Université de Mundiapolis

Module : Cloud Computing et Virtualisation

Année Universitaire : 2025/2026

RAPPORT DE PROJET

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

Sujet : Déploiement de Zabbix conteneurisé pour le monitoring d'un parc hybride

---

Réalisé par :
Zeineb BOUZERDA

Encadré par :
Prof. Azeddine KHIAT

29 décembre 2025

Table des matières

1 Introduction et Objectifs

Ce projet s'inscrit dans le cadre du module de Cloud Computing. L'objectif principal est de déployer une infrastructure de monitoring complète sur le cloud Amazon Web Services (AWS), capable de superviser un parc informatique hybride composé de machines Linux et Windows.

Nous avons choisi la solution **Zabbix**, déployée via des conteneurs **Docker**, pour centraliser la remontée d'alertes et l'analyse de performance. Ce rapport détaille les étapes techniques de la mise en place, de la configuration réseau avancée jusqu'à la visualisation des données.

2 Présentation des Technologies AWS Utilisées

Pour répondre aux exigences de sécurité et de performance, nous avons déployé une infrastructure VPC (Virtual Private Cloud) complète. Voici le détail des composants technologiques mobilisés :

2.1 VPC (Virtual Private Cloud)

Le VPC est notre réseau virtuel isolé dans le cloud AWS. Il nous permet de définir notre propre plage d'adresses IP (CIDR Block) et de configurer les tables de routage et les passerelles réseaux.

— **Choix technique :** Création d'un VPC personnalisé pour isoler le trafic de supervision du réseau public par défaut.

2.2 Sous-réseaux (Subnets)

Le VPC a été segmenté en sous-réseaux pour organiser les ressources :

— **Sous-réseau Public :** Héberge les instances nécessitant un accès direct depuis l'extérieur (Serveur Zabbix Web).

— **Sous-réseau Privé (Optionnel) :** Dédié aux bases de données ou aux agents critiques pour limiter l'exposition.

2.3 Table de Routage (Route Table) et IGW

— **Internet Gateway (IGW) :** Composant attaché au VPC permettant la communication entre les instances du sous-réseau public et Internet.

— **Table de routage :** Contient les règles (routes) pour diriger le trafic réseau. Une route par défaut (0.0.0.0/0) pointant vers l'IGW a été configurée pour le sous-réseau public.

2.4 Security Groups (SG)

Les groupes de sécurité agissent comme un pare-feu virtuel au niveau de l'instance pour contrôler le trafic entrant et sortant. Nous avons appliqué le principe du moindre privilège (n'ouvrir que les ports nécessaires).

3 Réalisation et Configuration de l'Infrastructure

Cette partie détaille les étapes techniques de la construction du réseau et du déploiement des serveurs, appuyée par les captures d'écran de la console AWS.

3.1 Configuration du Réseau (VPC et Subnets)

Nous avons d'abord instancié le VPC et défini les plages d'adresses IP.

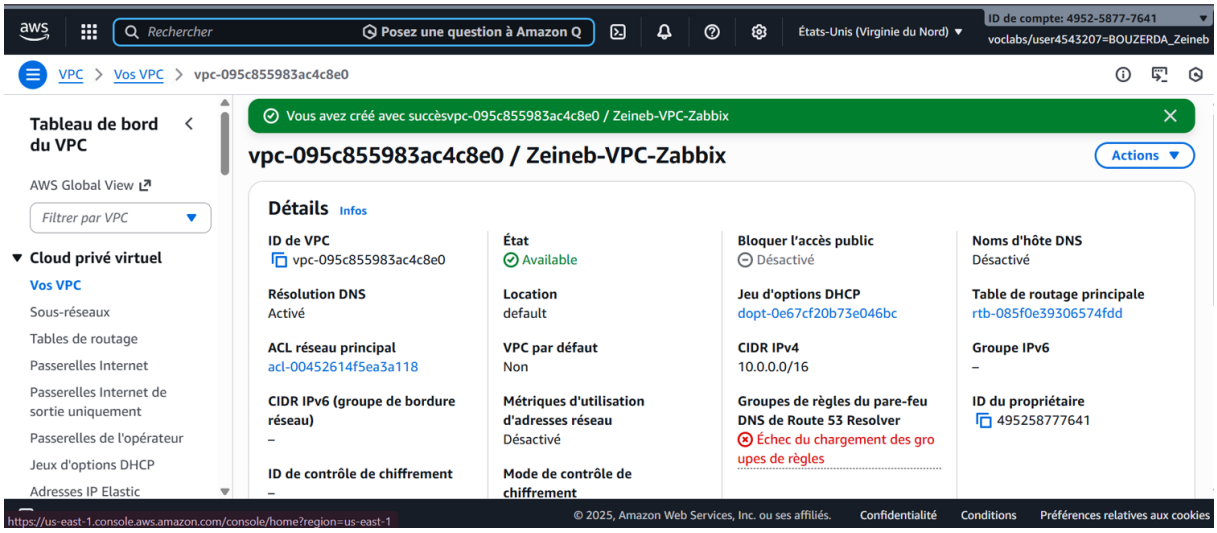

FIGURE 1 – Vue de la console AWS montrant la création du VPC et son CIDR.

Ensuite, nous avons configuré les sous-réseaux pour séparer logiquement nos zones de disponibilité.

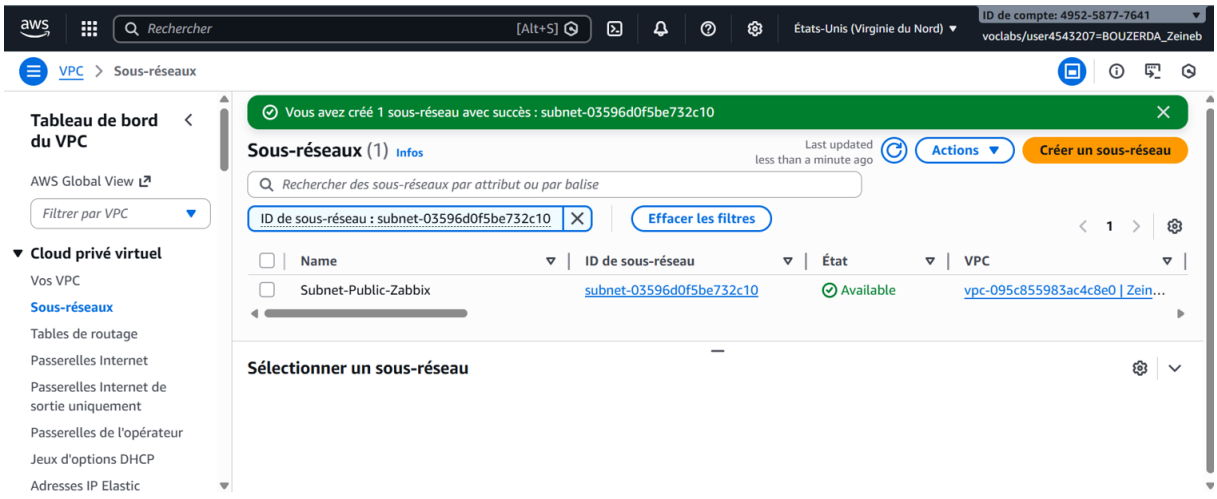

FIGURE 2 – Configuration des sous-réseaux (Subnets) associés au VPC.

3.2 Connectivité Internet et Routage

Pour permettre aux instances de télécharger les paquets (Docker, Zabbix) et d'être administrées, nous avons attaché une Internet Gateway et configuré la table de routage.

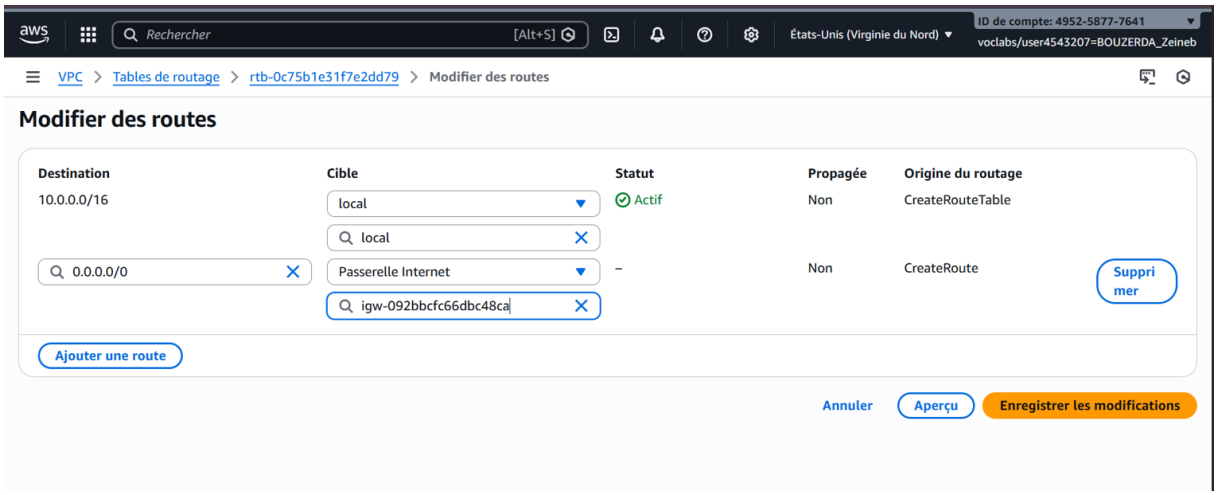

FIGURE 3 – Table de routage montrant la route 0.0.0.0/0 vers l'Internet Gateway.

3.3 Configuration de la Sécurité (Security Groups)

La configuration des pare-feux est critique pour Zabbix. Nous avons créé un groupe de sécurité spécifique `Zabbix-SG` autorisant les ports 80 (HTTP), 22 (SSH), 3389 (RDP) et 10050/10051 (Zabbix).

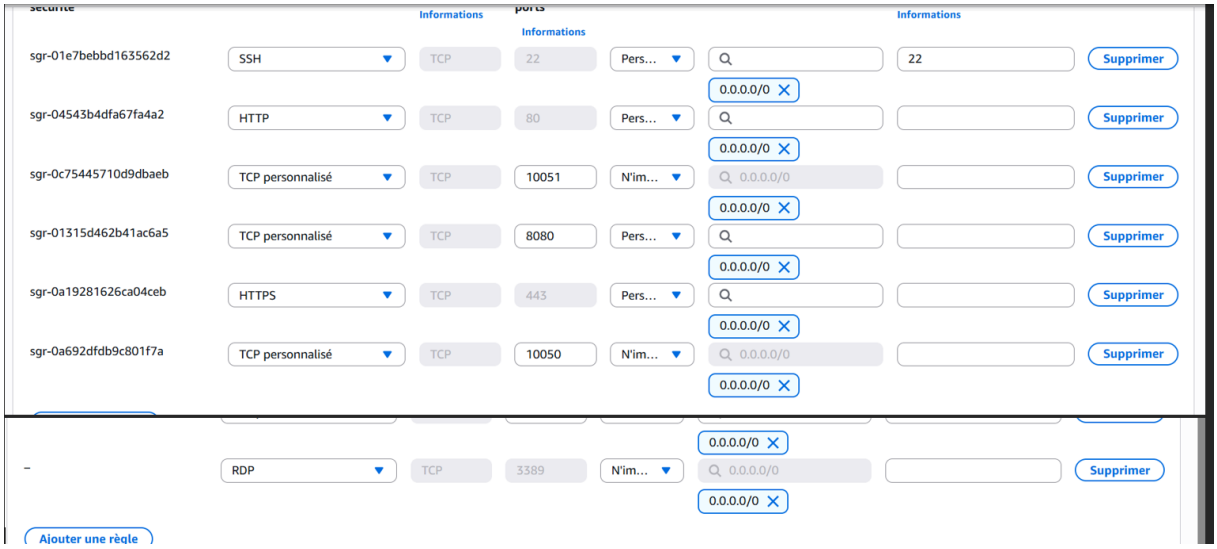

FIGURE 4 – Capture des règles entrantes (Inbound Rules) du Security Group AWS.

3.4 Déploiement des Instances EC2

Une fois le réseau prêt, nous avons lancé les trois instances requises par le cahier des charges :

- **Serveur Zabbix :** Ubuntu 22.04 LTS (*t3.large*).
- **Client Linux :** Ubuntu 22.04 LTS (*t3.medium*).
- **Client Windows :** Windows Server 2019 Base (*t3.large*).

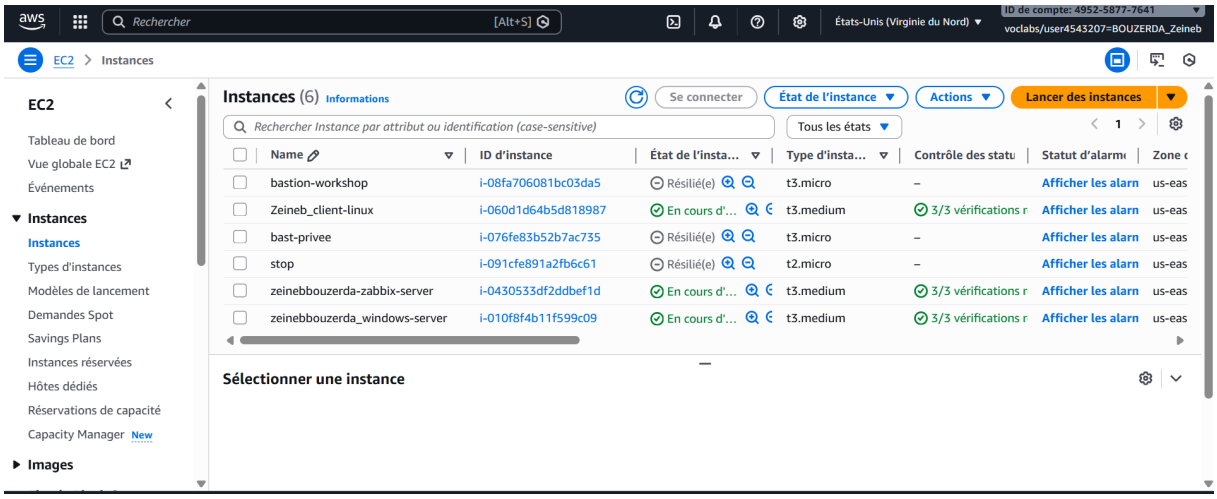

FIGURE 5 – Vue globale des instances EC2 en cours d'exécution dans la console.

4 Installation du Serveur de Supervision

4.1 Installation de Docker et Docker Compose

Sur l'instance Ubuntu "Serveur Zabbix", nous avons procédé à l'installation de Docker pour conteneuriser l'application.

```
ubuntu@ip-10-0-1-53:~$ sudo apt install -y docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
docker.io is already the newest version (28.2.2-0ubuntu1~24.04.1).
docker.io set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 68 not upgraded.
```

FIGURE 6 – Installation de Docker sur le serveur Ubuntu.

4.2 Déploiement via Docker Compose

Nous avons utilisé un fichier `docker-compose.yml` définissant les services Zabbix. La commande `docker ps` valide que tous les conteneurs (Serveur, Web, DB) sont actifs.

```
ubuntu@ip-10-0-1-53:~/zabbix$ sudo docker ps
CONTAINER ID   IMAGE                                          COMMAND                  CREATED         STATUS                   P
ORTS                                                                                   NAMES
674f90f437f8   zabbix/zabbix-web-apache-mysql:ubuntu-6.4-latest   "docker-entrypoint.sh"   2 minutes ago   Up 2 minutes (healthy)   0
.0.0.0:80->8080/tcp, [::]:80->8080/tcp, 0.0.0.0:443->8443/tcp, [::]:443->8443/tcp   zabbix_zabbix-web_1
2c492be51340   zabbix/zabbix-server-mysql:ubuntu-6.4-latest    "/usr/bin/tini -- /u…"   2 minutes ago   Up 2 minutes             0
.0.0.0:10051->10051/tcp, [::]:10051->10051/tcp                                         zabbix_zabbix-server_1
9ebf57844c72   mysql:8.0                                      "docker-entrypoint.s…"   2 minutes ago   Up 2 minutes             3
306/tcp, 33060/tcp                                                                     zabbix_zabbix-db_1
```

FIGURE 7 – Vérification des conteneurs en cours d'exécution via la commande `docker ps`.

5 Configuration des Agents (Clients)

5.1 Client Linux (Ubuntu)

Sur le client Linux, l'agent Zabbix a été installé et configuré (`/etc/zabbix/zabbix_agentd.conf`) pour pointer vers l'IP du Serveur.

```
ServerActive=3.87.213.249

### 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=Zeineb_client-linux

### Option: HostnameItem
#       Item used for generating Hostname if it is undefined. Ignored if Hostname is defined.
#       Does not support UserParameters or aliases.
#
# Mandatory: no
# Default:
```

FIGURE 8 – Configuration de l'adresse IP du serveur dans l'agent Linux.

5.2 Client Windows

Sur le serveur Windows, nous avons installé l'agent via le programme d'installation MSI officiel.

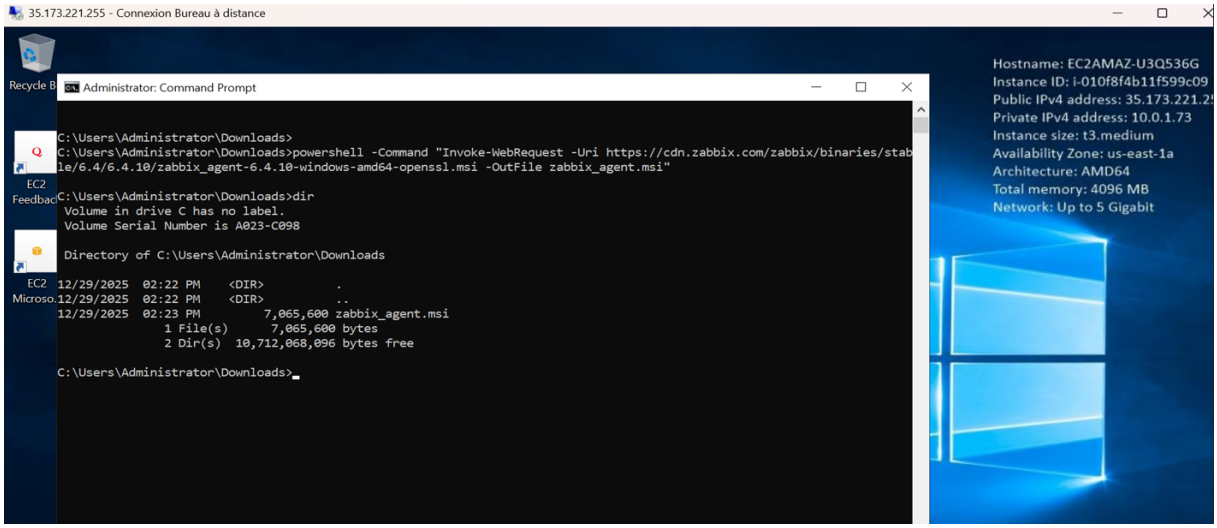

FIGURE 9 – Installation de l'agent Zabbix sur Windows Server.

6 Tests et Résultats

6.1 Détection des Hôtes

Une fois les agents configurés, nous avons ajouté les hôtes dans l'interface Web de Zabbix. L'icône "ZBX" verte confirme la connectivité.

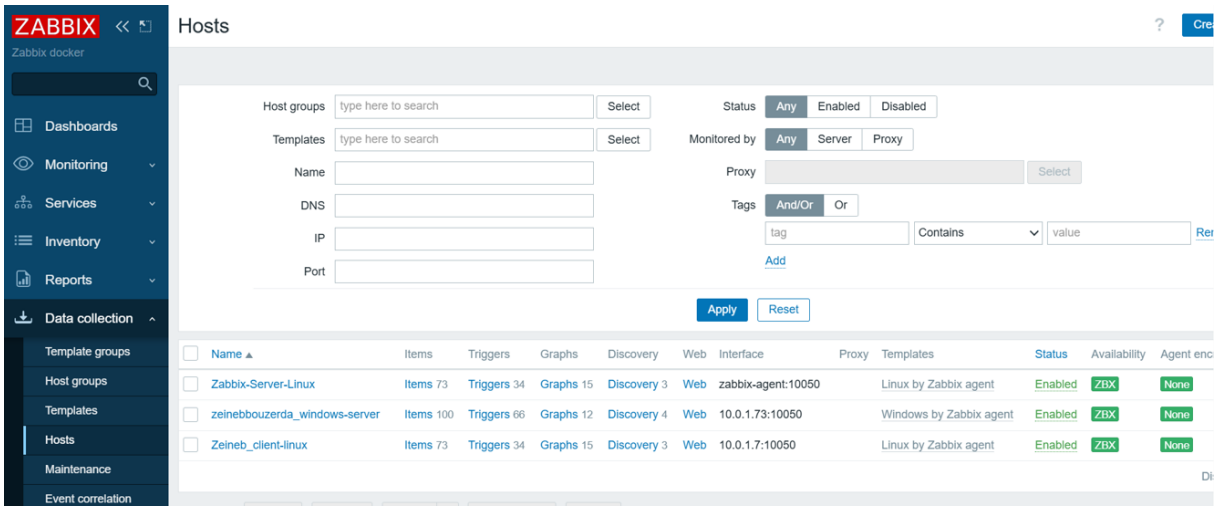

FIGURE 10 – Interface Zabbix : Les agents Linux et Windows sont connectés (statut vert).

6.2 Visualisation des Données

Nous recevons désormais les métriques en temps réel (CPU, Mémoire, Disque) des deux machines.

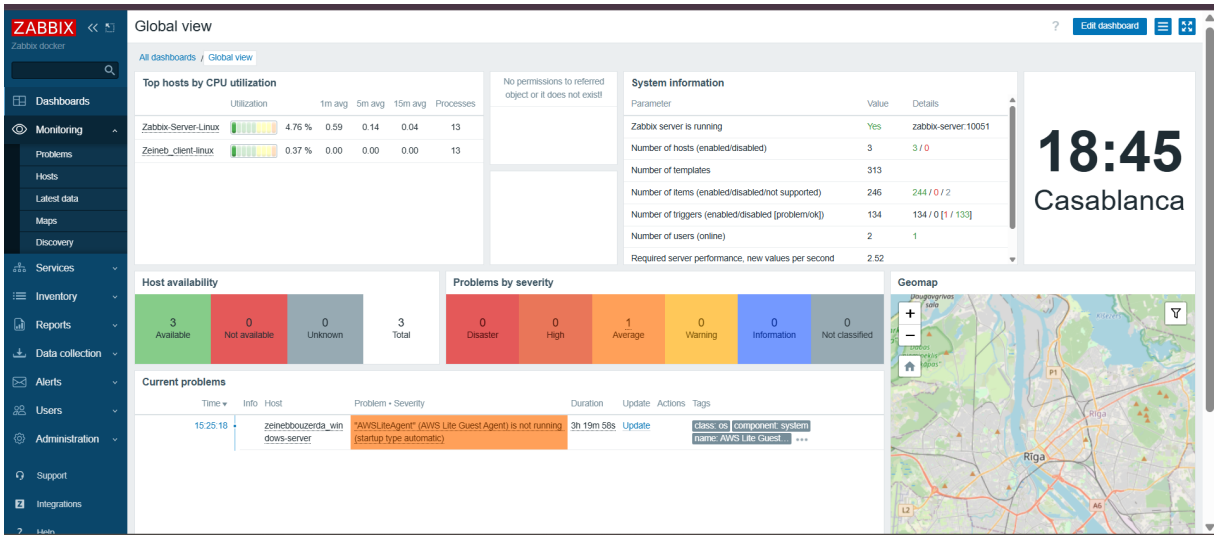

FIGURE 11 – Graphique de l'utilisation CPU remonté par l'agent Windows.

7 Conclusion

La mise en œuvre de ce projet a permis de valider la faisabilité d'une supervision centralisée sur AWS. L'utilisation de Zabbix conteneurisé offre une solution robuste, capable de s'adapter à un environnement hybride. Les tests effectués démontrent que les données remontent correctement aussi bien depuis Linux que Windows, répondant ainsi à la totalité des exigences du cahier des charges.