

PROJET:

Mise en œuvre d'une infrastructure cloud de supervision centralisée sous AWS : Déploiement de Zabbix conteneurisé pour le monitoring d'un parc hybride

Ingénierie Informatique option Cybersécurité



Réalisé par :

MONTASSIF Fatima Ezzahra

Encadré par :

Prof. Azeddine KHIAT

Année universitaire :

2025-2026

Table de Matières

1 INTRODUCTION.....	2
2 Architecture et Configuration Réseau.....	4
2.1 Mise en place du VPC et du Sous-réseau.....	4
2.2 Configuration des groupes de sécurité.....	5
2.3 Connectivité Internet et Routage.....	6
3 Création des Instances EC2.....	8
3.1 Tableau récapitulatif des ressources.....	8
3.2 État du déploiement dans la console AWS.....	8
4 Déploiement du Serveur Zabbix via Docker.....	9
4.1 Installation de Docker et Docker Compose.....	9
4.2 Configuration de l'Orchestration (Docker Compose).....	10
4.3 Lancement et Vérification.....	11
5 Configuration des agents de supervision.....	13
5.1 Installation de l'agent sur Linux-client.....	13
5.2 Configuration du service.....	14
5.3 Initialisation et persistance.....	14
5.4 Enregistrement sur l'interface Web.....	14
5.5 Installation de l'agent sur Windows.....	16
5.6 Validation de la Connectivité.....	17
5.7 Enregistrement et Supervision.....	17
6 Monitoring et Validation.....	18
6.1 Crédit du Dashboard Global.....	18
— Configuration du Graphique CPU :.....	19
— Configuration du Graphique RAM :.....	20
6.2 Mise en place d'un Trigger (Alerte Proactive).....	20
6.3 Synthèse Visuelle (Dashboard Consolidé).....	21
7 Conclusion.....	22
7.1 Bilan technique et objectifs atteints.....	22
7.2 Analyse des défis et solutions apportées.....	22
List of Tables.....	23

Liste des Figures

1	Création du VPC Projet Zabbix	4
2	Configuration du sous-réseau (Subnet)	4
3	Règles du groupe de sécurité Zabbix Server	5
4	Règles du groupe de sécurité des Agents	5
5	Création de l'Internet Gateway	6
6	Configuration de la table de routage	6
7	Visualisation globale de l'architecture réseau AWS	6
8	Création des instances EC2 en état dans la console AWS	7
9	Installation de Docker sur l'instance Ubuntu	8
10	Vérification de la version de Docker installée	8
11	Processus de création des conteneurs	10
12	Vérification de l'état des services	10
13	Interface et tableau de bord Zabbix	10
14	Installation du dépôt Zabbix sur l'hôte Linux	11
15	Finalisation de l'installation de l'agent	12
16	Redémarrage et vérification du statut du service zabbix-agent	12
17	Création de l'hôte Linux-client dans le tableau de bord Zabbix	13
18	Données reçus de Linux-Client sur l'interface de Zabbix	13
19	Interface de configuration de l'installation de l'agent Windows	14
20	Test de connectivité réussi via PowerShell (TcpTestSucceeded)	14
21	Configuration du nouvel hôte Windows dans l'interface Zabbix	15
22	Visualisation des données reçues en temps réel	15
23	Configuration du widget de monitoring CPU pour les clients hybrides	16
24	Mise en place du suivi de l'utilisation de la mémoire vive (RAM)	16
25	Définition des seuils d'alerte et de la sévérité du Trigger	17
26	Aperçu du Dashboard Global regroupant l'ensemble des métriques et alertes	17

Liste des Tables

1	Caractéristiques techniques des instances déployées sur AWS
---	-------------------------------------------------------------

2 Introduction

Dans le contexte de la gestion des infrastructures informatiques, ce projet vise donc la mise en place d'une solution de supervision centralisée hébergée sur le cloud Amazon Web Services (AWS).

L'objectif est d'implémenter une instance de Zabbix conteneurisée en s'appuyant sur Docker, afin de permettre le monitoring en temps réel d'un environnement informatique hybride, contenant deux instances EC2, une Linux et Windows.

Le projet se structure autour de trois axes principaux :

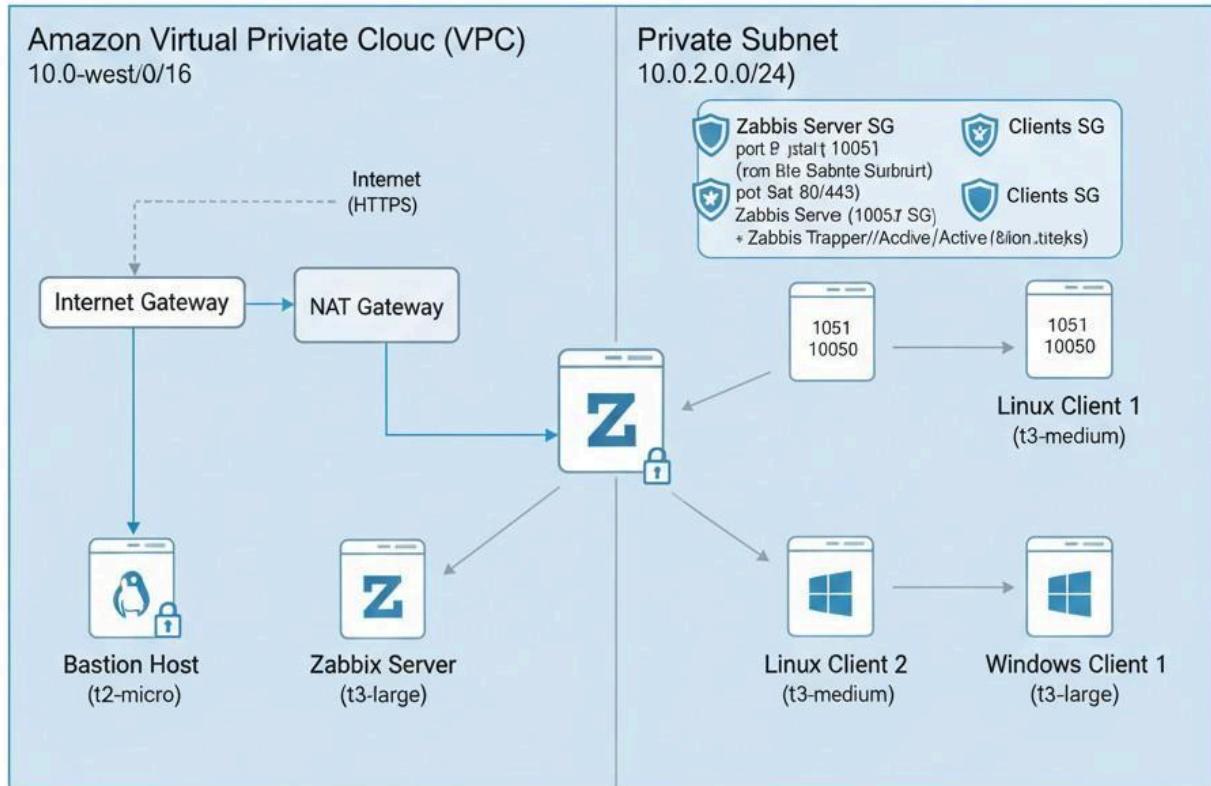
a- Configuration de l'infrastructure : Création d'un VPC avec des sous-réseaux et des groupes de sécurité adaptés pour gérer les flux nécessaires à la supervision (ports 10050 et 10051) ainsi que l'accès web (ports 80 et 443).

b- Déploiement du serveur Zabbix : Installation du serveur Zabbix en utilisant Docker sur une instance Ubuntu.

c- Supervision du parc informatique : Installation et paramétrage des agents Zabbix sur des machines clientes Ubuntu et Windows Server afin de collecter des métriques essentielles concernant l'utilisation du processeur, de la mémoire RAM.

Lien du dépôt GitHub :

Architecture Cloud de Supervision Centralisée - AWS & Zabbix



Legend

- AWS EC2 Instance
- AWS VPC
- Internet Gateway
- Security Group

Zabbix
Monitoring System

3 Architecture et Configuration Réseau

L'infrastructure réseau est isolée au sein d'un **VPC** spécifique avec une configuration stricte des flux de sécurité via les **Security Groups**.

3.1 Mise en place du VPC et du Sous-réseau

D'abord, on crée le VPC en précisant son nom, avec son bloc **CIDR : 10.0.0.0/16**

The screenshot shows the 'Créer un VPC' (Create a VPC) wizard. It starts with the 'Paramètres VPC' (VPC settings) step. Under 'Ressources à créer', 'VPC uniquement' is selected. In the 'Identification de nom - facultatif' section, the name 'VPC-Zabbix' is entered. The 'Bloc d'adresses CIDR IPv4' section shows 'Entrée manuelle CIDR IPv4' selected, with '10.0.0.0/16' specified. The 'CIDR IPv4' note indicates the range must be between /16 and /28. The 'Bloc CIDR IPv6' section has 'Aucun bloc d'adresses CIDR IPv6' selected. The wizard navigation bar at the top includes 'VPC > Vos VPC > Créer un VPC' and icons for help, copy, and close.

Figure 1 – Création du VPC Projet Zabbix

Par la suite, on a créé un sous-réseau pour le VPC créé auparavant avec un bloc CIDR : 10.0.0.0/24.

The screenshot shows the 'Créer un sous-réseau' (Create a subnet) wizard. It starts with the 'VPC' step, where 'vpc-0f64f0d4eb7ae19fc (VPC-Zabbix)' is selected from the dropdown. The 'CIDR de VPC associés' section shows 'CIDR IPv4' set to '10.0.0.0/16'. The 'Paramètres du sous-réseau' (Subnet settings) step follows, with the note 'Précisez les blocs d'adresse CIDR et la zone de disponibilité pour le sous-réseau.' The 'Sous-réseau 1 sur 1' (Subnet 1 of 1) section shows 'Nom du sous-réseau (subnet)' as 'Subnet-VPC-Zabbix' and 'Zone de disponibilité' as 'États-Unis (Virginie du Nord) / use1-az4 (us-east-1c)'. The wizard navigation bar at the top includes 'VPC > Sous-réseaux > Créer un sous-réseau' and icons for help, copy, and close.

Sous-réseaux (1) Infos						
Rechercher des sous-réseaux par attribut ou par balise <input type="text"/> Effacer les filtres						
<input type="checkbox"/> Name	ID de sous-réseau	État	VPC	Bloquer l'accès	Détails	Actions
<input type="checkbox"/> Subnet-VPC-Zabbix	subnet-07b6017278aa9cfed	Available	vpc-0f64f0d4eb7ae19fc VPC-	Désactivé		Actions

Figure 2 – Configuration du sous-réseau (Subnet)

3.2 Configuration des groupes de sécurité

Nous avons créé deux groupes de sécurité pour filtrer l'accès aux instances :

- **Security-Server-Zabbix-Montassif** : Ce groupe est destiné au serveur de supervision et définit les règles entrantes suivantes :

Port 22 : Protocole SSH pour la connexion à distance.

Ports 80 et 443 : Protocoles HTTP/HTTPS pour l'accès à l'interface web Zabbix.

Port 10051 : Protocole TCP, utilisé par le serveur pour recevoir les informations envoyées par les agents.

Détails de base

Nom du groupe de sécurité [Informations](#)
Security-Server-Zabbix-Montassif
Le nom ne peut pas être modifié après sa création.

Description [Informations](#)
Autorisation serveur Zabbix

VPC [Informations](#)
vpc-0f64f0d4eb7ae19fc (VPC-Zabbix)

Règles entrantes [Informations](#)

Type	Informations	Protocole	Informations	Plage de ports	Informations	Source	Informations	Description - facultatif	Informations
SSH	Informations	TCP	Informations	22	N'im... ▾	0.0.0.0/0 X	Supprimer		
HTTP	Informations	TCP	Informations	80	Mon IP ▾	41.141.123.7/32 X	Supprimer		
HTTPS	Informations	TCP	Informations	443	Mon IP ▾	41.141.123.7/32 X	Supprimer		
TCP personnalisé	Informations	TCP	Informations	10051	N'im... ▾	0.0.0.0/0 X	Supprimer		

[Ajouter une règle](#)

Figure 3 – Règles du groupe de sécurité Zabbix Server

- **Agents-Server-Montassif** : Ce groupe est appliqué aux clients et agents de surveillance:

Port 10050 : TCP pour l'écoute de l'agent.

Port 3389 : RDP pour l'accès distant aux machines Windows.

Port 22 : SSH pour l'administration des machines Linux.

The screenshot shows the AWS VPC 'Groupes de sécurité' (Security Groups) section. In the top navigation bar, it says 'VPC > Groupes de sécurité > Créer un groupe de sécurité'. The main title is 'Créer un groupe de sécurité' with a 'Informations' link. A note below states: 'Un groupe de sécurité agit comme un pare-feu virtuel pour votre instance afin de contrôler le trafic entrant et sortant. Pour créer un groupe de sécurité, complétez les champs ci-dessous.' The 'Détails de base' (Base Details) section includes fields for 'Nom du groupe de sécurité' (Name) containing 'Agent-Server-Montassif', 'Description' (Description) containing 'Autorise les agents', and a 'VPC' dropdown set to 'vpc-0f64f0d4eb7ae19fc (VPC-Zabbix)'. The 'Règles entrantes' (Ingress Rules) section lists three rules:

Type	Protocole	Plage de ports	Source	Description - facultatif
SSH	TCP	22	N'im... (0.0.0.0/0)	Supprimer
RDP	TCP	3389	Mon IP (41.141.123.7/32)	Supprimer
TCP personnalisé	TCP	10050	N'im... (0.0.0.0/0)	Supprimer

An 'Ajouter une règle' (Add a rule) button is at the bottom left of the rules table.

Figure 4 – Règles du groupe de sécurité des Agents

3.3 Connectivité Internet et Routage

Pour permettre aux ressources d'accéder à Internet pour les mises à jour ou installation avec Docker, une passerelle a été configuré en s'appuyant sur une:

- **Internet Gateway (IGW)** : Création et attachement de la passerelle au VPC.
- **Table de Routage** : Ajout d'une route par défaut (0.0.0.0/0) pointant vers l'IGW.

VPC > Passerelles Internet > Créer une passerelle Internet

Créer une passerelle Internet Infos

Une passerelle Internet est un routeur virtuel qui connecte un VPC à Internet. Pour créer une nouvelle passerelle Internet, spécifiez le nom de la passerelle ci-dessous.

Paramètres de passerelle Internet

Identification de nom
Crée une identification avec une clé du « Nom » et une valeur que vous spécifiez.

Balises - facultatif
Une balise est une étiquette que vous attribuez à une ressource AWS. Chaque balise est constituée d'une clé et d'une valeur facultative. Vous pouvez utiliser des balises pour rechercher et filtrer vos ressources ou réaliser le suivi de vos coûts AWS.

Clé	Valeur - facultatif
<input type="text" value="Name"/>	<input type="text" value="IGW-Zabbix-Montassif"/> X Supprimer

Ajouter une nouvelle balise

Vous pouvez ajouter 49 d'autres balises.

Annuler Créer une passerelle Internet

IGW-Zabbix-Montassif igw-0229b02cc27074b94 Attached vpc-0f64f0d4eb7ae19fc | VPC-Zabbix 702696

igw-0229b02cc27074b94 / IGW-Zabbix-Montassif

Détails Balises

Détails

ID de passerelle Internet igw-0229b02cc27074b94	État Attached	ID de VPC vpc-0f64f0d4eb7ae19fc VPC-Zabbix	Propriétaire 702696232812
--------------------------------------------------------------------------------------	-----------------------------------------------------	-----------------------------------------------------------------------------------	----------------------------------------------------------------

⚙️ ▼

Figure 5 – Création de l'Internet Gateway

VPC > Tables de routage > Créer une table de routage

Créer une table de routage Infos

Une table de routage spécifie comment les paquets sont transférés entre les sous-réseaux au sein de votre VPC, sur Internet et votre connexion VPN.

Paramètres de la table de routage

Nom - facultatif
Crée une balise avec une clé « Name » et une valeur à spécifier.

VPC
VPC à utiliser pour cette table de routage.

vpc-0f64f0d4eb7ae19fc (VPC-Zabbix)

Balises
Une balise est une étiquette que vous attribuez à une ressource AWS. Chaque balise est constituée d'une clé et d'une valeur facultative. Vous pouvez utiliser des balises pour rechercher et filtrer vos ressources ou réaliser le suivi de vos coûts AWS.

Clé	Valeur - facultatif
<input type="text" value="Name"/>	<input type="text" value="Route-table-Zabbix"/> X Supprimer

Ajouter une nouvelle balise

Vous pouvez ajouter 49 d'autres balises.

Annuler Créer une table de routage

<input type="text" value="0.0.0.0"/> X	Passerelle Internet ▼	-	Non	CreateRoute	Supprimer
<input type="text" value="igw-0229b02cc27074b94"/> X					

Ajouter une route

Annuler Aperçu Enregistrer les modifications

Figure 6 – Configuration de la table de routage



Figure 7 – Visualisation globale de l'architecture réseau AWS

4 Crédation des Instances EC2

Le déploiement de l'infrastructure de supervision repose sur trois instances **Amazon EC2** distinctes, dimensionnées pour répondre aux besoins spécifiques de chaque rôle (particulièrement pour le serveur Zabbix et le client Windows qui nécessitent plus de ressources).

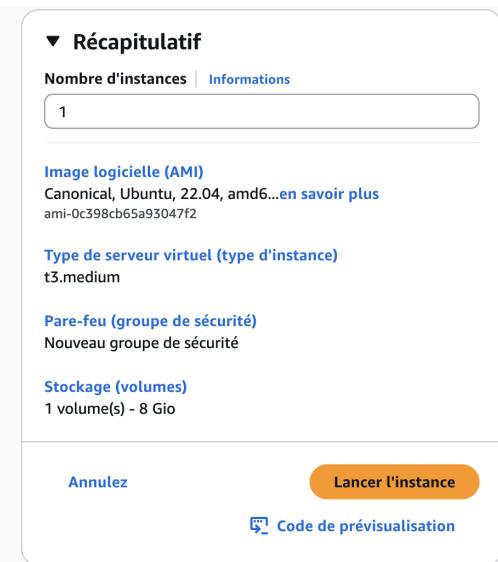
4.1 Tableau récapitulatif des ressources

Rôle	Type d'instance	Système d'Exploitation	Usage / Fonction
Serveur Zabbix	t3.large	Ubuntu 22.04 LTS	Serveur Docker & Dashboard
Client Linux	t3.medium	Ubuntu 22.04 LTS	Monitoring Agent Linux
Client Windows	t3.large	Windows Server 2022	Monitoring Agent Windows

Table 1 – Caractéristiques techniques des instances déployées sur AWS

4.2 État du déploiement dans la console AWS

Une fois les instances lancées et configurées avec leurs groupes de sécurité respectifs, nous pouvons vérifier leur état de fonctionnement depuis le tableau de bord EC2.



The screenshot shows the AWS EC2 instance creation interface and the resulting EC2 Instances list.

Instance Creation:

- Step 1: Nom et balises** (Informations)
 - Nom: Client_Windows
 - Ajouter des balises supplémentaires
- Step 2: Images d'applications et de systèmes d'exploitation (Amazon Machine Image)** (Informations)
 - Une AMI contient le système d'exploitation, le serveur d'applications et les applications de votre instance. Si aucune AMI appropriée ne s'affiche ci-dessous, utilisez le champ de recherche ou choisissez Parcourir d'autres AMI.
 - Search bar: Effectuer une recherche dans notre catalogue complet, qui comprend des milliers d'images d'applications et de systèmes d'exploitation.
 - Recent Images: Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux, Debian.
 - Windows Image: Microsoft, Red Hat, SUSE, Debian.
 - Search: Explorer plus d'AMI (Y compris les AMI d'AWS, de Marketplace et de la communauté).
- Summary (Récapitulatif):**
 - Nombre d'instances: 1
 - Image logicielle (AMI): Microsoft Windows Server 2022 ...en savoir plus
 - Type de serveur virtuel (type d'instance): t3.large
 - Pare-feu (groupe de sécurité): Nouveau groupe de sécurité
 - Stockage (volumes): 1 volume(s) - 30 Gio
 - Buttons: Annuler, Lancer l'instance, Code de prévisualisation.

EC2 Instances List:

ID d'instance	Nom	Etat de l'instance	Type d'instance	Contrôle des statut	Statut d'alarme	Zone de disponibilité	DNS IPv4 public	Adresse IP
i-027f1441e1624edc3	Serveur Zabbix	En cours d'initialisation	t3.large	3/3 vérifications r	Afficher les alarmes	us-east-1c	ec2-18-234-211-163.co...	18.2
i-032e0980a752b9dde	TP_Securite_R...	En cours d'initialisation	t2.micro	2/2 vérifications r	Afficher les alarmes	us-east-1c	ec2-98-84-113-202.co...	98.8
i-0d1295747f321142e	Client_Linux	En cours d'initialisation	t3.medium	3/3 vérifications r	Afficher les alarmes	us-east-1c	ec2-18-207-122-92.co...	18.2
i-09e50213e2bc509df	Client_Windows	En cours d'initialisation	t3.large	-	Afficher les alarmes	us-east-1c	ec2-34-224-39-12.com...	34.2

Figure 8 – Création des instances EC2 dans la console AWS

5 Déploiement du Serveur Zabbix via Docker

Nous nous connectons à l'instance **Zabbix-server** afin de déployer le serveur Zabbix. Pour garantir la portabilité et la facilité de maintenance, nous avons opté pour un déploiement via **conteneurs Docker**. Cette section détaille l'installation de l'environnement et la configuration des services.

5.1 Installation de Docker et Docker Compose

La première étape consiste à installer le moteur Docker ainsi que Docker Compose pour orchestrer nos services.

```
sudo apt update && sudo apt install docker.io docker-compose -y
```

```
ubuntu@ip-172-31-29-12:~$ sudo apt update && sudo apt install docker . io docker - compose -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [5652 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 c-n-f Metadata [286 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [217 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse Translation-en [112 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 c-n-f Metadata [8372 B]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [3163 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [485 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [19.1 kB]
Gathering packages and their dependencies...done.

```

Figure 9 – Installation de Docker sur l’instance Ubuntu

```
ubuntu@ip-172-31-29-12:~$ docker --version
Docker version 28.2.2, build 28.2.2-0ubuntu1-22.04.1
ubuntu@ip-172-31-29-12:~$ docker-compose --version
docker-compose version 1.29.2, build unknown
ubuntu@ip-172-31-29-12:~$
```

Figure 10 – Vérification de la version de Docker installée

5.2 Configuration de l’Orchestration (Docker Compose)

Nous avons structuré notre infrastructure dans un fichier docker-compose.yml. Ce fichier définit trois services essentiels : le serveur Zabbix, l’interface Web (Nginx) et une base de données MySQL persistante.

Pour des raisons de **sécurité**, les informations sensibles (mots de passe, utilisateurs) ne sont pas inscrites en dur mais stockées dans un fichier de variables d’environnement nommé .env.

```
ubuntu@ip-172-31-29-12:~$ mkdir project-Zabbix
ubuntu@ip-172-31-29-12:~$ cd project-Zabbix/
ubuntu@ip-172-31-29-12:~/project-Zabbix$ nano docker-compose.yml
ubuntu@ip-172-31-29-12:~/project-Zabbix$ nano .env
ubuntu@ip-172-31-29-12:~/project-Zabbix$
```

```
GNU nano 6.2                                            docker-compose.yml
services:
  zabbix-db:
    image: mysql:8.0
    container_name: zabbix-db
    restart: always
    command:
      --character-set-server=utf8
      --collation-server=utf8_bin
      --default-authentication-plugin=mysql_native_password
      --log_bin_trust_function_creator=1
    environment:
      - MYSQL_USER=${MYSQL_USER}
      - MYSQL_PASSWORD=${MYSQL_PASSWORD}
      - MYSQL_ROOT_PASSWORD=${MYSQL_ROOT_PASSWORD}
      - MYSQL_DATABASE=${MYSQL_DATABASE}
    volumes:
      - zabbix-db-data:/var/lib/mysql

  zabbix-server:
    image: zabbix/zabbix-server-mysql:ubuntu-6.4-latest
    container_name: zabbix-server
    restart: always
    ports:
      - "10051:10051"
    environment:
      - DB_SERVER_HOST=zabbix-db
      - MYSQL_USER=${MYSQL_USER}
      - MYSQL_PASSWORD=${MYSQL_PASSWORD}
      - MYSQL_DATABASE=${MYSQL_DATABASE}
```

Figure 2 – Fichier docker-compose.yml

```
GNU nano 6.2                                              .env *
# Configuration Base de donnees
MYSQL_ROOT_PASSWORD = root
MYSQL_USER = zabbix
MYSQL_PASSWORD = root
MYSQL_DATABASE = zabbix
# Configuration System
ZBX_PHP_TZ = Europe / Paris ( Afrique / Casablanca )
```

Figure 3 – Le fichier .env

5.3 Lancement et Vérification

Le déploiement est initié via la commande suivante :

docker - compose up -d

Une fois les services démarrés, l'interface de gestion Zabbix est accessible via l'adresse IP publique de l'instance sur le port web 80.

```
ubuntu@ip-172-31-29-12:~/project-Zabbix$ sudo docker-compose up -d
Creating network "project-zabbix_default" with the default driver
Creating volume "project-zabbix_zabbix-db-data" with default driver
Pulling zabbix-db (mysql:8.0)...
8.0: Pulling from library/mysql
658e67031dba: Pull complete
ef7bcbeff3f90: Pull complete
caeee3c6664fb: Pull complete
80caa0f442af: Pull complete
ea7ef84dcdea: Pull complete
f40e06a2cd88: Pull complete
83bab8f9c85f: Pull complete
655eb0dfc8fe: Pull complete
c9ae3e76dfc9: Pull complete
cc3a161f487a: Pull complete
af8a12fa4e1: Pull complete
Digest: sha256:9c3380eac945af0736031b200027f581925927c81e010056214a4bd6b6693714
Status: Downloaded newer image for mysql:8.0
Pulling zabbix-server (zabbix/zabbix-server-mysql:ubuntu-6.4-latest)...
ubuntu-6.4-latest: Pulling from zabbix/zabbix-server-mysql
de44b265507a: Extracting [=====] 25.56MB/29.75MB
c15ab064eea3: Download complete
a2ab853cdf6c: Download complete
5f5562f77cda: Download complete
66367a2ddd2d: Download complete
210f7f100111: Downloading [=====] 65.54MB/71.17MB
4f4fb700ef54: Download complete
037dlacdde55: Download complete
[...]
```

Figure 11 – Processus de création des conteneurs

CONTAINER ID	IMAGE	NAMES	COMMAND	CREATED	STATUS	PORTS
258dc258f466	zabbix/zabbix-web-nginx-mysql:ubuntu-6.4-latest	"docker-entrypoint.sh"	About a minute ago	Up About a minute (health: starting)	8443/tcp	
p, 0.0.0.0:80->8080/tcp, [::]:80->8080/tcp	zabbix-web					
349a722bbf2b	zabbix/zabbix-server-mysql:ubuntu-6.4-latest	" /usr/bin/tini -- /u..."	About a minute ago	Up About a minute	0.0.0.0:	
:10051->10051/tcp, [::]:10051->10051/tcp	zabbix-server					
c99a18ff9b34	mysql:8.0	" docker-entrypoint.s..."	About a minute ago	Restarting (1) 3 seconds ago		
	zabbix-db					

Figure 12 – Vérification de l'état des services

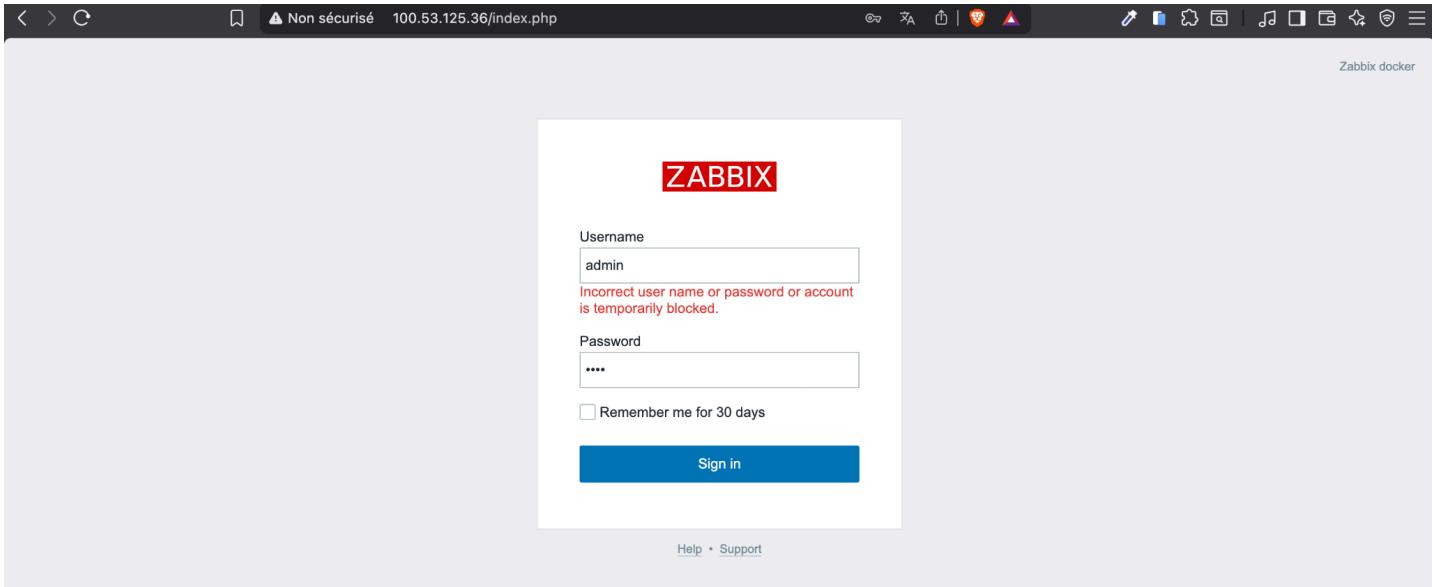


Figure 13 – Page login de Zabbix server

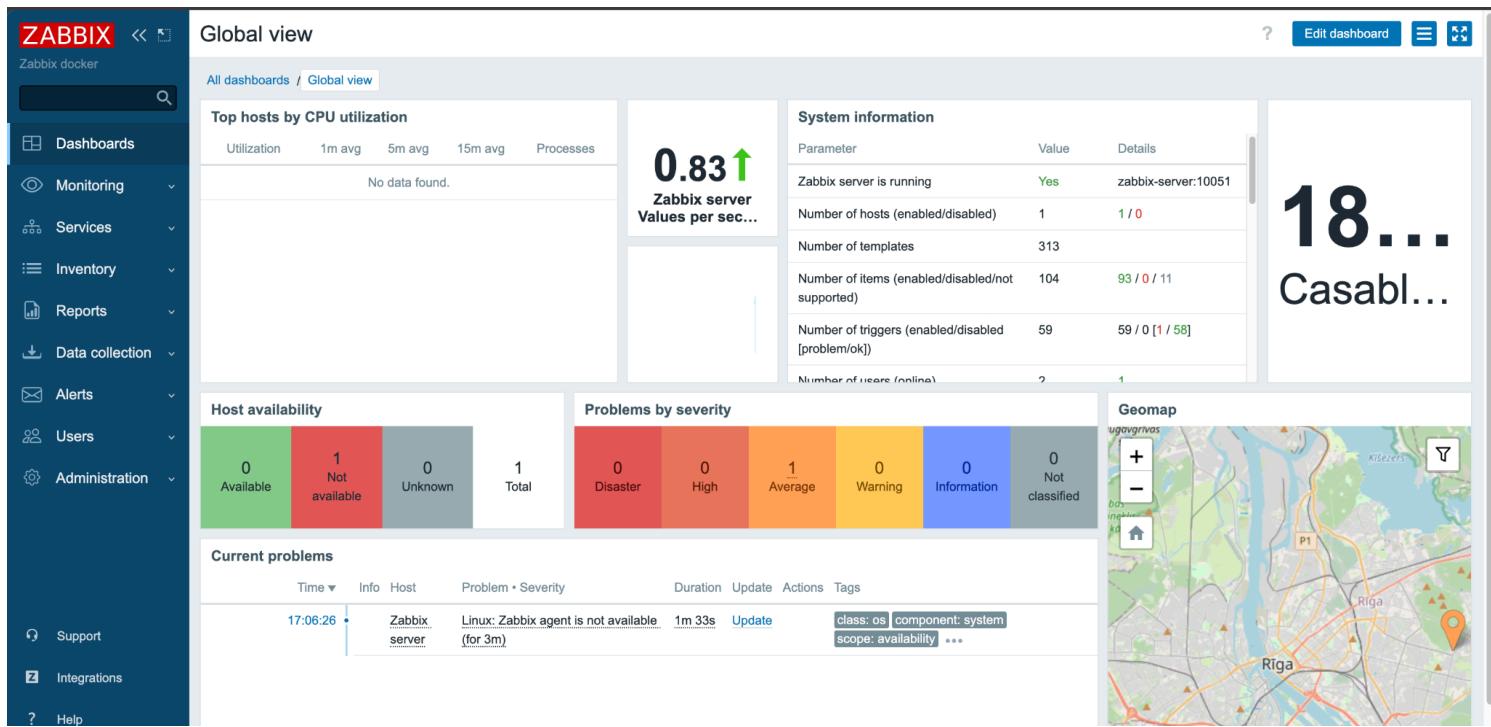


Figure 14 – Interface et tableau de bord de Zabbix

6 Configuration des agents de supervision

Après le déploiement de l'infrastructure centrale, nous procéderons à la configuration de l'agent sur l'instance cible nommée **Linux-client-zabbix**. L'agent Zabbix est responsable de la collecte des métriques locales et de leur transmission au serveur.

6.1 Installation de l'agent sur Linux-client

L'installation nécessite l'ajout du dépôt officiel Zabbix pour accéder à la version 6.4. Nous exécutons les commandes suivantes sur l'instance cliente :

```
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
```

```
ubuntu@ip-172-31-21-195:~$ wget https://repo.zabbix.com/zabbix/6.4/ubuntu/pool/main/z/zabbix-release/zabbix-release_6.4-1+ubuntu22.04_all.deb  
--2026-01-23 17:20:19-- https://repo.zabbix.com/zabbix/6.4/ubuntu/pool/main/z/zabbix-release/zabbix-release_6.4-1+ubuntu22.04_all.deb  
Resolving repo.zabbix.com (repo.zabbix.com)... 178.128.6.101, 2604:a880:2:d0::2062:d001  
Connecting to repo.zabbix.com (repo.zabbix.com)|178.128.6.101|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 3744 (3.7K) [application/octet-stream]  
Saving to: 'zabbix-release_6.4-1+ubuntu22.04_all.deb'  
  
zabbix-release_6.4-1+ubuntu22.04_all.deb 100%[=====] 3.66K --.-KB/s in 0s  
  
2026-01-23 17:20:19 (2.05 GB/s) - 'zabbix-release_6.4-1+ubuntu22.04_all.deb' saved [3744/3744]  
  
ubuntu@ip-172-31-21-195:~$ sudo dpkg -i zabbix-release_6.4-1+ubuntu22.04_all.deb  
Selecting previously unselected package zabbix-release.  
(Reading database ... 65993 files and directories currently installed.)  
Preparing to unpack zabbix-release_6.4-1+ubuntu22.04_all.deb ...  
Unpacking zabbix-release (1:6.4-1+ubuntu22.04) ...  
Setting up zabbix-release (1:6.4-1+ubuntu22.04) ...  
ubuntu@ip-172-31-21-195:~$ sudo apt update  
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease  
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]  
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]  
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]  
Get:5 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]  
Get:6 https://repo.zabbix.com/zabbix/6.4/ubuntu jammy InRelease [2883 B]  
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [5652 kB]  
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 c-n-f Metadata [286 kB]  
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [217 kB]  
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse Translation-en [112 kB]  
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 c-n-f Metadata [8372 B]  
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [3165 kB]  
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [485 kB]  
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [19.1 kB]
```

Figure 15 – Installation du dépôt Zabbix sur l'hôte Linux

Ensuite vient l'installation de l'agent zabbix :

```
sudo apt install zabbix-agent -y
```

```
ubuntu@ip-172-31-21-195:~$ sudo apt install zabbix-agent -y  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following additional packages will be installed:  
libmodbus5  
The following NEW packages will be installed:  
libmodbus5 zabbix-agent  
0 upgraded, 2 newly installed, 0 to remove and 58 not upgraded.  
Need to get 289 kB of archives.  
After this operation, 827 kB of additional disk space will be used.  
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 libmodbus5 amd64 3.1.6-2 [23.5 kB]  
Get:2 https://repo.zabbix.com/zabbix/6.4/ubuntu jammy/main amd64 zabbix-agent amd64 1:6.4.21-1+ubuntu22.04 [265 kB]  
Fetched 289 kB in 1s (481 kB/s)  
Selecting previously unselected package libmodbus5:amd64.  
(Reading database ... 65999 files and directories currently installed.)  
Preparing to unpack .../libmodbus5_3.1.6-2_amd64.deb ...  
Unpacking libmodbus5:amd64 (3.1.6-2) ...  
Selecting previously unselected package zabbix-agent.  
Preparing to unpack .../zabbix-agent_183a6.4.21-1+ubuntu22.04_amd64.deb ...  
Unpacking zabbix-agent (1:6.4.21-1+ubuntu22.04) ...  
Setting up libmodbus5:amd64 (3.1.6-2) ...  
Setting up zabbix-agent (1:6.4.21-1+ubuntu22.04) ...  
Created symlink /etc/systemd/system/multi-user.target.wants/zabbix-agent.service → /lib/systemd/system/zabbix-agent.service.  
Processing triggers for man-db (2.10.2-1) ...  
Processing triggers for libc-bin (2.35-0ubuntu3.11) ...  
Scanning processes...  
Scanning linux images...
```

Figure 16 – Finalisation de l'installation de l'agent

6.2 Configuration du service

La communication entre l'agent et le serveur repose sur la modification du fichier de configuration principal situé dans /etc/zabbix/.

```
ubuntu@ip-172-31-21-195:~$ sudo nano /etc/zabbix/zabbix_agentd.conf
ubuntu@ip-172-31-21-195:~$ █
```

Figure 17 – Édition du fichier de configuration

Il faut modifier le fichier de façon à ajouter l'adresse IP du serveur Zabbix afin que son nom afin de l'identifier.

```
Server=100.53.125.36
### Option: ListenPort
#       Agent will listen on this port for connections from the server.
```

```
ServerActive=100.53.125.36
### 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=Zabbix server
```

6.3 Initialisation et persistance

Pour appliquer ces changements, le service est démarré et activé pour se lancer automatiquement au démarrage du système :

```
sudo systemctl restart zabbix - agent
enable zabbix - agent
```

```
ubuntu@ip-172-31-21-195:~$ sudo systemctl restart zabbix-agent
ubuntu@ip-172-31-21-195:~$ sudo systemctl enable zabbix-agent
Synchronizing state of zabbix-agent.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable zabbix-agent
ubuntu@ip-172-31-21-195:~$ █
```

Figure 18 – Redémarrage et vérification du statut du service zabbix-agent

6.4 Enregistrement sur l'interface Web

Maintenant il faut configurer et ajouter les agents au sein de l'interface d'administration Zabbix afin de lier l'agent aux modèles de surveillance.

New host

Host IPMI Tags Macros Inventory Encryption Value mapping

* Host name Client_Linux_Zabbix

Visible name Client_Linux_Zabbix

Templates Linux by Zabbix agent active

* Host groups Linux servers

Interfaces Type IP address DNS name Connect to Port Default

Agent	54.221.144.4		IP DNS	10050	<input checked="" type="radio"/> Remove
-------	--------------	--	--------	-------	-----------------------------------------

Add

Description

Monitored by proxy (no proxy)

Enabled

Add Cancel

Figure 19 – Crédation de l'hôte Linux-client dans le tableau de bord Zabbix

Name	Interface	Availability	Tags	Status	Latest data	Problems	Graphs	Dashboards	Web
Linux_client	54.221.144.4:10050	ZBX	class: os target: linux	Enabled	Latest data 68	Problems	Graphs 14	Dashboards 3	Web

<input type="checkbox"/> Linux_client	Interface ens5: Inbound packets discarded	component: network	interface: ens5	Graph	
<input type="checkbox"/> Linux_client	Interface ens5: Inbound packets with errors	component: network	interface: ens5	Graph	
<input type="checkbox"/> Linux_client	Interface ens5: Interface type <input type="button" value="?"/>	31s	Ethernet (1)	Graph	
<input type="checkbox"/> Linux_client	Interface ens5: Operational status <input type="button" value="?"/>	31s	up (6)	Graph	
<input type="checkbox"/> Linux_client	Interface ens5: Outbound packets discarded	component: network	interface: ens5	Graph	
<input type="checkbox"/> Linux_client	Interface ens5: Outbound packets with errors	component: network	interface: ens5	Graph	
<input type="checkbox"/> Linux_client	Interface ens5: Speed <input type="button" value="?"/>		component: network	Graph <input type="button" value="i"/>	
<input type="checkbox"/> Linux_client	Linux: Active agent availability <input type="button" value="?"/>	8s	available (1) +1	component: health component: network	Graph
<input type="checkbox"/> Linux_client	Linux: Available memory <input type="button" value="?"/>	46s	3.26 GB	component: memory	Graph
<input type="checkbox"/> Linux_client	Linux: Available memory in % <input type="button" value="?"/>	46s	87.0749 %	component: memory	Graph
<input type="checkbox"/> Linux_client	Linux: Checksum of /etc/passwd	46s	f50111678fdb3...	component: security	History
<input type="checkbox"/> Linux_client	Linux: Context switches per second <input type="button" value="?"/>		component: cpu	Graph	
<input type="checkbox"/> Linux_client	Linux: CPU guest nice time <input type="button" value="?"/>	46s	0 %	component: cpu	Graph
<input type="checkbox"/> Linux_client	Linux: CPU guest time <input type="button" value="?"/>	46s	0 %	component: cpu	Graph
<input type="checkbox"/> Linux_client	Linux: CPU idle time <input type="button" value="?"/>	46s	99.9583 %	component: cpu	Graph
<input type="checkbox"/> Linux_client	Linux: CPU interrupt time <input type="button" value="?"/>	46s	0 %	component: cpu	Graph

Figure 20 – Données reçus du client linux sur l'interface de Zabbix

6.5 Installation de l'agent sur Windows

Maintenant, c'est à l'agent sur Windows d'être configuré, d'abord, il faut activer le RDP pour l'accès à distance tout en obtenant le mot de passe pour se connecter, ensuite télécharger le fichier d'installation Zabbix Agent.

Session Manager | **Client RDP** | EC2 Serial Console

ID d'instance
i-09e50213e2bc509df (Client_Windows)

Type de connexion

Connexion à l'aide du client RDP
Téléchargez un fichier à utiliser avec votre client RDP et récupérez votre mot de passe.

Connexion à l'aide du gestionnaire de parc
Pour vous connecter à l'instance à l'aide du Bureau à distance du gestionnaire de parc, l'agent SSM doit être installé et en cours d'exécution sur l'instance. Pour plus d'informations, consultez [Utilisation de l'agent SSM](#)

Vous pouvez également vous connecter à votre instance Windows en utilisant un client Bureau à distance de votre choix et en téléchargeant et en exécutant le fichier de raccourci RDP ci-dessous :

[Télécharger le fichier bureau à distance](#)

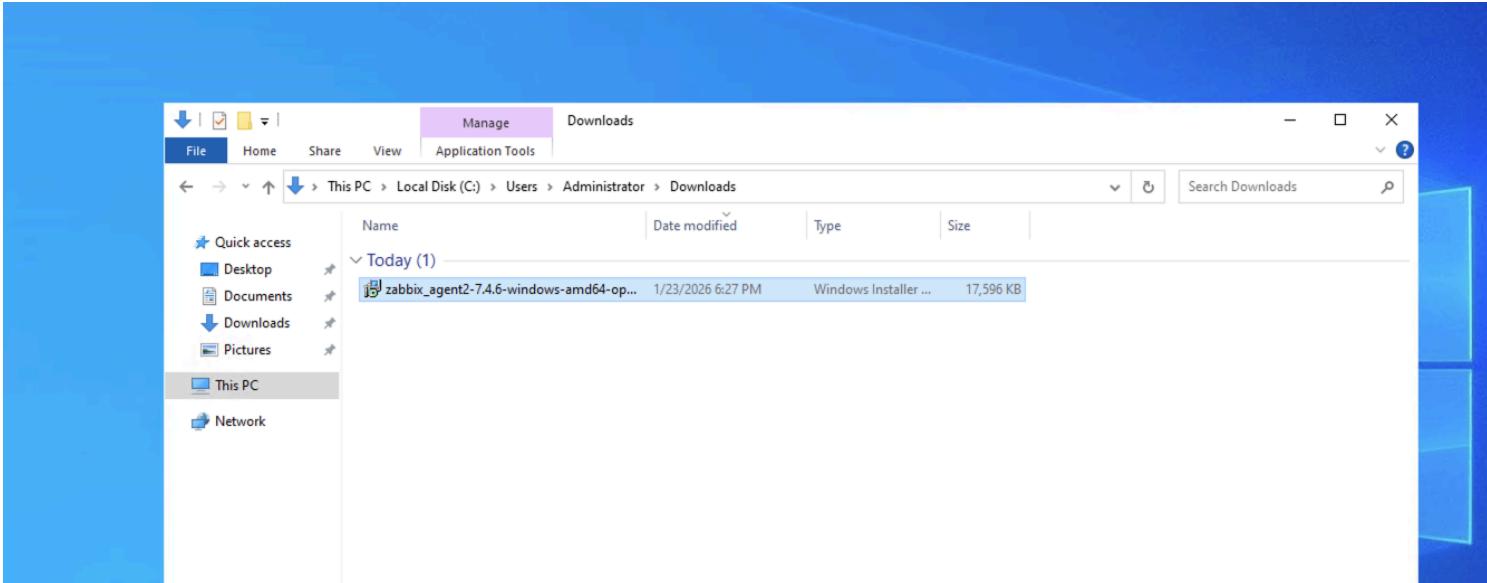
À l'étape correspondante, connectez-vous à votre instance à l'aide du nom d'utilisateur et du mot de passe suivants :

Public DNS
ec2-54-175-85-14.compute-1.amazonaws.com

Nom d'utilisateur [Informations](#)
 Administrator

Mot de passe [Obtenir le mot de passe](#)

i Si vous avez joint votre instance à un répertoire, vous pouvez utiliser vos informations d'identification pour vous connecter à votre instance.



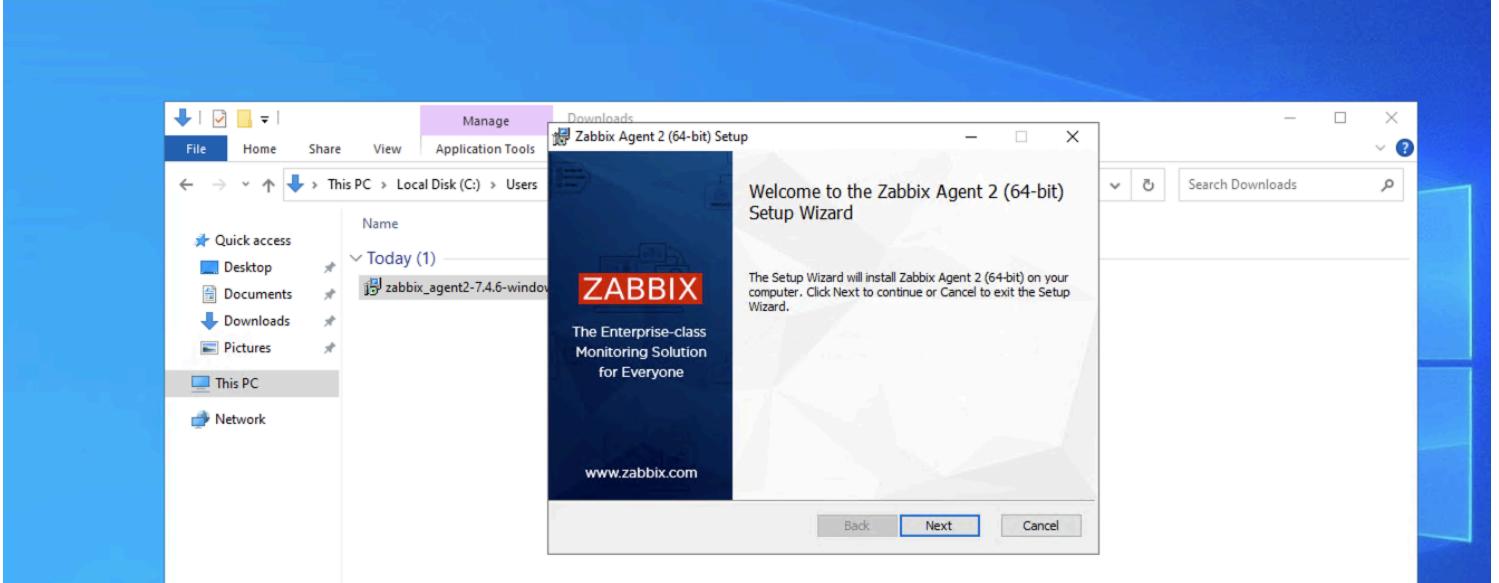


Figure 21 – Processus de configuration et installation de l’agent Windows

6.6 Validation de la Connectivité

Avant de déclarer l’hôte dans la console de gestion, une vérification de la connectivité réseau est effectuée pour confirmer que le flux TCP sur le port 10051 est opérationnel.

```
PS C:\Users\Administrator> Test-NetConnection -ComputerName 172.31.29.12 -Port 10051

ComputerName      : 172.31.29.12
RemoteAddress    : 172.31.29.12
RemotePort       : 10051
InterfaceAlias   : Ethernet 3
SourceAddress    : 172.31.24.164
TcpTestSucceeded : True

PS C:\Users\Administrator>
```

Figure 22 – Test de connectivité réussi via PowerShell (TcpTestSucceeded)

6.7 Enregistrement et Supervision

- **Création de l’hôte** : Le nouveau client est enregistré dans l’interface Web de Zabbix en créant un *hôte* associé au nom d’hôte configuré précédemment.

The screenshot shows the 'New host' configuration dialog in Zabbix. The 'Host' tab is selected. Key fields include:

- Host name:** EC2AMAZ-CK57OCP
- Visible name:** EC2AMAZ-CK57OCP
- Templates:** Windows by Zabbix agent active
- Host groups:** Virtual machines
- Interfaces:** Agent 54.175.85.14 (IP selected, Port 10050, Default checked)
- Description:** (empty)
- Monitored by proxy:** (no proxy)
- Enabled:** checked

Figure 23 – Configuration du nouvel hôte Windows dans l’interface Zabbix

- **Vérification de la réception :** La confirmation finale est obtenue via l’onglet *Latest Data*, attestant du flux entrant des métriques de performance.

Name	Interface	Availability	Tags	Status	Latest data	Problems	Graphs	Dashboards	Web
EC2AMAZ-CK57OCP	54.175.85.14:10050	ZBX	class: os target: windows	Enabled	Latest data 48	Problems	Graphs 8	Dashboards 3	Web
Linux_client	54.221.144.4:10050	ZBX	class: os target: linux	Enabled	Latest data 68	Problems	Graphs 14	Dashboards 3	Web

Latest data						
EC2AMAZ-CK57OCP	Windows: Number of cores	17s	2	component: cpu	Graph	
EC2AMAZ-CK57OCP	Windows: Number of processes	16s	112	component: os	Graph	
EC2AMAZ-CK57OCP	Windows: Number of threads			component: os	Graph	
EC2AMAZ-CK57OCP	Windows: Operating system			component: os	History	
EC2AMAZ-CK57OCP	Windows: Operating system architecture			component: os	History	
EC2AMAZ-CK57OCP	Windows: System description			component: system	History	
EC2AMAZ-CK57OCP	Windows: System local time	11s	2026-01-23 19:...	component: system	Graph	
EC2AMAZ-CK57OCP	Windows: System name			component: system	History	
EC2AMAZ-CK57OCP	Windows: Total memory	9s	7.9 GB	component: memory	Graph	
EC2AMAZ-CK57OCP	Windows: Total swap space	8s	1.88 GB	component: memory component: storage	Graph	
EC2AMAZ-CK57OCP	Windows: Uptime	7s	02:55:04 +00:00:30	component: system	Graph	component: storage
EC2AMAZ-CK57OCP	Windows: Used memory	6s	2.99 GB	component: memory	Graph	
EC2AMAZ-CK57OCP	Windows: Used swap space in %			component: memory component: storage	Graph	
EC2AMAZ-CK57OCP	Windows: Version of Zabbix agent running			component: application	History	
EC2AMAZ-CK57OCP	Windows: Zabbix agent ping	3s	Up (1)	component: system	Graph	

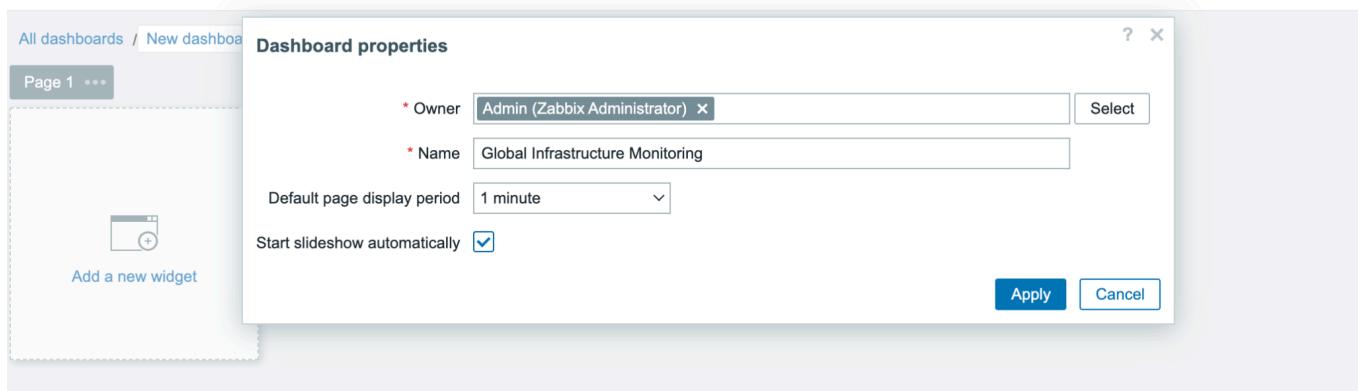
Figure 24 – Visualisation des données reçues par windows

7 Visualisation finale

7.1 Crédation du Dashboard Global

Pour centraliser la surveillance, on a mis en place un tableau de bord global intitulé “**Global Infrastructure Monitoring**” . Il permet de visualiser les métriques critiques de l’ensemble du parc informatique sur une interface unique.

New dashboard



— Configuration du Graphique CPU :

- Ajout d'un widget de type *Graph*.
- Utilisation du motif d'item (*Item pattern*) : CPU utilization.
- Sélection simultanée des hôtes *Linux-client* et *Windows-client* dans les *Host patterns*.

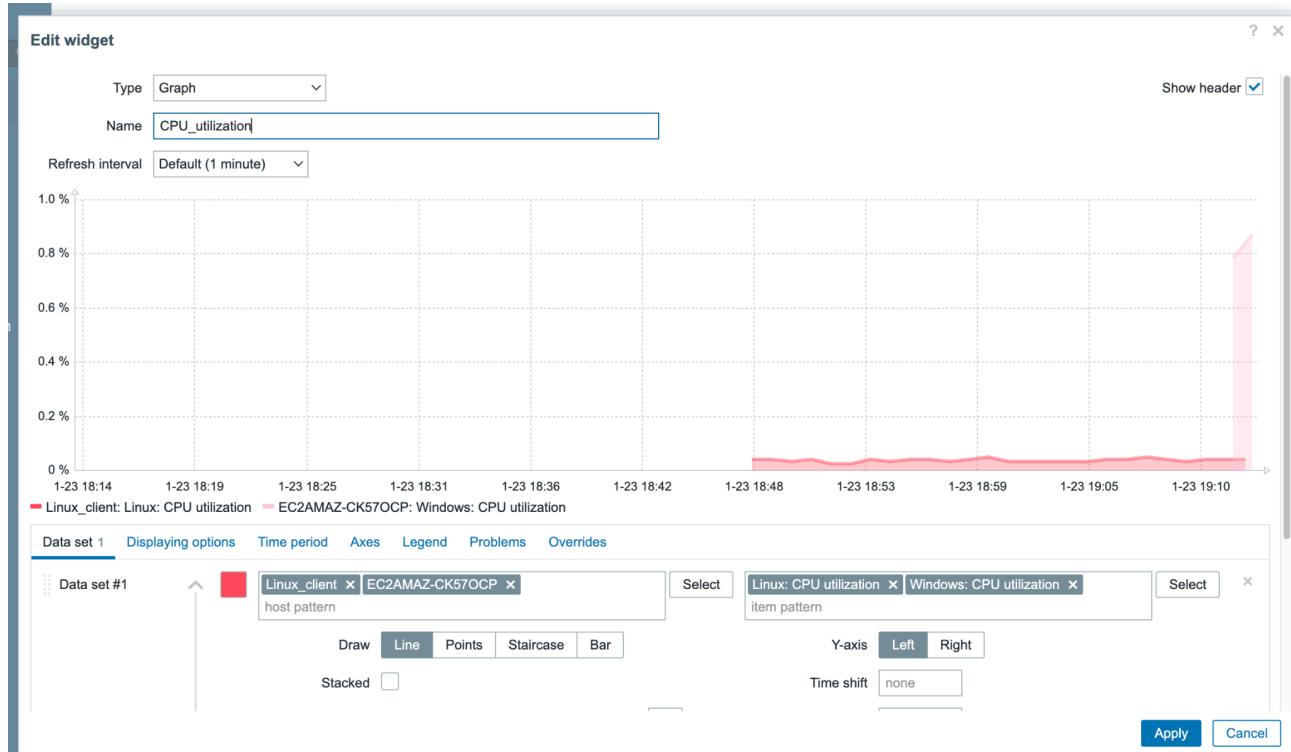


Figure 25 – Configuration du widget de monitoring CPU

— Configuration du Graphique RAM :

- Ajout d'un second widget de type *Graph*.
- Sélection de l'item spécifique : Utilisation mémoire.

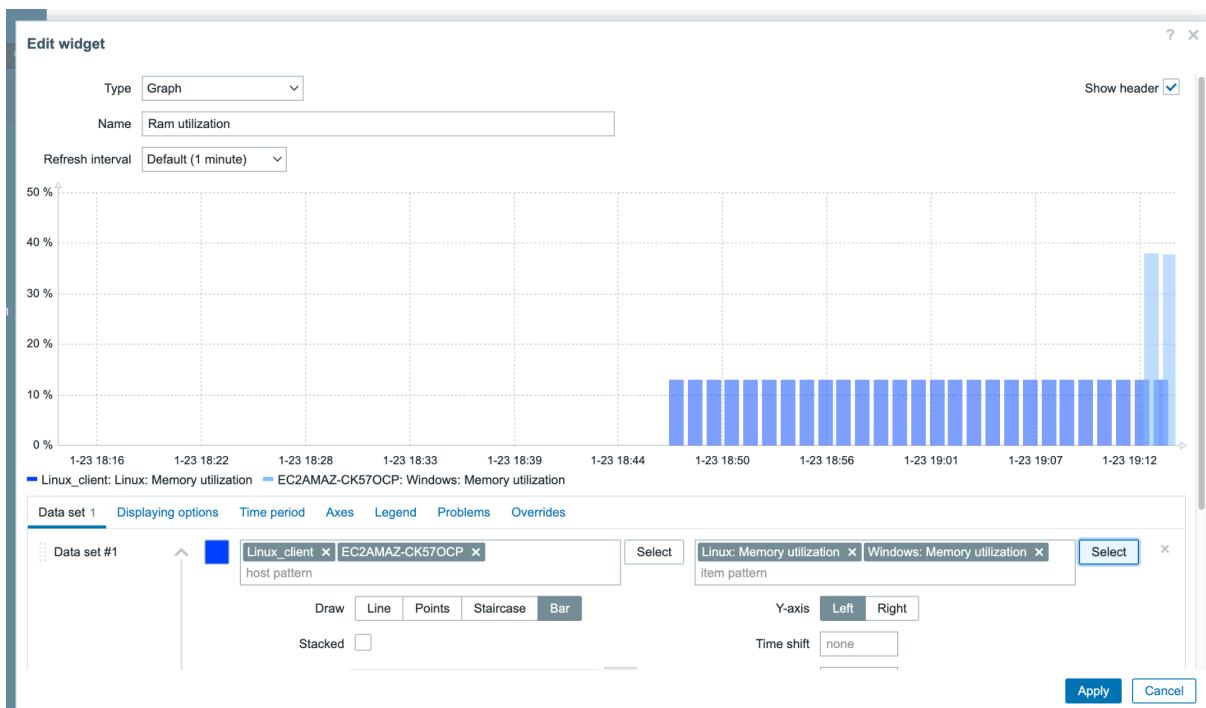


Figure 26 – Mise en place du suivi de l'utilisation de la mémoire vive (RAM)

7.2 Mise en place d'un Trigger (Alerte Proactive)

Afin de démontrer les capacités d'auto-surveillance de la solution, nous avons configuré une alerte proactive simulant une charge CPU élevée.

- **Procédure de création :** Accès via *Data Collection > Hosts*, puis sélection de l'hôte cible (*Linux-client*).

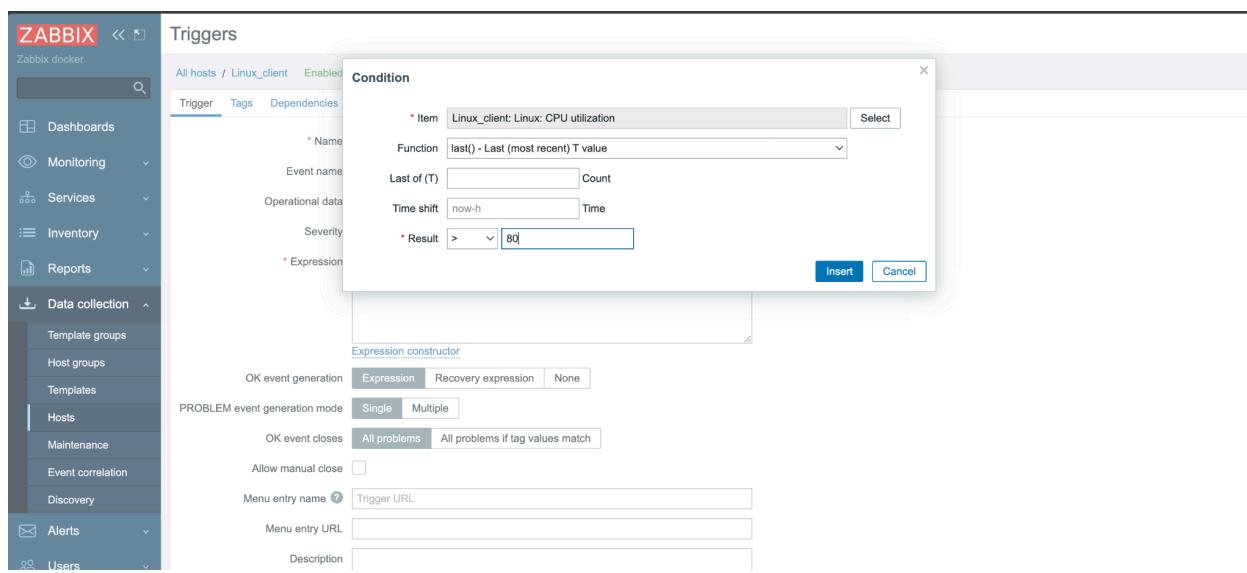
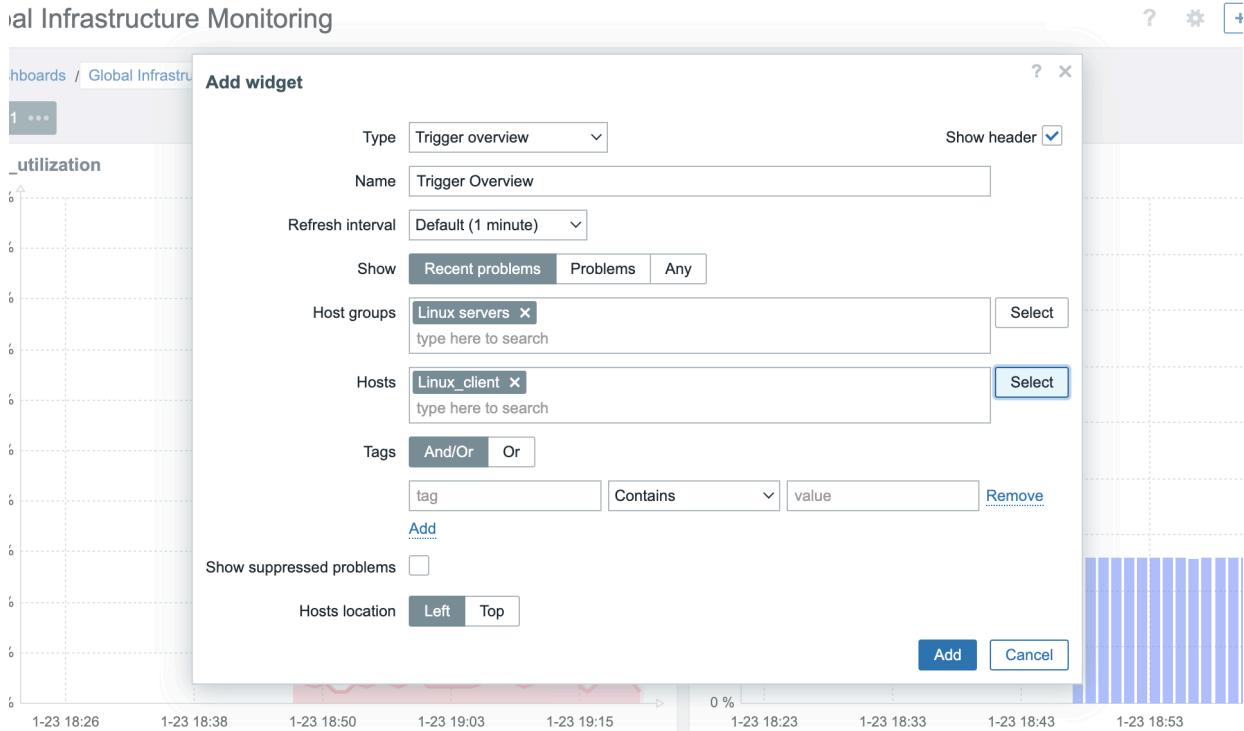


Figure 27 – Crédit d'un nouveau déclencheur



7.3 Synthèse Visuelle (Dashboard Consolidé)

La figure ci-dessous présente le résultat final : un tableau de bord fusionné offrant une visibilité complète et corrélée sur l'état de santé des infrastructures Linux et Windows.



Figure 28 – Aperçu du Dashboard Global regroupant l'ensemble des métriques et alertes

8 Conclusion

Le déploiement de cette infrastructure de monitoring hybride sur AWS illustre pleinement l'interopérabilité entre les technologies Cloud, la conteneurisation et la gestion de systèmes multiplateformes.

L'objectif principal, à savoir centraliser la supervision d'un parc hétérogène composé d'environnements Linux et Windows dans un réseau sécurisé, a été entièrement rempli. L'intégration de Zabbix via Docker a permis d'assurer une stricte séparation des services (serveur, base de données, interface web) tout en offrant une portabilité optimale à la solution.