZBX_SERVER_HOST: zabbix-server
      PHP_TZ: Africa/Casablanca
    ports:
      - "80:8080"
    depends_on:
      - mysql-server
      - zabbix-server
    restart: unless-stopped

volumes:
  mysql-data:

```

Fichier docker-compose.yml

## 7. Configuration Agents - Linux :

Dans cette section, nous détaillons la configuration des agents Zabbix sur les machines Linux et Windows afin de permettre au serveur Zabbix de collecter les données.

---

## 7.1 Agent Linux

### 7.1.1 Vérification du service Zabbix Agent

Après avoir installé l'agent Zabbix sur l'instance Linux, nous avons vérifié que le service était actif avec la commande :

```
sudo systemctl status zabbix-agent
```

```
ubuntu@ip-10-0-1-126:~$ ubuntu@ip-10-0-1-126:~$ sudo systemctl status zabbix-agent
● zabbix-agent.service - Zabbix Agent
  Loaded: loaded (/usr/lib/systemd/system/zabbix-agent.service; enabled; preset: enabled)
  Active: active (running) since Tue 2026-01-06 21:18:47 UTC; 1h 5min ago
    Process: 559 ExecStart=/usr/sbin/zabbix_agentd -c $CONFFILE (code=exited, status=0/SUCCESS)
   Main PID: 589 (zabbix_agentd)
     Tasks: 7 (limit: 2213)
    Memory: 6.5M (peak: 7.0M)
      CPU: 1.208s
     CGroup: /system.slice/zabbix-agent.service
             └─589 /usr/sbin/zabbix_agentd -c /etc/zabbix/zabbix_agentd.conf
                  ├─600 "/usr/sbin/zabbix_agentd: collector [idle 1 sec]"
                  ├─604 "/usr/sbin/zabbix_agentd: listener #1 [waiting for connection]"
                  ├─608 "/usr/sbin/zabbix_agentd: listener #2 [waiting for connection]"
                  ├─609 "/usr/sbin/zabbix_agentd: listener #3 [waiting for connection]"
                  ├─612 "/usr/sbin/zabbix_agentd: active checks #1 [idle 1 sec]"
                  └─616 "/usr/sbin/zabbix_agentd: active checks #2 [idle 1 sec]"

Jan 06 21:18:46 ip-10-0-1-126 systemd[1]: Starting zabbix-agent.service - Zabbix Agent...
Jan 06 21:18:47 ip-10-0-1-126 systemd[1]: Started zabbix-agent.service - Zabbix Agent.
ubuntu@ip-10-0-1-126:~$
```

Agent Linux

### 7.1.2 Configuration du fichier zabbix\_agentd.conf

Le fichier de configuration de l'agent Linux se trouve à :

```
/etc/zabbix/zabbix_agentd.conf
```

Nous avons vérifié et modifié les paramètres essentiels pour assurer la communication avec le serveur :

```
sudo nano /etc/zabbix/zabbix_agentd.conf
```

```

GNU nano 7.2                               /etc/zabbix/zabbix_agentd.conf
# This is a configuration file for Zabbix agent daemon (Unix)
# For more info: http://www.zabbix.com

#####
# GENERAL PARAMETERS #####
#####

# PID file
PidFile=/run/zabbix/zabbix_agentd.pid

# Logging
LogType=file
LogFile=/var/log/zabbix/zabbix_agentd.log
LogFileSize=0
DebugLevel=3

#####
# SERVER PARAMETERS #####
#####

# IP du serveur Zabbix (pour les checks passifs)
Server=10.0.1.126,172.18.0.2

# IP du serveur Zabbix pour les checks actifs
ServerActive=10.0.1.126,172.18.0.2

# Nom de l'hôte tel qu'il est défini dans Zabbix Web
Hostname=Linux-Client

#####
# INCLUDE PARAMETERS #####
#####

# Inclure d'autres fichiers de configuration si besoin
Include=/etc/zabbix/zabbix_agentd.d/*.conf

```

Fichier de configuration Agent Linux

Les paramètres importants sont :

Paramètre	Valeur recommandée	Description
Server	10.0.1.126	IP du serveur Zabbix pour les vérifications passives
ServerActive	10.0.1.126	IP du serveur Zabbix pour les vérifications actives (une seule IP)
Hostname	Linux-Client	Nom exact de l'hôte tel qu'il apparaît dans Zabbix Web (sensible à la casse)
LogFile	/var/log/zabbix/zabbix_agentd.log	Fichier de logs de l'agent
LogFileSize	0	Taille maximale du fichier de log (0 = illimité)
DebugLevel	3	Niveau de détails des logs (debug)

#### □ Points importants à vérifier :

- **ServerActive** doit contenir **une seule IP**, pas de virgule.
- **Hostname** doit être exactement identique au nom de l'hôte dans Zabbix Web.
- Aucun espace supplémentaire après les adresses IP.

```
PS C:\Users\Administrator> Get-Service "Zabbix Agent"
```

Status	Name	DisplayName
Running	Zabbix Agent	Zabbix Agent

```
PS C:\Users\Administrator>
```

### Agent Windows

```
## # Mandatory: no
PS # Default:
# ServerActive=
ServerActive=10.0.1.126

### Option: Hostname
#      List of comma delimited unique, case sensitive hostnames.
#      Required for active checks and must match hostnames as configured on the server.
#      Value is acquired from HostnameItem if undefined.
#
# Mandatory: no
# Default:
# Hostname=

Hostname=Windows-Client
```

### Fichier de configuration Agent Windows

## 8. Monitoring

### Ajout des Hosts

Après avoir déployé le serveur Zabbix et configuré les agents sur Linux et Windows, nous avons ajouté ces hôtes dans l'interface Zabbix Web pour les surveiller.

#### Processus suivi :

1. Se connecter à l'interface Zabbix Web.
2. Aller dans **Configuration -> Hosts**.
3. Cliquer sur **Create host** pour ajouter chaque machine :
  - o **Nom de l'hôte** : exactement le nom de l'agent (Linux-Client ou Windows-Client)
  - o **IP de l'interface** : l'adresse privée de l'instance (ex. 10.0.1.126 pour Linux, 10.0.1.197 pour Windows)
  - o **Groupe** : Linux servers ou Windows servers
  - o **Agent interface** : activée sur le port 10050

## Visualisation des performances (Graphiques)

Une fois les hôtes ajoutés et les agents actifs, Zabbix commence à collecter les données et à générer des graphiques de performances pour chaque machine.

### Graphiques principaux observés pour Linux-Client :

- CPU Usage** : Utilisation du processeur en temps réel.
- Memory Usage** : Utilisation de la mémoire RAM.
- Load Average** : Charge moyenne du système.

## Ajout des hôtes

## Graphiques

## **Graphiques principaux observés pour Windows-Client :**

- **CPU Usage** : Utilisation du processeur.
- **Memory Usage** : Utilisation de la RAM.
- **Disk Usage** : Utilisation du disque dur.

## **9. Conclusion**

Au terme de ce projet, nous avons mis en place un environnement de supervision complet des instances EC2 sur AWS en utilisant **Zabbix**.

Les principales réalisations sont les suivantes :

- Déploiement d'un serveur Zabbix et d'une interface Web fonctionnelle avec **Docker**.
- Configuration des agents Zabbix sur **Linux** et **Windows** pour assurer la collecte des données.
- Ajout et surveillance des hôtes dans Zabbix Web avec suivi des **ressources système** (CPU, mémoire, disque, charge).
- Mise en place d'alertes pour détecter rapidement toute indisponibilité ou anomalie sur les hôtes.

## **Difficultés rencontrées :**

- Problèmes initiaux de communication entre les agents et le serveur Zabbix, liés à une mauvaise configuration des paramètres `Hostname`, `Server` et `ServerActive`.
- Gestion des IP autorisées dans le fichier de configuration de l'agent Linux et des agents Windows.
- Vérification et suivi des services Docker pour garantir que tous les conteneurs Zabbix étaient actifs.

## **Solutions apportées :**

- Correction des fichiers de configuration avec des `ServerActive` et `Hostname` exacts.
- Redémarrage des agents après modification et vérification des logs.
- Suivi précis avec `docker ps` et `systemctl status zabbix-agent` pour garantir le bon fonctionnement.

## **Compétences acquises :**

- Maîtrise de l'installation et de la configuration de Zabbix dans un environnement cloud.
- Gestion et dépannage des agents Zabbix sous Linux et Windows.
- Visualisation et analyse des métriques de performance système.
- Déploiement et gestion de conteneurs Docker pour des services de monitoring.

En conclusion, ce projet a permis de **mettre en pratique les compétences en administration système, cloud et supervision**, tout en fournissant un environnement complet et opérationnel pour la surveillance proactive des instances EC2.

## **10. Annexes**

Voici le lien du GitHub : <https://github.com/nelimane-png/aws-zabbix-monitoring>