HESTIM
ENGINEERING & BUSINESS SCHOOL

Rapport de Travail Pratique

Infrastructure de Monitoring Centralisé sur AWS

Titre : Mise en œuvre d'une infrastructure cloud de supervision centralisée sous AWS

Réalisé par :
El Abssi Sami

Encadré par :
Prof. Azeddine KHIAT

Filière : Ingénieur informatique et intelligence artificielle

Année universitaire : 2025/2026

Dépôt GitHub :
https://github.com/samielabssi/Infrastructure-de-Monitoring-Centralis-sur-AWS

Table des matières

1 Introduction

1.1 Présentation du projet

Ce travail pratique a pour objectif de déployer une infrastructure de monitoring centralisée sur Amazon Web Services (AWS) en utilisant Zabbix conteneurisé avec Docker. L'infrastructure vise à surveiller un parc hybride composé de machines Linux et Windows, démontrant ainsi la flexibilité et la puissance de Zabbix dans un environnement cloud.

1.2 Outils utilisés

— **AWS (Amazon Web Services)** : Plateforme cloud utilisée pour héberger toute l'infrastructure (instances EC2, VPC, groupes de sécurité).

— **Docker** : Solution de conteneurisation permettant de déployer facilement Zabbix et ses composants (serveur, base de données, interface web).

— **Zabbix** : Solution de monitoring open-source offrant une surveillance complète des ressources système et réseau.

1.3 Architecture globale

L'infrastructure déployée comprend :

— Un VPC avec un sous-réseau public

— Trois instances EC2 : serveur Zabbix, client Linux, client Windows

— Des groupes de sécurité pour contrôler les flux réseau

— Zabbix déployé via Docker Compose

— Des agents Zabbix installés sur les clients

2 Architecture Réseau

2.1 Création du VPC

La première étape consiste à créer un VPC (Virtual Private Cloud) isolé pour héberger notre infrastructure. Nous avons opté pour une configuration simple avec un sous-réseau public pour faciliter l'accès sans VPN.

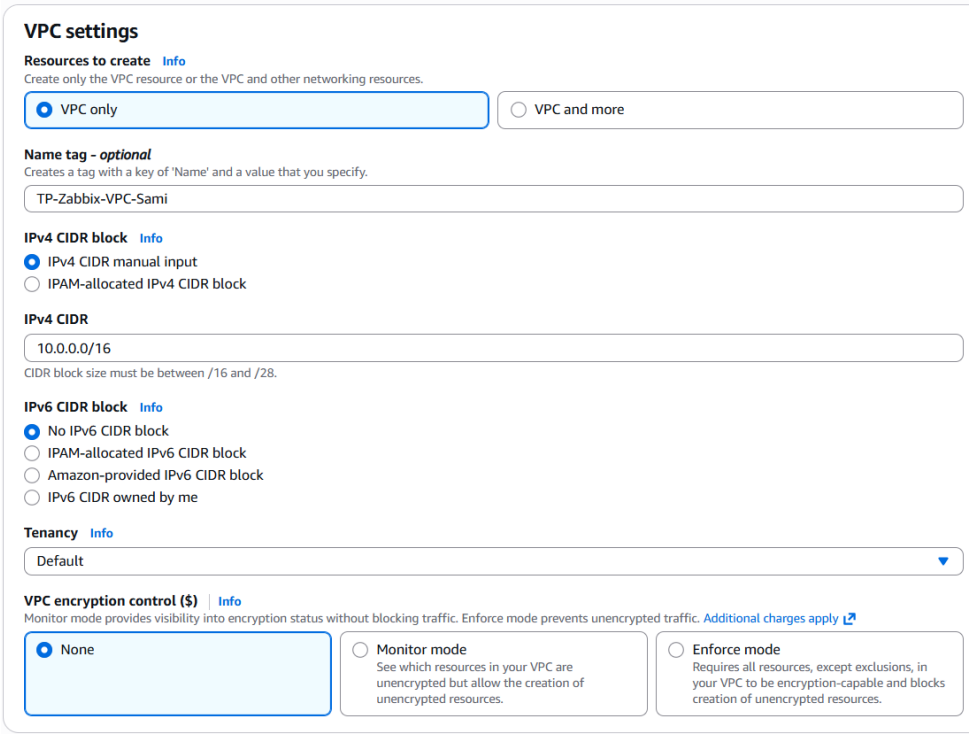

FIGURE 1 – Création du VPC 'TP-Zabbix-VPC' avec le bloc CIDR 10.0.0.0/16

2.2 Configuration du sous-réseau et de la passerelle Internet

Un sous-réseau public a été créé dans la zone de disponibilité us-east-1a. Une passerelle Internet a été attachée au VPC pour permettre la communication avec l'extérieur.

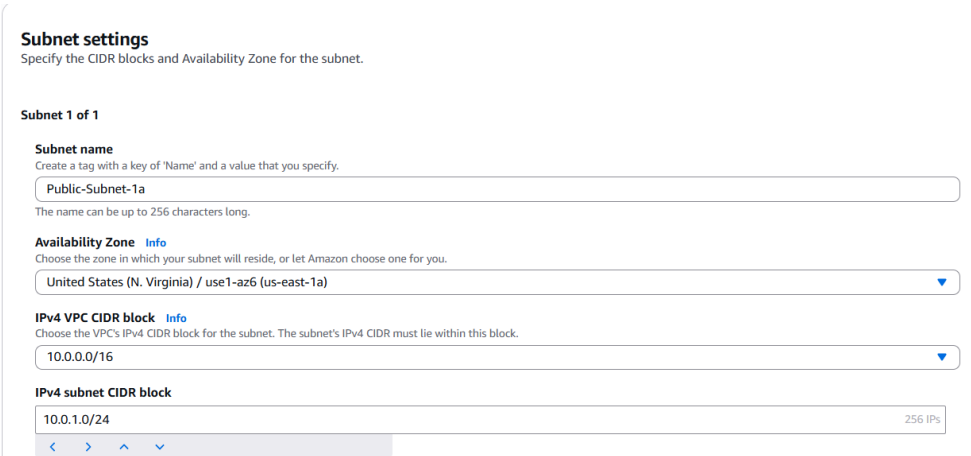

FIGURE 2 – Configuration du sous-réseau public avec le CIDR 10.0.1.0/24

2.3 Table de routage

La table de routage a été configurée pour diriger le trafic Internet (0.0.0.0/0) vers la passerelle Internet.

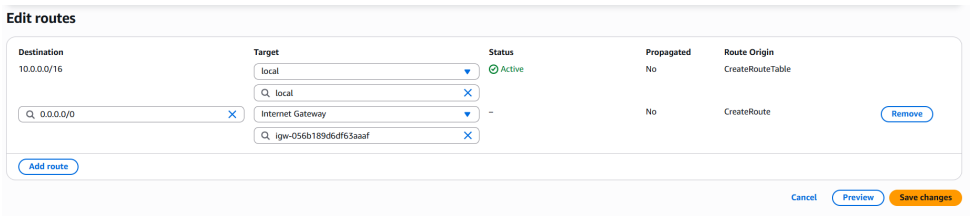

FIGURE 3 – Table de routage avec la route par défaut vers l'Internet Gateway

2.4 Groupes de sécurité

Deux groupes de sécurité ont été créés pour sécuriser les communications :

2.4.1 Groupe de sécurité du serveur Zabbix

Ce groupe contrôle l'accès au serveur Zabbix :

— Port 22 (SSH) : Accès administrateur

— Port 80/443 (HTTP/HTTPS) : Interface web Zabbix

— Ports 10050/10051 : Communications avec les agents Zabbix

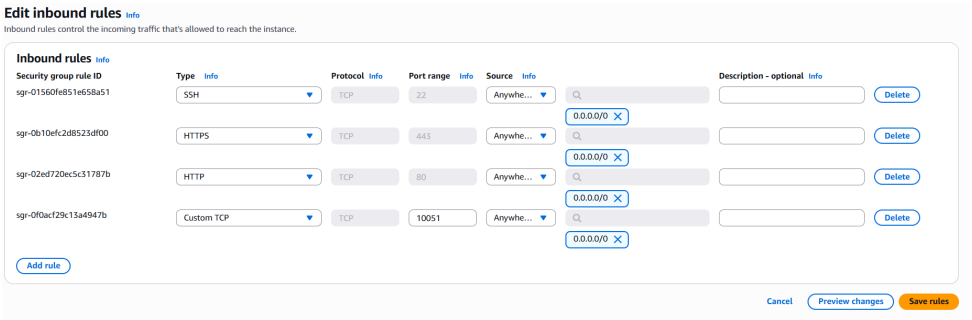

FIGURE 4 – Règles d'entrée du groupe de sécurité 'SG_Serveur_Zabbix'

2.4.2 Groupe de sécurité des clients

Ce groupe sécurise les instances clientes :

— Port 22 (SSH) : Accès au client Linux

— Port 3389 (RDP) : Accès au client Windows

— Ports 10050/10051 : Restreints à l'IP privée du serveur Zabbix

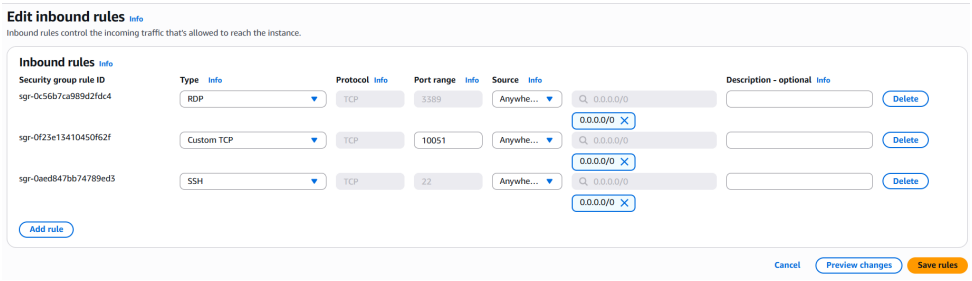

FIGURE 5 – Règles d'entrée du groupe de sécurité 'SG_Clients'

3 Déploiement des Instances EC2

3.1 Types d'instances choisies

Conformément aux limitations du Learner Lab, nous avons déployé trois instances :

Instance	Type	OS	Rôle
Serveur Zabbix	t3.large	Ubuntu 22.04	Serveur de monitoring
Client Linux	t3.medium	Ubuntu 22.04	Hôte monitoré Linux
Client Windows	t3.large	Windows Server 2022	Hôte monitoré Windows

TABLE 1 – Configuration des instances EC2 déployées

3.2 Instances en cours d'exécution

Toutes les instances ont été lancées dans la région us-east-1 (N. Virginia) pour assurer la stabilité du Lab.

FIGURE 6 – Récapitulatif des trois instances EC2 en cours d'exécution avec leurs IPs

3.3 Connexion au serveur Zabbix

La connexion SSH à l'instance serveur a été établie avec succès.

FIGURE 7 – Connexion SSH réussie à l'instance 'Serveur Zabbix'

4 Déploiement du Serveur Zabbix en Mode Conteneurisé

4.1 Installation de Docker et Docker Compose

Sur l'instance serveur, nous avons installé Docker et Docker Compose.

```
# Installation de Docker
sudo apt update && sudo apt upgrade -y
sudo apt install docker.io -y
sudo systemctl start docker
sudo systemctl enable docker

# Installation de Docker Compose
sudo apt install docker-compose -y
```

Listing 1 – Installation de Docker et Docker Compose

4.2 Configuration du fichier docker-compose.yml

Le fichier `docker-compose.yml` définit trois services : MySQL (base de données), Zabbix Server et l'interface web Zabbix.

```
version: '3.5'
services:
    mysql-server:
        image: mysql:8.0
        container_name: zabbix-mysql
        command: --character-set-server=utf8mb4 --collation-
   server=utf8mb4_unicode_ci
        volumes:
          - ./zabbix-mysql-data:/var/lib/mysql
        environment:
          MYSQL_DATABASE: 'zabbix'
          MYSQL_USER: 'zabbix'
          MYSQL_PASSWORD: 'zabbix_pwd'
          MYSQL_ROOT_PASSWORD: 'root_pwd'
        restart: always
        networks:
          - zbx_network

    zabbix-server:
        image: zabbix/zabbix-server-mysql:alpine-6.4-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:
```

```
28            - "10051:10051"
29          depends_on:
30            - mysql-server
31          restart: always
32          networks:
33            - zbx_network
34
35      zabbix-web:
36          image: zabbix/zabbix-web-nginx-mysql:alpine-6.4-latest
37          container_name: zabbix-web
38          environment:
39            ZBX_SERVER_HOST: 'zabbix-server'
40            DB_SERVER_HOST: 'mysql-server'
41            MYSQL_DATABASE: 'zabbix'
42            MYSQL_USER: 'zabbix'
43            MYSQL_PASSWORD: 'zabbix_pwd'
44            MYSQL_ROOT_PASSWORD: 'root_pwd'
45            PHP_TZ: 'Europe/Paris'
46          ports:
47            - "80:8080"
48          depends_on:
49            - mysql-server
50            - zabbix-server
51          restart: always
52          networks:
53            - zbx_network
54
55  networks:
56      zbx_network:
57          driver: bridge
```

Listing 2 – Fichier docker-compose.yml

4.3 Lancement des conteneurs

Les conteneurs ont été lancés avec Docker Compose.

```
ubuntu@ip-10-0-1-83:~/zabbix-docker$ sudo docker-compose up -d
Creating network "zabbix-docker_zbx_network" with driver "bridge"
Pulling mysql-server (mysql:8.0)...
8.0: Pulling from library/mysql
4f37333d1be6: Pull complete
bde62e757594: Pull complete
f508d7fab5b3: Pull complete
d442b2c1726e: Pull complete
a9a9deeee02a: Pull complete
23fbf4028535: Pull complete
2e2c1f6f8d57: Pull complete
ce98f3559366: Pull complete
bae900376130: Pull complete
e7a04c019bde: Pull complete
e05db5310ebc: Pull complete
Digest: sha256:99d774bf02a48a1bb1c599920d2571946d31e5940b854b02737d5e95c184358f
Status: Downloaded newer image for mysql:8.0
Pulling zabbix-server (zabbix/zabbix-server-mysql:alpine-6.4-latest)...
alpine-6.4-latest: Pulling from zabbix/zabbix-server-mysql
1f3e46996e29: Pull complete
1772a90c6a5c: Pull complete
f803498c2d79: Pull complete
152feda2c3af: Pull complete
fe17bf02638c: Pull complete
e266bac3ed30: Pull complete
4f4fb700ef54: Pull complete
2d84b546e482: Pull complete
Digest: sha256:cda6ddbda3fc0c5d18294fa2a5d6df13e2decae692e33458847e5bf9dadfd9cb
Status: Downloaded newer image for zabbix/zabbix-server-mysql:alpine-6.4-latest
Pulling zabbix-web (zabbix/zabbix-web-nginx-mysql:alpine-6.4-latest)...
alpine-6.4-latest: Pulling from zabbix/zabbix-web-nginx-mysql
1f3e46996e29: Already exists
f22e0f609cf2: Pull complete
447f8b7a49df: Pull complete
17c1c2cf73ef: Pull complete
4f4fb700ef54: Pull complete
47f2a23549dd: Pull complete
Digest: sha256:002c4d7da19d3de020c278b1b442eb8e413f319c1f00e2a7c9b9e8e9b6e30013
Status: Downloaded newer image for zabbix/zabbix-web-nginx-mysql:alpine-6.4-latest
Creating zabbix-mysql ... done
Creating zabbix-server ... done
Creating zabbix-web    ... done
ubuntu@ip-10-0-1-83:~/zabbix-docker$
```

FIGURE 8 – Exécution de la commande `sudo docker-compose up -d` sur le serveur

4.4 Vérification de l'état des conteneurs

Les trois conteneurs sont bien en état 'Up', indiquant un fonctionnement correct.

```
ubuntu@ip-10-0-1-83:~/zabbix-docker$ sudo docker-compose ps
    Name                  Command               State                          Ports
-------------------------------------------------------------------------------------------------------------
zabbix-mysql    docker-entrypoint.sh --cha ...   Up            3306/tcp, 33060/tcp
zabbix-server   /sbin/tini -- /usr/bin/doc ...   Up            0.0.0.0:10051->10051/tcp,:::10051->10051/tcp
zabbix-web      docker-entrypoint.sh             Up (healthy)  0.0.0.0:80->8080/tcp,:::80->8080/tcp, 8443/tcp
```

FIGURE 9 – Résultat de la commande `sudo docker-compose ps` montrant les conteneurs en état 'Up'

4.5 Accès à l'interface web Zabbix

L'interface web est accessible via l'adresse IP publique du serveur.

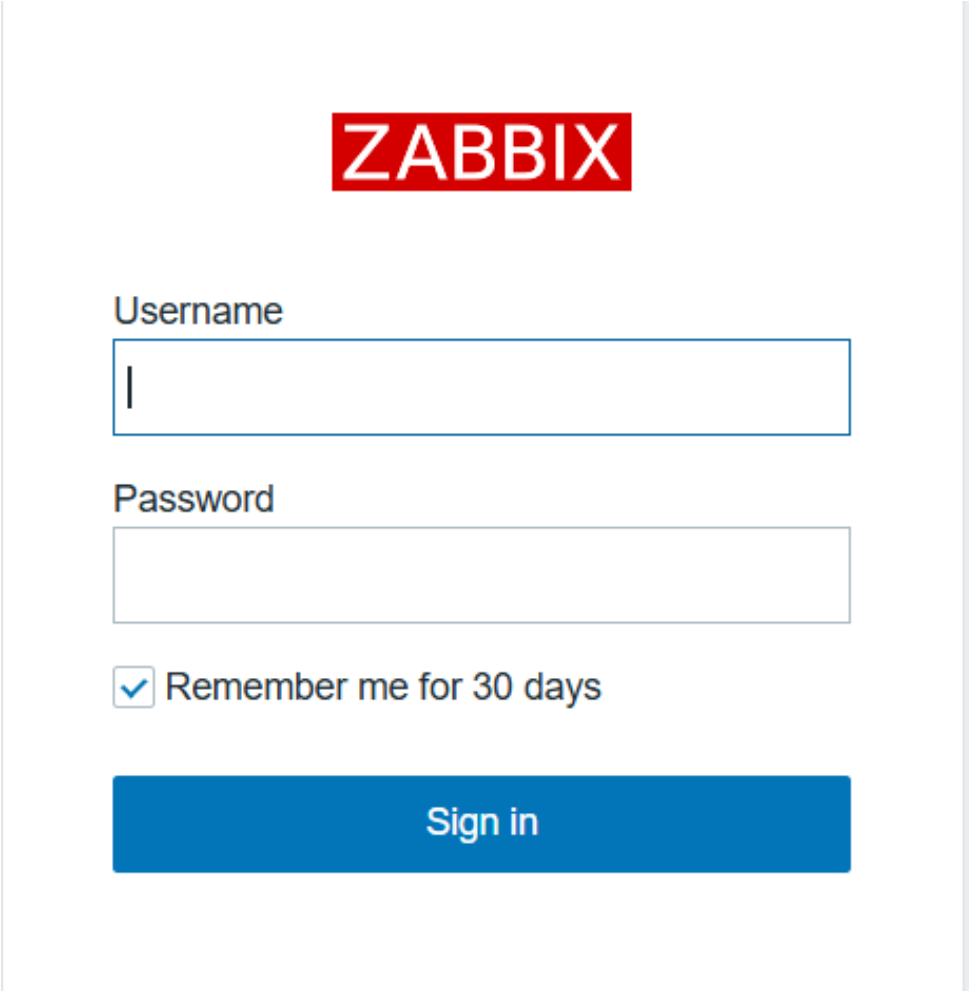

FIGURE 10 – Page de connexion à l'interface web de Zabbix

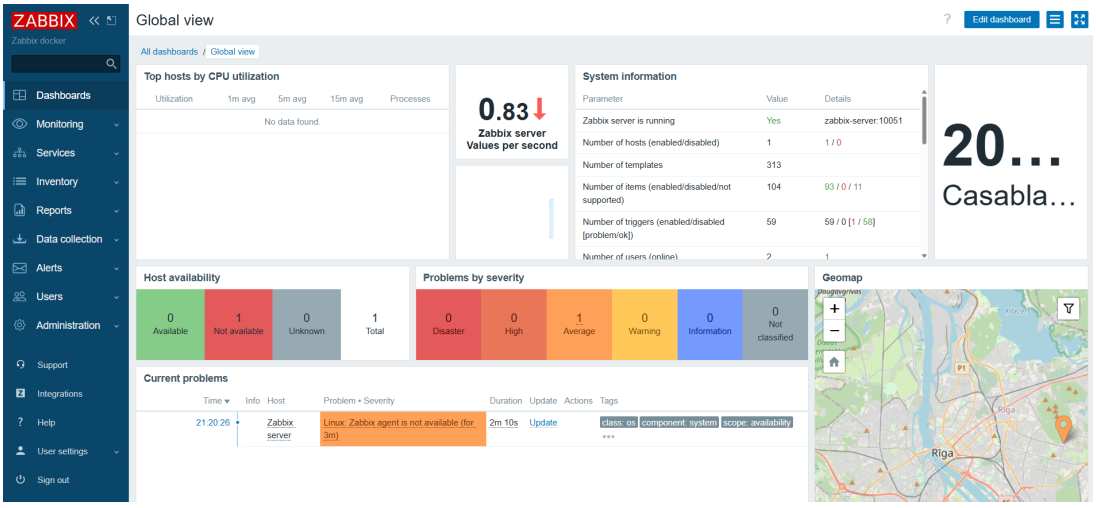

FIGURE 11 – Interface principale de Zabbix après connexion réussie

5 Configuration des Clients (Agents Zabbix)

5.1 Configuration du client Linux

5.1.1 Installation de l'agent

Sur l'instance client Linux, nous avons installé l'agent Zabbix.

```
wget https://repo.zabbix.com/zabbix/6.4/ubuntu/pool/main/z/zabbix-release/zabbix-release_6.4-1+ubuntu22.04_all.deb
sudo dpkg -i zabbix-release_6.4-1+ubuntu22.04_all.deb
sudo apt update
sudo apt install zabbix-agent -y
```

Listing 3 – Installation de l'agent sur Linux

5.1.2 Configuration de l'agent

Le fichier de configuration a été modifié pour pointer vers le serveur Zabbix.

```
ServerActive=10.0.1.83

### 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=Client-Linux-Ubuntu
```

FIGURE 12 – Configuration du fichier /etc/zabbix/zabbix_agentd.conf sur le client Linux

5.1.3 Démarrage du service

Le service a été démarré et configuré pour se lancer automatiquement.

```
ubuntu@ip-10-0-1-212:/etc/zabbix$ sudo systemctl restart zabbix-agent
sudo systemctl enable zabbix-agent
Synchronizing state of zabbix-agent.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable zabbix-agent
```

FIGURE 13 – Démarrage du service zabbix-agent avec vérification de son statut

5.2 Configuration du client Windows

5.2.1 Installation de l'agent

Sur l'instance Windows, l'agent a été installé via l'installateur MSI.

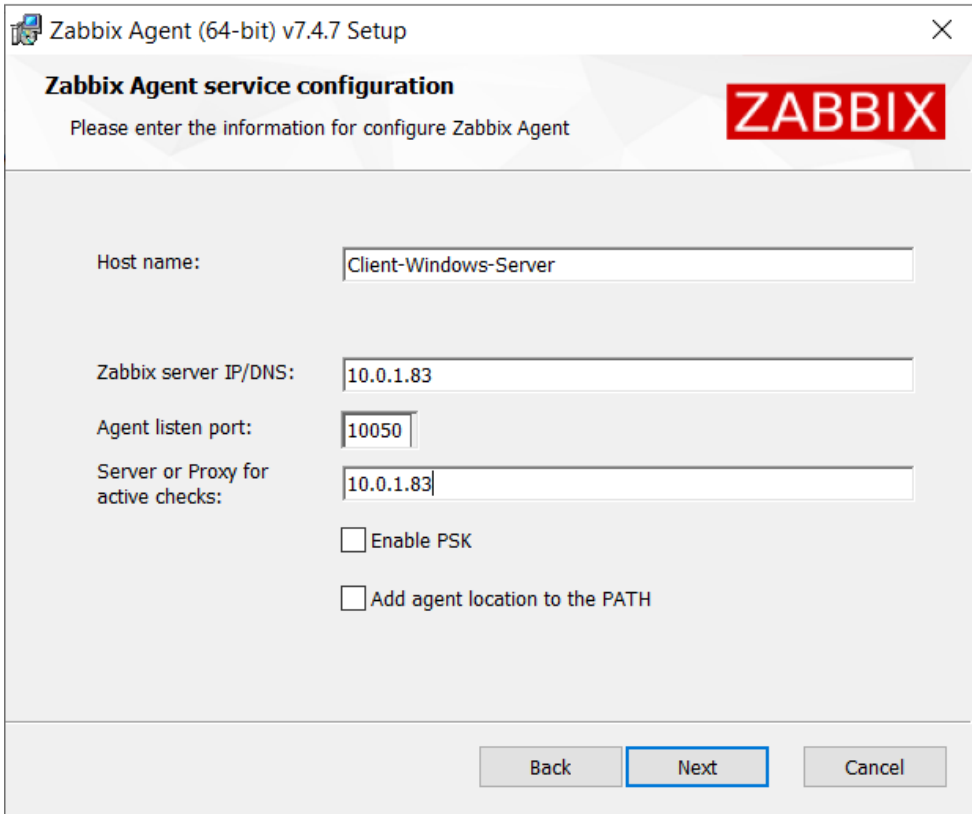

FIGURE 14 – Assistant d'installation de l'agent Zabbix pour Windows

5.2.2 Vérification du service

Le service Windows a été vérifié dans le gestionnaire de services.

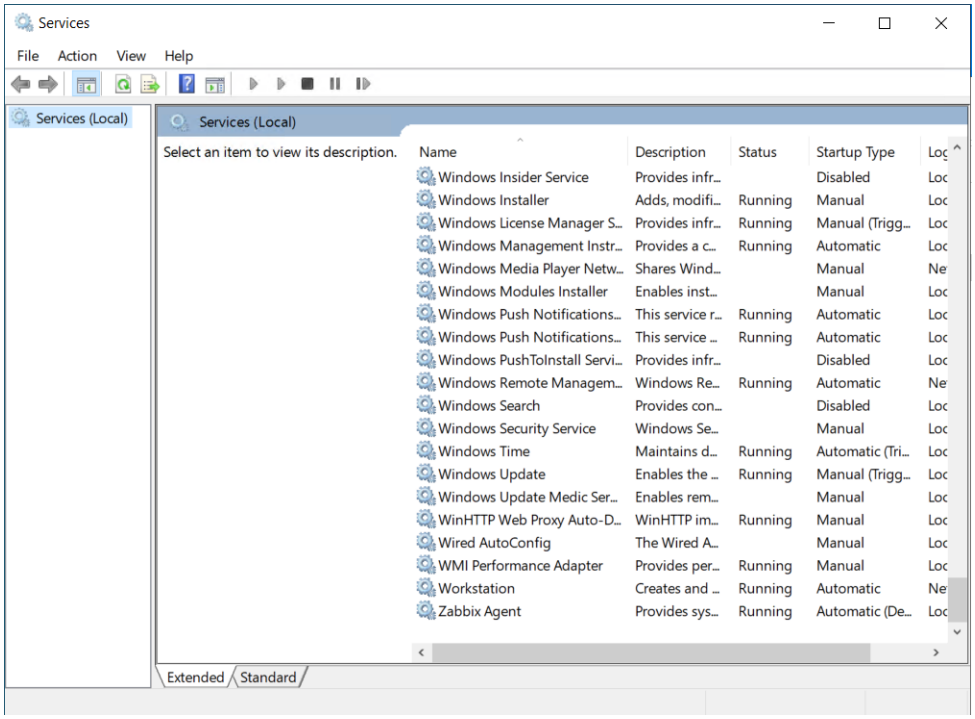

FIGURE 15 – Vérification du service 'Zabbix Agent' dans le gestionnaire de services Windows

6 Monitoring et Tableaux de Bord

6.1 Ajout des hôtes dans Zabbix

Dans l'interface web de Zabbix, nous avons ajouté les deux clients comme nouveaux hôtes.

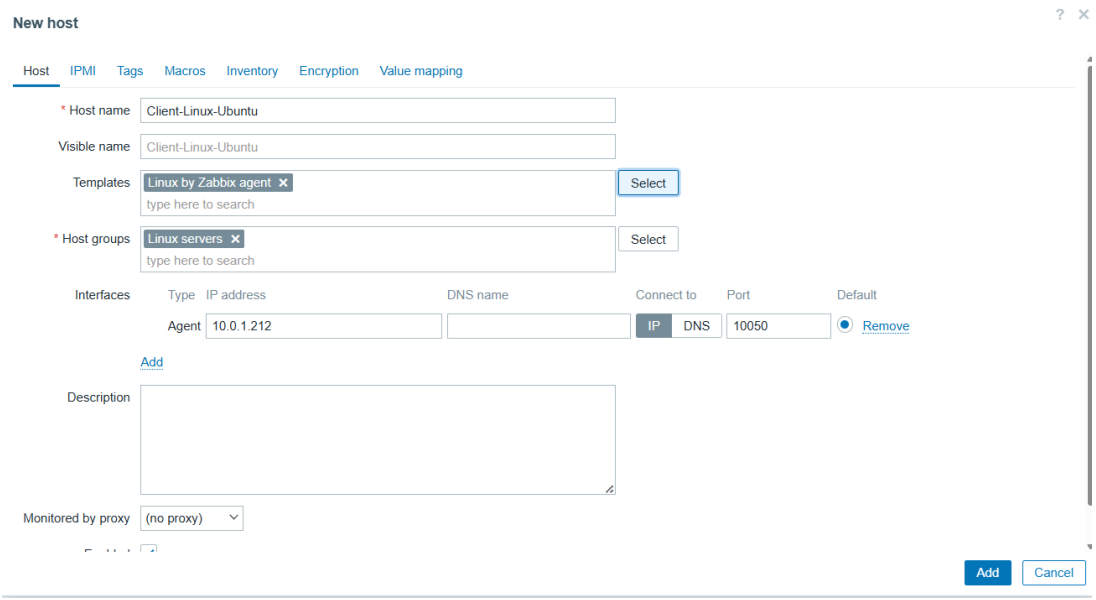

FIGURE 16 – Formulaire de création de l'hôte pour le client Linux

6.2 Statut des hôtes

Après quelques minutes, les deux hôtes apparaissent avec une disponibilité verte.

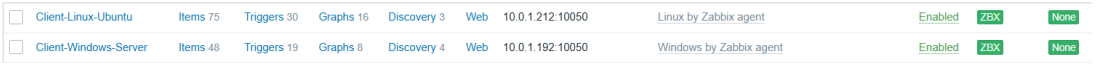

FIGURE 17 – Liste des hôtes dans Zabbix avec disponibilité VERTE (ZBX)

6.3 Visualisation des données

6.3.1 Client Linux

Le graphique suivant montre l'évolution de la charge CPU du client Linux.

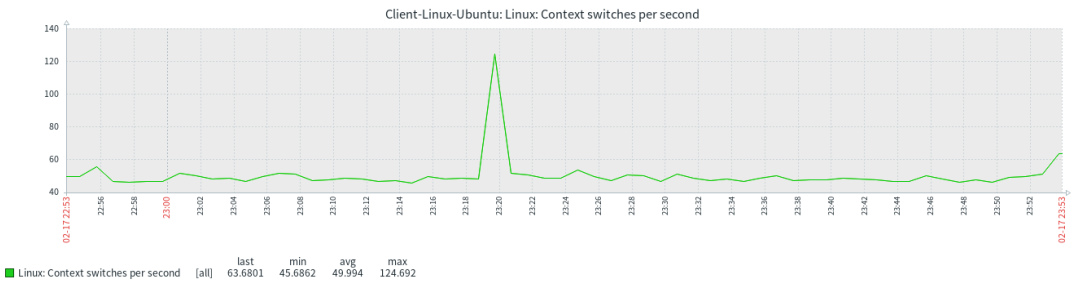

FIGURE 18 – Graphique de la charge CPU du client Linux sur les dernières minutes

6.3.2 Client Windows

Le graphique suivant montre l'utilisation mémoire du client Windows.

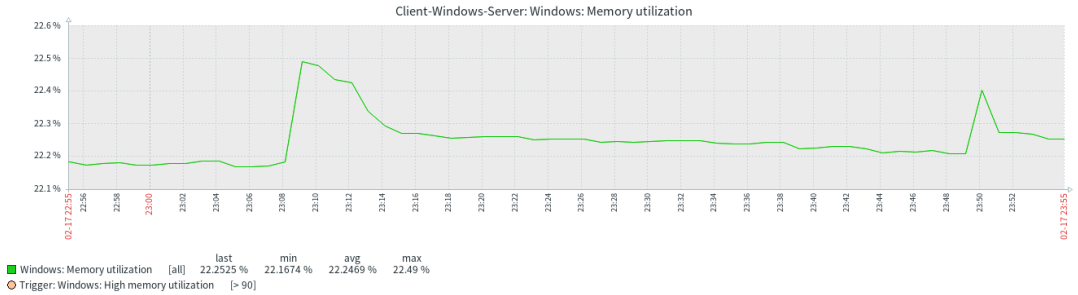

FIGURE 19 – Graphique de l'utilisation mémoire du client Windows

6.4 Données récentes

La vue des données récentes permet de vérifier tous les indicateurs collectés.

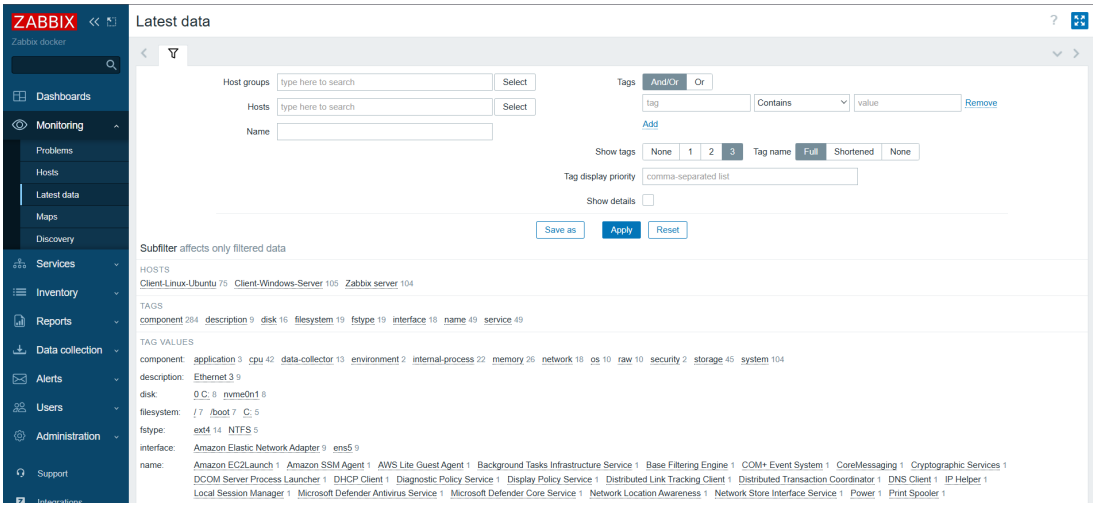

FIGURE 20 – Vue des données récentes pour le client Linux

7 Conclusion

7.1 Bilan du travail réalisé

Ce travail pratique a permis de déployer avec succès une infrastructure de monitoring centralisée sur AWS. L'objectif principal a été atteint : la mise en place d'un serveur Zabbix conteneurisé capable de surveiller un parc hybride composé de machines Linux et Windows.

Les principales réalisations incluent :

— Configuration complète d'un VPC avec sous-réseau public et groupes de sécurité

— Déploiement de trois instances EC2 avec les configurations appropriées

— Installation et configuration de Zabbix via Docker Compose

— Mise en place des agents sur les systèmes Linux et Windows

— Intégration réussie des hôtes dans l'interface Zabbix avec collecte de données

7.2 Difficultés rencontrées et solutions

Plusieurs difficultés ont été rencontrées durant ce TP :

— **Limitations du Learner Lab** : Les instances s'arrêtent automatiquement après une période d'inactivité. La solution a été de redémarrer systématiquement les conteneurs Docker avec `docker-compose up -d` après chaque reprise du Lab.

— **Correspondance des noms d'hôtes** : Un décalage entre le nom configuré dans l'agent et celui déclaré dans Zabbix empêchait la communication. La solution a été de vérifier rigoureusement que les noms correspondent exactement.

— **Flux réseau bloqués** : Initialement, les agents ne parvenaient pas à communiquer avec le serveur. La résolution a consisté à vérifier et ajuster les règles des groupes de sécurité pour autoriser les ports Zabbix depuis l'IP privée du serveur.

7.3 Améliorations possibles

Pour aller plus loin, plusieurs améliorations pourraient être apportées :

— Mise en place d'un Zabbix Proxy pour surveiller des réseaux distants

— Configuration d'alertes par email ou par d'autres canaux (Slack, Teams)

— Création de tableaux de bord personnalisés avec les métriques les plus critiques

— Surveillance d'autres services AWS via des templates spécifiques

— Automatisation du déploiement avec des outils comme Terraform ou CloudFormation

Ce TP a permis de démontrer la puissance et la flexibilité de la combinaison AWS/Docker/Zabbix pour la mise en place d'une solution de monitoring professionnelle, scalable et adaptée aux environnements hybrides.