Mémoire – Projet de fin d'études

Formation : Administrateur Systèmes et Réseaux

& Base De Données

Stagiaire : BOUALI Yassine

Tuteur : DRUBBELS Alexandre

Table des matières :

Table des matières

Table des matières :
Remerciement
Introduction
I. Présentation entreprise
A. Organigramme de l'entreprise
B. Partenaire client NevaOne
C. Partenaire fournisseur NevaOne
D. Description des différents services à NevaOne
E. Schéma reseaux
Projet1 :
II. Gestions de projet
A. Description de la situation initiale
B. Le contexte et la problématique
C. Analyse fonctionnelle des besoins
d. Positionnement (responsabilité confiée, autonomie, part depilotage)
E. Objectifs fixés
F. Contraintes
G. Moyen et ressources
H. Comparaisons des offres
I. Description des tâches
III. Projet Grafana
A. Mise en place du serveur Grafana Prometheus
B. Mise en place des exporters
Authentification anonyme :
C. Installations serveur SMTP
D. Script
Script PowerShell
Script python
E. Fonctionnalités Grafana
F. Dashboard
IV. Problèmes rencontrés
A. Permissions certificat SSL
B. Authentifications anonyme

C. Valeurs retournés .....
D. Seuil CPU personnalisés .....
Projet2 : .....
I. Gestions de projet .....
A. Description de la situation initiale .....
B. Le contexte et la problématique .....
C. L'analyse fonctionnelle des besoins .....
D. Positionnement (responsabilité confiée, autonomie, part de pilotage) .....
E. Objectifs fixés .....
F. Contraintes .....
G. Moyen et ressources .....
H. Comparaisons des offres .....
I. Description des tâches .....
II. Projet Vaultwarden .....
A. Installations Docker .....
B. Installations Nginx .....
C. Découvertes de Vaultwarden .....
Annexe photos entreprise .....
Annexe mots clés, liens et définitions .....
Conclusion .....

Remerciement

Je tiens à exprimer ma sincère gratitude à toute l'équipe de NevaOne pour m'avoir offert une expérience de stage enrichissante, et pour avoir créé une atmosphère de travail agréable où je me suis senti à l'aise.

Je souhaite remercier tout particulièrement mon tuteur de stage, Alexandre DRUBBELS, le gérant de l'entreprise, ainsi que son co-directeur, M. Julien GEORGET pour m'avoir accompagné, soutenu et conseillé tout au long de mon stage. Le sujet de stage qu'ils m'ont confié s'est révélé être extrêmement polyvalent, enrichissant mon expérience professionnelle.

Je souhaite également exprimer ma gratitude envers l'équipe pédagogique du CESI, ainsi que les intervenants professionnels qui ont dispensé les cours théoriques et pratiques durant ma formation en Administration Système et Réseaux et Base de Données.

Enfin, je tiens à remercier chaleureusement tout le personnel de NevaOne pour leur bienveillance et leur accueil durant cette période

Introduction

J'ai eu la chance de faire ce stage chez NevaOne, basée à Goxwiller, du 4 mars au 19 juin 2024. NevaOne, une entreprise spécialisée dans les services numériques, se distingue par sa dynamique et son expertise en optimisation, évolution et gestion de l'infrastructure informatique pour ses clients. Son offre de services est très variée, couvrant tous les besoins allant de la sécurité informatique à l'hébergement en passant par le support aux utilisateurs et la surveillance des systèmes informatiques.

J'ai principalement eu pour mission chez NevaOne de mettre en place et d'améliorer les systèmes de monitoring ainsi que de gérer la sécurité des mots de passe des infrastructures clients. Il était essentiel d'assurer une surveillance efficace des infrastructures clients et de garantir une gestion sécurisée des mots de passe afin de maintenir la continuité et la qualité des services fournis.
Pendant ce stage, j'ai eu l'opportunité de me familiariser avec divers projets incluant le déploiement des technologies Grafana et Prometheus pour la surveillance, ainsi que la mise en place de Vaultwarden pour la gestion des mots de passe. Ces projets ont couvert un large éventail d'aspects tels que la surveillance des serveurs Linux et Windows, la gestion des équipements FortiGate, des connexions internet et la sécurisation des mots de passe.

Pour le projet de monitoring, j'ai travaillé sur l'implémentation de Grafana et Prometheus pour surveiller en temps réel les performances et la disponibilité des infrastructures clients. C'est une surveillance pour nos serveurs internes et du Datacenter. Cela comprenait la configuration des alertes, la visualisation des métriques et l'intégration avec les différents systèmes utilisés par les clients de NevaOne.

Concernant le projet Vaultwarden, mon rôle a été de déployer une solution de gestion des mots de passe sécurisée et centralisée. Cela impliquait l'installation et la configuration de Vaultwarden sur un serveur Docker, la création de comptes utilisateurs, l'organisation des mots de passe par client et infrastructure, et la mise en place de partages sécurisés. J'ai également documenté les procédures et formé les utilisateurs internes pour garantir une adoption efficace de la nouvelle solution.

Je vais expliquer dans ce rapport le projet de monitoring avec Grafana et Prometheus ainsi que le projet Vaultwarden, en détaillant les mesures que j'ai prises pour améliorer le système de surveillance et de gestion des mots de passe chez NevaOne. Travailler chez NevaOne a été une expérience très positive qui m'a permis de concrétiser mes connaissances théoriques et d'acquérir de nouvelles compétences professionnelles.

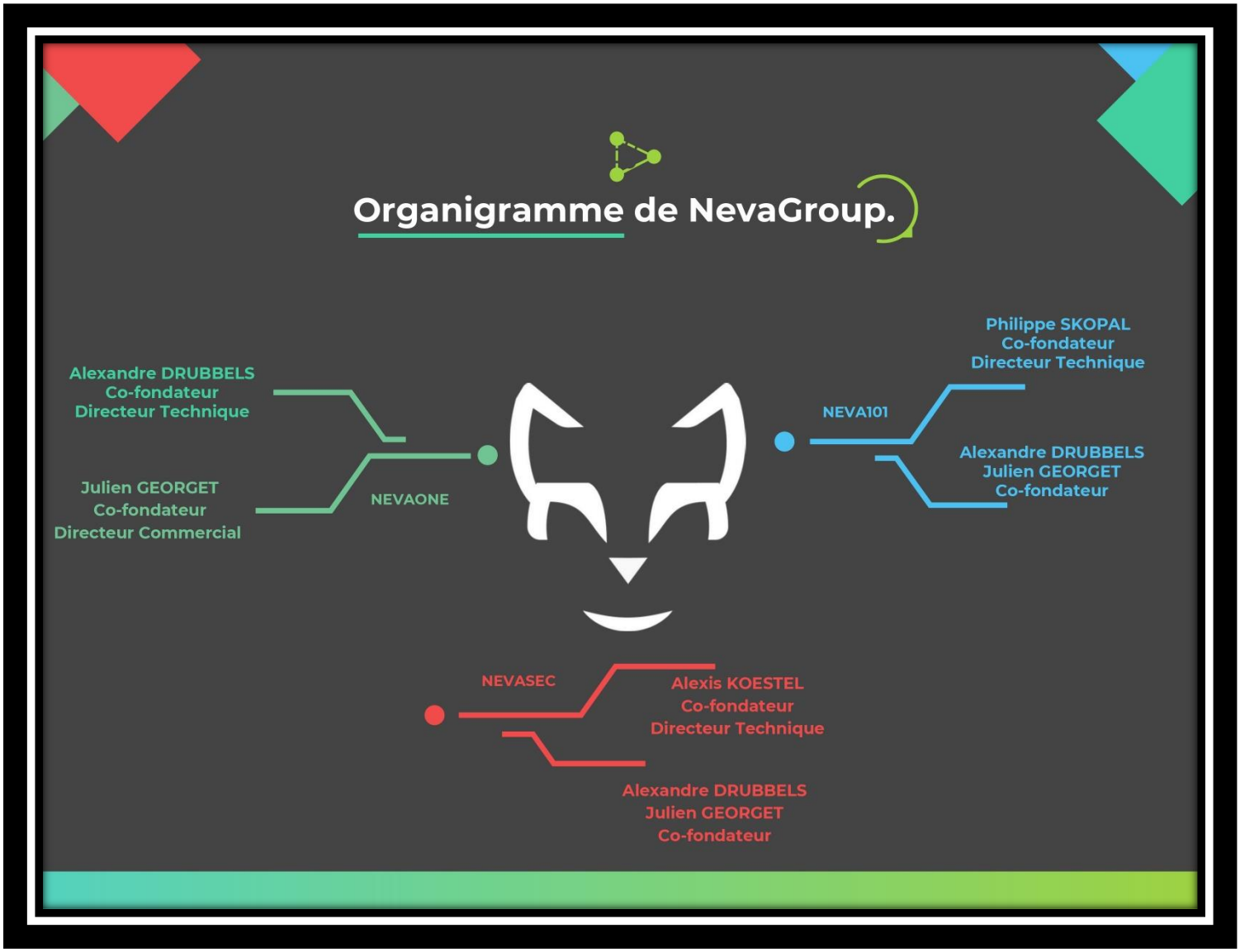
I. Présentation entreprise

NevaOne est une entreprise de services du numérique créée le 19 avril 2021, spécialisée dans l'accompagnement des professionnels dans le monde informatique d'entreprise. Elle propose une gamme étendue de services incluant :

Matériel : Ordinateurs fixes, portables, stations de travail, serveurs, réseaux, téléphonie, routeurs, switchs, WIFI.
Téléphonie et connexion internet : Offres de fibre optique professionnelle (FTTH, FTTO), standard téléphonique, solutions de meeting via téléphone.
Sécurité et cybersécurité : Filtrage mail, antivirus Nextgen, Endpoint Detection and Response (EDR), firewall, filtrage web, accès VPN sécurisé.
Cloud et sauvegarde : Sauvegarde externalisée, plan de reprise d'activité, stockage cloud hybride, hébergement web, serveurs dédiés et virtuels.
Support et assistance : Intervention sur site, contrat d'assistance, hotline.

Avec un savoir-faire 100% Alsacien, NevaOne se distingue par sa volonté de prôner le local et d'offrir une expérience unique à ses clients. L'entreprise doit continuellement s'adapter aux nouvelles technologies et aux évolutions du marché pour fournir les solutions les plus avancées. Elle met un point d'honneur à offrir des services de qualité, fiables et sécurisés, et à proposer des solutions personnalisées adaptées aux besoins spécifiques de chaque client.

A. Organigramme de l'entreprise



B. Partenaire client NevaOne

TOP IMMO

Rocket Tower

Linxya

Brasserie le TIGRE

Accès culture

C. Partenaire fournisseur NevaOne

D. Description des différents services à NevaOne

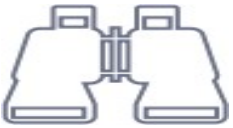
NevaOne Cloud : Hébergement
Des services d'hébergement hyper fiables, avec du PaaS, IaaS et DaaS selon les besoins.

NevaOne Unity : Opérateur & Collaboration
Des solutions opérateurs et des outils collaboratifs, dispo dans le cloud ou sur site, parfaits pour le flex office et le télétravail.

NevaOne Support : Régie / Helpdesk
Une offre complète et qualitative de services demaintenance et de support, qui vous permet de vous concentrer sereinement sur la création de valeur au sein de votre organisation.

NevaOne Infogérance : Service Desk
Compétence, qualité et transparence : l'offre Infogérance adaptative Serenity Prime vouspropose, par le biais de son Service Desk, un panel de services complet répondant de manière précise et en temps réel aux besoins de votre entreprise, quelle que soit votre approche (interne, externe ou hybride).

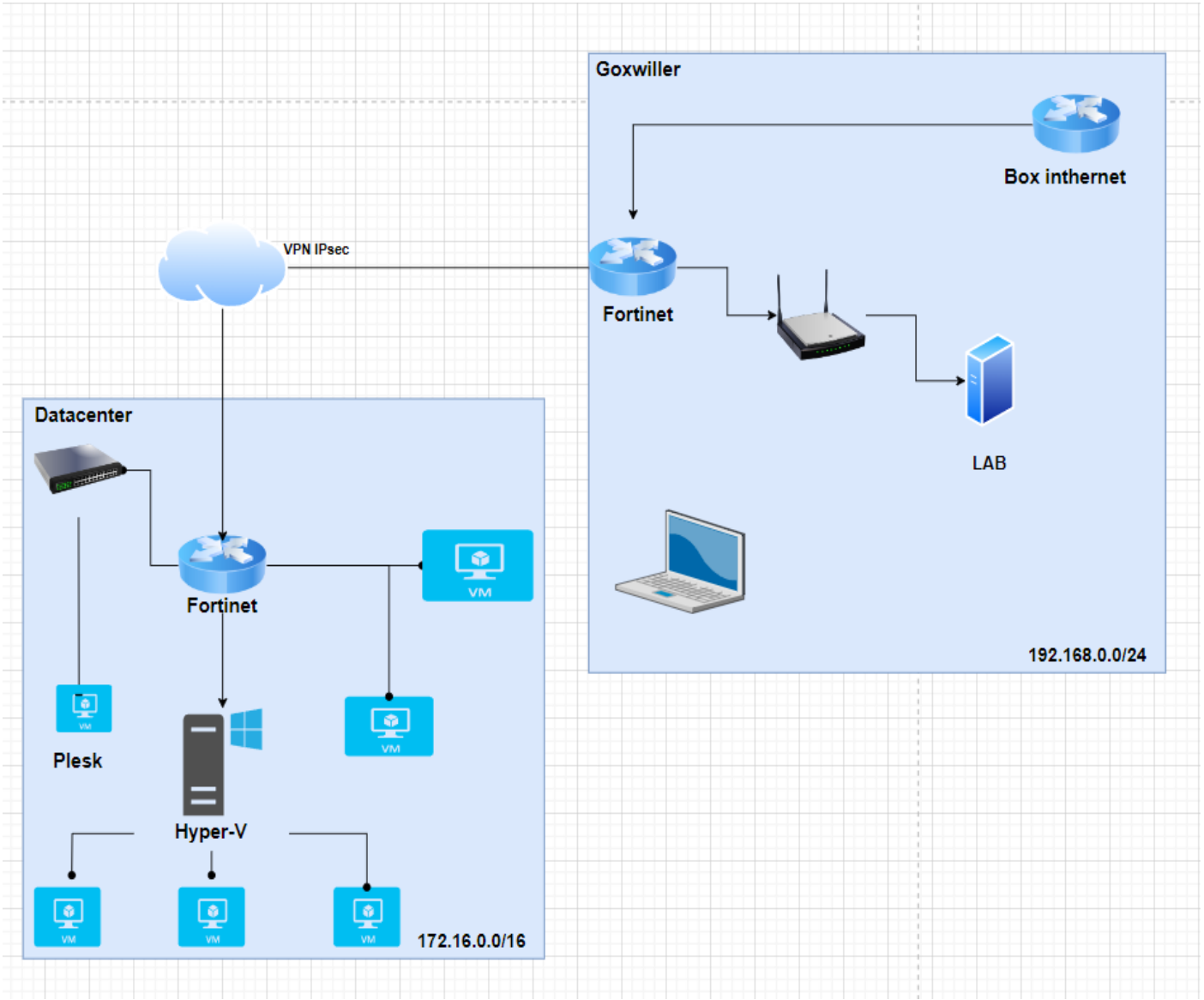
NevaOne Safety : Cybersécurité
Offre A Safety accompagne selon un périmètre complet pour répondre aux enjeux de votre cybersécurité : diagnostic et audits, conseil stratégique ciblé, SOC et suivi opérationnel de votre sécurité.



NevaOne View : Supervision du SI
Offre outillée pensée pour mieux détecter les anomalies et les dysfonctionnements des équipements de votre Système d'Information. Chaque risque potentiel remonte de manière automatique en temps réel et vous permet d'agir, selon une démarche proactive.

NevaOne Managed Services
Offre de services managés complète et de qualité, confiée à nos experts, pour vous permettre de vous consacrer à 100% à votre cœur de métier ! Sur mesure, évolutive et adaptée à vos besoins.

E. Schéma réseaux entreprise



Projet1 :

II. Gestions de projet

A. Description de la situation initiale

NevaOne est une entreprise créée le 19 avril 2021, qui accompagne les professionnels dans le domaine informatique. Elle propose divers services tels que l'installation d'infrastructure informatique, des solutions télécoms (connexion internet, téléphonie fixe et mobile), la sécurité informatique, l'audit et le conseil, ainsi que la création de sites web.
L'entreprise dispose déjà d'un système de monitoring basé sur Grafana et Prometheus. Toutefois, ce système nécessite une extension pour inclure différents types d'équipements supplémentaires.
Actuellement, l'accent est mis sur la disponibilité, la performance et la sécurité des équipements, ainsi que sur la satisfaction des clients en leur fournissant un accès facile aux informations de monitoring

B. Le contexte et la problématique

Le projet souhaite améliorer le système de surveillance en y ajoutant davantage d'éléments, comme des ordinateurs Linux, des serveurs FortiGate et des connexions Internet. Il est important de permettre aux clients d'accéder facilement aux données de surveillance en leur fournissant un lien unique vers lequel accéder.

C. Analyse fonctionnelle des besoins

Surveillance des systèmes Linux et Windows Server : Collecte des métriques de performance, d'utilisation des ressources, et des alertes sur les anomalies.

Surveillance des équipements FortiGate : Surveillance de la santé des dispositifs, du trafic réseau, et des événements de sécurité.

Surveillance des connexions internet : Suivi de la disponibilité, de la latence, et du débit de connexion.

Accès client centralisé : Mise en place d'un portail ou d'un lien unique pour que les clients puissent visualiser les données de monitoring de leurs équipements.

D. Positionnement (responsabilité confiée, autonomie, part de pilotage)

Cette matrice R.A.C.I à 2 acteurs principaux : - Les stagiaires composée de Moi et Jules nous étudions et mettons on place le projet monitoring - Mon tuteur au sein de NevaOne m'apportera une aide et approuvera la réalisation des tâches.

R (réalisateur) : personne qui réalise la tâche
A (approbateur) : personne qui supervise et approuve la réalisation de la tâche.
C (consulté) : personne qui conseille en fonction de son domaine de compétence.
I (informé) : personne qui est informé des progrès réalisés

Tâches à Réaliser	Sous-tâches	Stagiaires	Tuteurs
1. Préparation de l'environnement		R	I, A, C
	Analyse des besoins	R	I, A, C
	Préparation de l'infrastructure	R	C
2. Installation des logiciels		R	I, A, C
	Téléchargement des logiciels	R	I
	Installation et configuration	R	A, C
3. Configuration de Prometheus		R	I, A, C
	Installation de Prometheus	R	A
	Configuration des alertes	R	I, C
4. Configuration de Grafana		R	I, A, C
	Installation de Grafana	R	I
	Création des dashboards	R	A, C
5. Développement et déploiement des scripts		R	I, A, C
	Écriture des scripts	R	A
	Test des scripts	R	I, C
6. Sécurité et accès		R	I, A, C
	Configuration des accès utilisateurs	R	I

	Mise en place des règles de sécurité	R	A, C
7. Documentation et formation		R	I, A, C
	Rédaction de la documentation	R	I
	Formation des utilisateurs	R	A, C
8. Tests et validation		R	I, A, C
	Planification des tests	R	I
	Exécution des tests	R	A, C
9. Mise en production		R	I, A, C
	Préparation des machines de production		R
	Déploiement des applications	R	I, A
10. Suivi et amélioration continue		R	I, A, C
	Suivi des performances	R	I
	Amélioration des processus	R	A, C

E. Objectifs fixés

Mise en place d'un système : Une surveillance plus fine et en temps réel des infrastructures informatiques, permettant une réaction rapide en cas de problème. A l'aide de tableaux de bord intuitifs et personnalisés pour chaque client, facilitant la compréhension et l'analyse des métriques.

Mise en place de seuils d'alerte : L'objectif est de définir des seuils d'alerte appropriés. Il s'agit d'établir des niveaux de tolérance et de définir des seuils critiques, afin d'identifier rapidement les situations anormales et de prendre des mesures correctives appropriées.

Tests de surveillance : La réalisation de test complet de surveillance pour valider la fiabilité et l'efficacité des services de supervision. Ces tests permettront de s'assurer que les équipements sont correctement surveillés, que les seuils d'alerte sont pertinents et que les services fonctionnent correctement.

Satisfaction des Clients : Un accès facile et transparent aux données de leur infrastructure renforce la confiance des clients envers les services de NevaOne.

F. Contraintes

Contraintes Techniques :

- **Compatibilité des Outils** : Les outils de monitoring, tels que Prometheus et Grafana, doivent fonctionner avec divers équipements et systèmes d'exploitation. Par exemple, Prometheus doit être capable de collecter des métriques des systèmes Linux, des serveurs Windows, et des dispositifs FortiGate via des exportateurs adaptés comme Node Exporter, Windows Exporter, et SNMP Exporter.
- **Intégration des Données** : Les données collectées à partir de différentes sources doivent être intégrées et présentées de manière cohérente dans Grafana. Cela peut nécessiter la création de tableaux de bord personnalisés et l'ajustement des formats de données pour assurer une visualisation uniforme et précise.
- **Scalabilité** : Le système doit être capable de s'adapter à une augmentation du nombre d'équipements surveillés sans compromettre les performances. Cela implique une architecture de collecte et de stockage des métriques évolutive et flexible.
- **Latence et Fréquence de Collecte** : Il est essentiel de trouver un équilibre entre la précision des informations et la charge sur les systèmes surveillés en ajustant la fréquence de collecte des données. Une collecte trop fréquente peut entraîner une surcharge des ressources, tandis qu'une collecte trop rare peut manquer des événements critiques.
- **URL Publique** : La contrainte initiale était de pouvoir disposer d'une URL publique pour intégrer le portail de monitoring dans notre portail partenaire, facilitant ainsi l'accès et l'utilisation par nos partenaires.

Contraintes de Sécurité :

- **Accès Sécurisé** : L'accès au portail de monitoring doit être strictement sécurisé pour empêcher les accès non autorisés. Cela inclut l'utilisation de l'authentification multi-facteurs (MFA), le chiffrement des données en transit et au repos, et des contrôles d'accès basés sur les rôles (RBAC).
- **Protection des Données** : Les données de surveillance doivent être protégées contre les fuites et les attaques. Des mesures comme le chiffrement des données sensibles, la surveillance des accès, et la mise en place de pare-feu et de systèmes de détection d'intrusion (IDS) sont essentielles.
- **Gestion des Incidents** : Des procédures doivent être en place pour répondre rapidement aux incidents de sécurité, comme la détection d'une violation ou une tentative d'accès non autorisé. Cela inclut des plans de réponse aux incidents et des équipes prêtes à intervenir.
- **Conformité** : Le système de monitoring doit respecter les réglementations et les standards de sécurité applicables, comme le RGPD pour la protection des données personnelles, ou d'autres normes spécifiques à certains secteurs.

G. Moyen et ressources

Pour la réalisation du projet, j'ai bénéficié de différents moyens et ressources qui m'ont permisd'accomplir mes missions de manière efficace. Les principaux moyens et ressources mis à ma disposition sont les suivants :
J'ai eu accès à un lab pour effectuer des tests, où j'ai pu créer un serveur Grafana et Prometheus ainsi que d'autres machines virtuelles sous Windows Server et Linux. J'y ai configuré des connexions et collecté des métriques via un script pythons PowerShell, que j'ai ensuite affichées sur un tableau de bord.
J'ai été doté des droits d'accès nécessaires aux systèmes, aux plate formes et aux outils utilisés dans le cadre de la supervision chez NevaOne. Ces droits m'ont permis d'accéder aux différentes interfaces degestion, aux configurations des équipements supervisés et aux fonctionnalités pertinentes pour mon projet.

En plus des droits d'accès, j'ai bénéficié des permissions appropriées pour effectuer les actionsrequises dans le cadre de la supervision. Cela inclut l'autorisation de configurer l'outils de supervision, d'ajuster les seuils d'alerte, de créer des contacts et des plages horaires, ainsi qued'effectuer d'autres tâches liées à la gestion des équipements supervisés.
En utilisant ces moyens et ressources, j'ai pu mettre en œuvre les différentes tâches de mon projetde stage, la configuration des Dashboard, la définition des seuils d'alerte, la création de contacts, l'ajustement des plages horaires, et l'utilisation de scripts PowerShell pour l'automatisation.
Il convient de noter que l'accès aux droits, aux permissions, aux fichiers et aux procédures étaitaccordé dans le respect des règles de sécurité et de confidentialité. J'ai dû respecter les bonnespratiques et les politiques internes pour garantir la sécurité des systèmes et des informations.

H. Comparatif de solutions

Ces solutions offrent une variété d'outils et de fonctionnalités pour la surveillance et la visualisation des données, chacune ayant ses propres avantages en fonction des besoins spécifiques

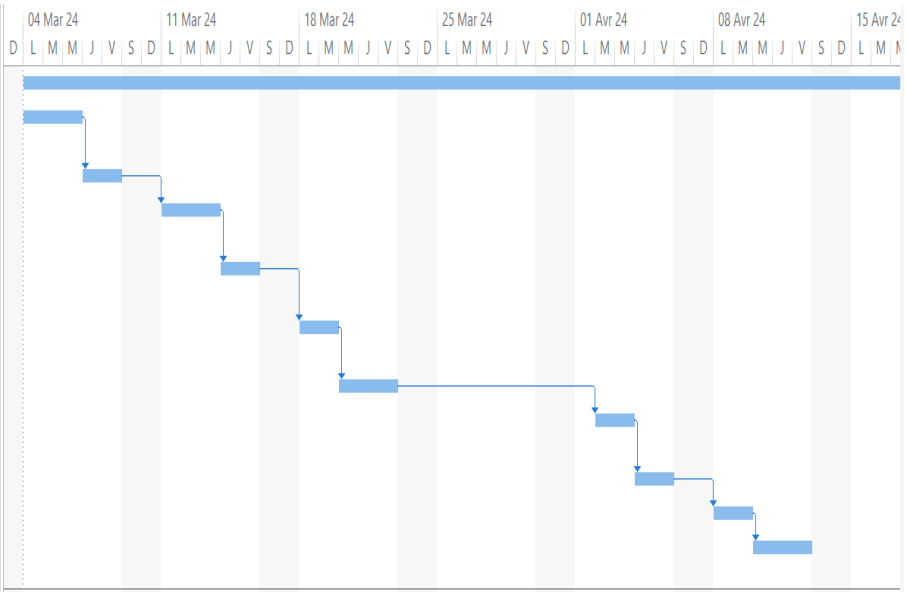
Open Source	Oui	Oui	Oui	Non	Oui	Oui
Facilité d'installation	Moyenne	Moyenne	Moyenne	Facile	Moyenne	Moyenne
Intégrations	Large (via exporters)	Large	Large	Très large	Large	Large
Alerting	Oui (Alertmanager)	Oui	Oui	Oui	Oui (Kapacitor)	Oui (Watchers)
Scalabilité	Haute	Moyenne	Moyenne	Très haute	Haute	Très haute
Support communautaire	Fort	Fort	Fort	Limité (support payant)	Fort	Fort
Tarification	Gratuit	Gratuit	Gratuit	Payant	Gratuit	Gratuit
Langage de requête	PromQL	SQL-like	Nagios-specific	Query Language	Flux	Elasticsearch DSL
Utilisation principale	Monitoring des systèmes et services	Monitoring des infrastructures	Monitoring des systèmes et services	Monitoring des infrastructures	Time-series DB & monitoring	Log management & monitoring
Historique de données	Conservé localement	Conservé localement	Conservé localement	Cloud-based	Conservé localement	Conservé localement ou cloud
Déploiement	Local ou cloud	Local ou cloud	Local ou cloud	Cloud	Local ou cloud	Local ou cloud

I. Description des tâches

Dans le cadre de mon projet de stage, j'ai été chargé de réaliser plusieurs tâches essentielles pourl'optimisation de la supervision chez NevaOne. Les principales tâches effectuées sont les suivantes :

Préparation de l'environnement :
Accéder au lab pour effectuer les tests.
Créer les serveurs nécessaires, incluant un serveur Grafana et un serveur Prometheus.
Configurer les machines virtuelles (VM) supplémentaires, notamment des serveurs Windows et Linux.
Installation des logiciels :
Installer Prometheus sur le serveur dédié.
Installer Grafana sur le serveur approprié.
Assurer que les prérequis sont satisfaits, comme Docker, si nécessaire.
Configuration de Prometheus :
Configurer Prometheus pour collecter les métriques à partir des différentes sources.
Définir les règles de collecte et les cibles de surveillance.
Assurer la connectivité avec les serveurs Windows et Linux pour la collecte des données via SNMP ou WMI.
Configuration de Grafana :
Configurer Grafana pour se connecter à la base de données Prometheus.
Créer et configurer les dashboards pour visualiser les métriques collectées.
Mettre en place des alertes pour des conditions spécifiques de surveillance.
Développement et déploiement des scripts :
Écrire des scripts pour automatiser la collecte des métriques sur les serveurs.
Déployer ces scripts sur les serveurs Windows et Linux.
Tester les scripts pour assurer leur bon fonctionnement et leur intégration avec Prometheus.
Sécurité et accès :
Configurer l'accès sécurisé aux interfaces de Grafana et Prometheus.
Mettre en place des accès restreints pour les utilisateurs finaux, permettant aux clients de visualiser uniquement les données pertinentes.
Documentation et formation :
Documenter toutes les configurations et les étapes de mise en place.
Former les équipes internes et les clients sur l'utilisation des dashboards et des fonctionnalités de surveillance.
Tests et validation :
Effectuer des tests de charge pour vérifier la performance du système.
Valider la collecte correcte des métriques et l'affichage sur les dashboards.
Corriger les éventuelles erreurs ou ajuster les configurations en fonction des résultats des tests.
Mise en production :
Déployer la solution en production après validation des tests.
Superviser la transition et assurer un support initial pour résoudre tout problème qui pourrait survenir.
Suivi et amélioration continue :
Surveiller régulièrement les performances et l'efficacité de la solution mise en place.
Recueillir les feedbacks des utilisateurs pour des améliorations continues.

Nom de la tâche	Durée	Début	Fin	Prédécesseurs
Grafana	78 jours	Lun 04/03/24	Mer 19/06/24	
Préparation de l'environnement:	3 jours	Lun 04/03/24	Mer 06/03/24	
Installation des logiciels	2 jours	Jeu 07/03/24	Ven 08/03/24	2
Configuration de Prometheus	3 jours	Lun 11/03/24	Mer 13/03/24	3
Configuration de Grafana	2 jours	Jeu 14/03/24	Ven 15/03/24	4
Développement et déploiement des scripts	2 jours	Lun 18/03/24	Mar 19/03/24	5
Sécurité et accès	3 jours	Mer 20/03/24	Ven 22/03/24	6
Documentation et formation	2 jours	Mar 02/04/24	Mer 03/04/24	7
Tests et validation	2 jours	Jeu 04/04/24	Ven 05/04/24	8
Mise en production	2 jours	Lun 08/04/24	Mar 09/04/24	9
Suivi et amélioration continue	3 jours	Mer 10/04/24	Ven 12/04/24	10



III. Projet Grafana

A. Mise en place du serveur Grafana Prometheus

Installations se passe sur un environnement Linux
Prometheus :
Étape 1 : Téléchargement de Prometheus
`wget https://github.com/prometheus/prometheus/releases/download/v2.32.0/prometheus-2.32.0.linux-amd64.tar.gz`
Étape 2 : Extraire l'archive
`tar -xvf prometheus-2.32.0.linux-amd64.tar.gz`
Étape 3 : Accéder au répertoire extrait
`cd prometheus-2.32.0.linux-amd64`
Étape 4 : Configuration de Prometheus

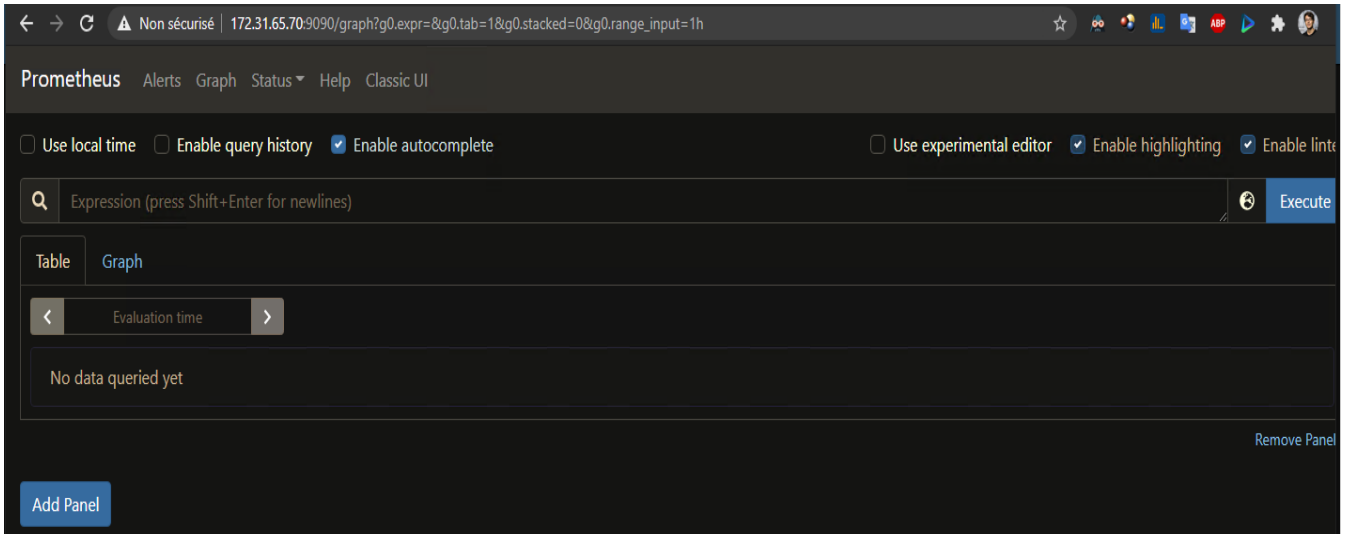
Créez un fichier de configuration prometheus.yml : `nano prometheus.yml`
Ajoute notre configuration.

```
# my global config
global:
  scrape_interval: 5s # Set the scrape interval to every 15 seconds. Default is>
  evaluation_interval: 5s # Evaluate rules every 15 seconds. The default is eve>
  # scrape_timeout is set to the global default (10s).

# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
        - targets:
          # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'evalu>
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
```

```
# The job name is added as a label `job=<job_name>` to any timeseries scraped>
- job_name: "prometheus"
  static_configs:
 - targets: ["localhost:9090"]
```

- **Global Configuration** : Définit les paramètres globaux pour la collecte et l'évaluation des métriques.
- **Alertmanager Configuration** : Configure où envoyer les alertes, bien que cette configuration soit actuellement commentée.
- **Rule Files Configuration** : Charge des fichiers contenant des règles pour les alertes et l'enregistrement des métriques, actuellement non activé.
- **Scrape Configuration** : Définit des jobs de scraping pour collecter des métriques, avec un job pour scraper Prometheus lui-même.

Étape 5 : Démarrer Prometheus
Prometheus est maintenant en cours d'exécution et accessible à l'adresse http://localhost:9090



Grafana :

Étape 1 : Installer les dépendances
```
apt-get install -y gnupg gnupg2 gnupg1 apt-transport-https software-properties-common wget
```

Étape 2 : Télécharger et ajouter la clé GPG de Grafana
```
wget -q -O - https://packages.grafana.com/gpg.key | apt-key add -
```

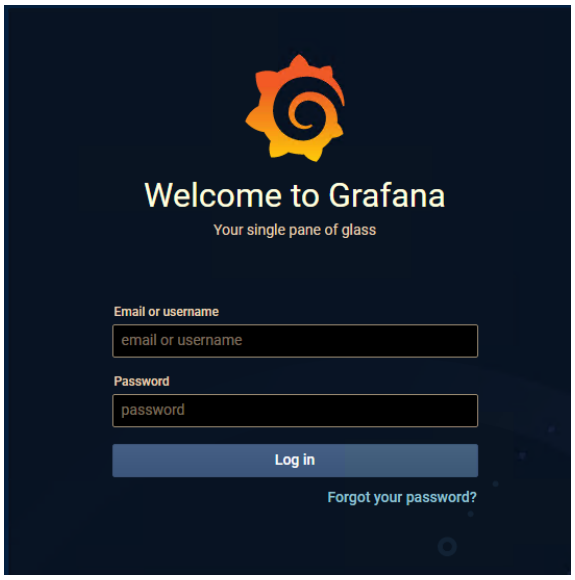
Étape 3 : Ajouter le dépôt Grafana à la liste des sources
```
echo "deb https://packages.grafana.com/oss/deb stable main" | tee -a /etc/apt/sources.list.d/grafana.list
```

Étape 4 : Mettre à jour la liste des paquets disponibles
```
apt-get update
```

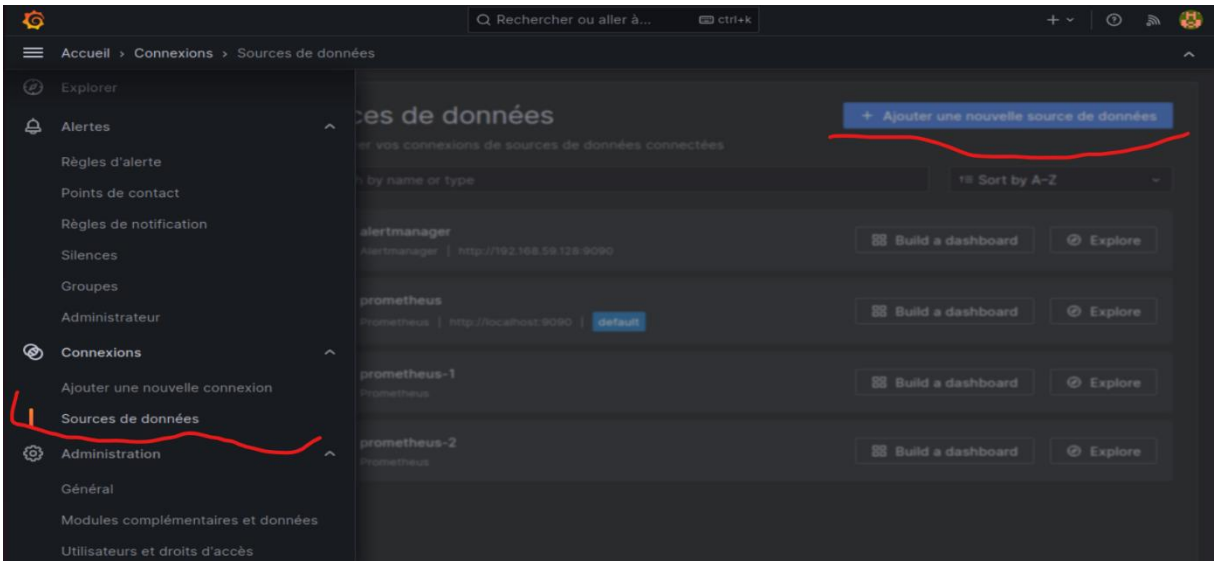
Étape 5: nstaller Grafana
```
apt-get install -y grafana
```

Étape 6 : Démarrer le service Grafana
```
systemctl start grafana-server
```

Grafana est maintenant en cours d'exécution et accessible à l'adresse http://localhost:3000
Identifiants et mot de passe par défaut : **admin**



Nous allons ajouter comme source de données Prometheus
Dans Grafana nous allons dans la playlist Connexions>Source de données
Nous cliquons sur le bouton « **ajouter une nouvelle base de données** »

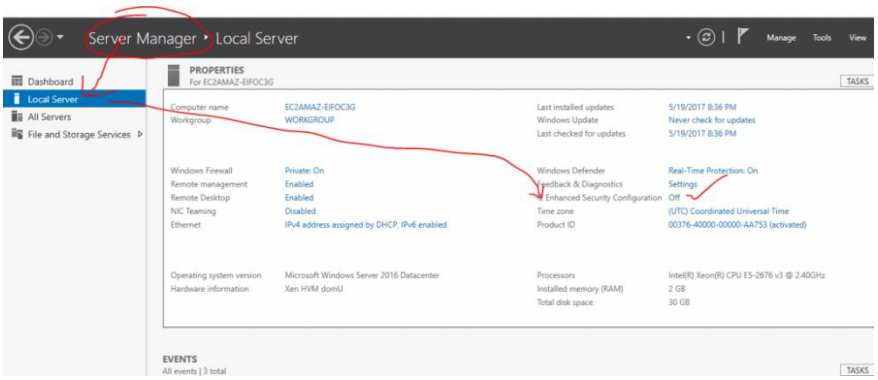


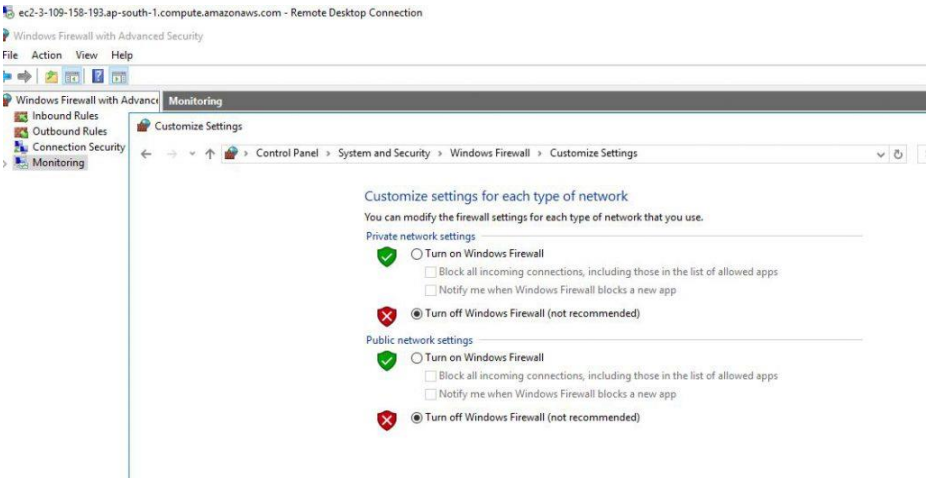
Nous recherchons dans la barre de recherche « Prometheus » puis nous configurons la Connexions
Et nous inscrivons http://localhost:9090 puis on sauvegarde

B. Mise en place des exporters

Windows exporter
Nous allons voir comment reliée des environnements Windows à Prometheus/Grafana

Windows
Prérequis :



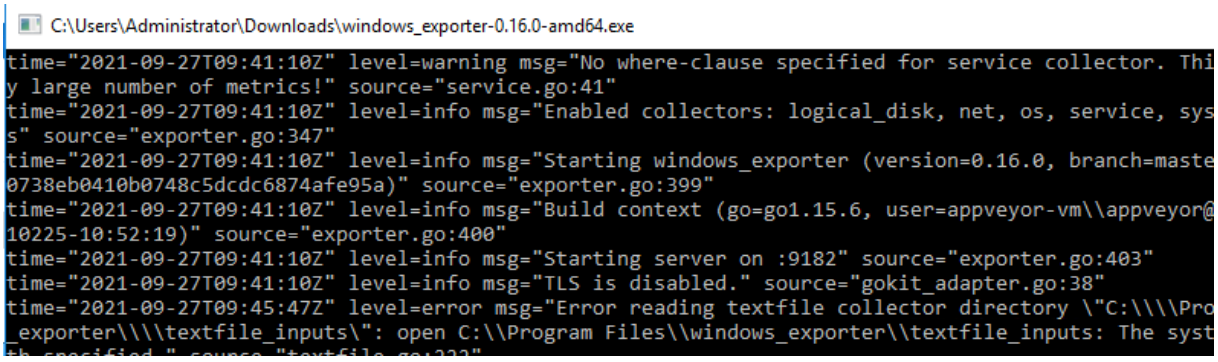


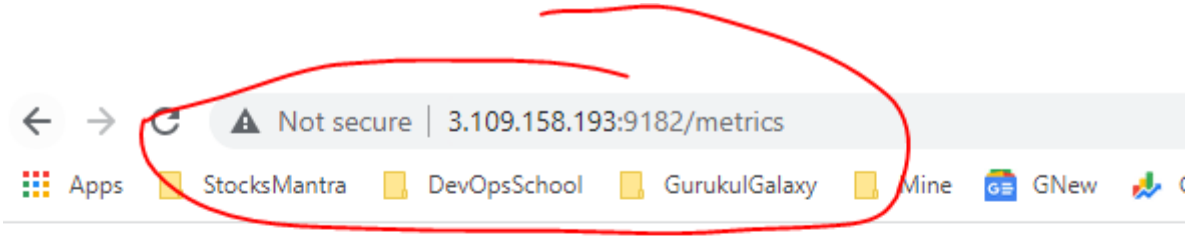
Étape 1 : Téléchargez et installez l'exportateur Windows

https://prometheus.io/docs/instrumenting/exporters/
https://github.com/prometheus-community/windows_exporter
https://github.com/prometheus-community/windows_exporter/releases
windows_exporter-0.16.0-amd64.msi – Installé avec les services
windows_exporter-0.16.0-amd64.exe – Exécutez Adhoc à l'écran

Étape 2 : Démarrez un exportateur Windows

Le port auquel se lier. La valeur par défaut est 9182.On valide – http://ip-address:9182/metrics






```
# HELP go_gc_duration_seconds A summary of the pause duration of garbage collectio
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 0
go_gc_duration_seconds{quantile="0.25"} 0
go_gc_duration_seconds{quantile="0.5"} 0
go_gc_duration_seconds{quantile="0.75"} 0
go_gc_duration_seconds{quantile="1"} 0.0010003
go_gc_duration_seconds_sum 0.0049813
go_gc_duration_seconds_count 102
# HELP go_goroutines Number of goroutines that currently exist.
# TYPE go_goroutines gauge
go_goroutines 20
# HELP go_info Information about the Go environment.
# TYPE go_info gauge
go_info{version="go1.15.6"} 1
# HELP go_memstats_alloc_bytes Number of bytes allocated and still in use.
# TYPE go_memstats_alloc_bytes gauge
go_memstats_alloc_bytes 8.399432e+06
# HELP go_memstats_alloc_bytes_total Total number of bytes allocated, even if free
# TYPE go_memstats_alloc_bytes_total counter
go_memstats_alloc_bytes_total 4.82389512e+08
# HELP go_memstats_buck_hash_sys_bytes Number of bytes used by the profiling bucke
```

Étape 3 : Configurez Promethus prometheus.yml pour la cible

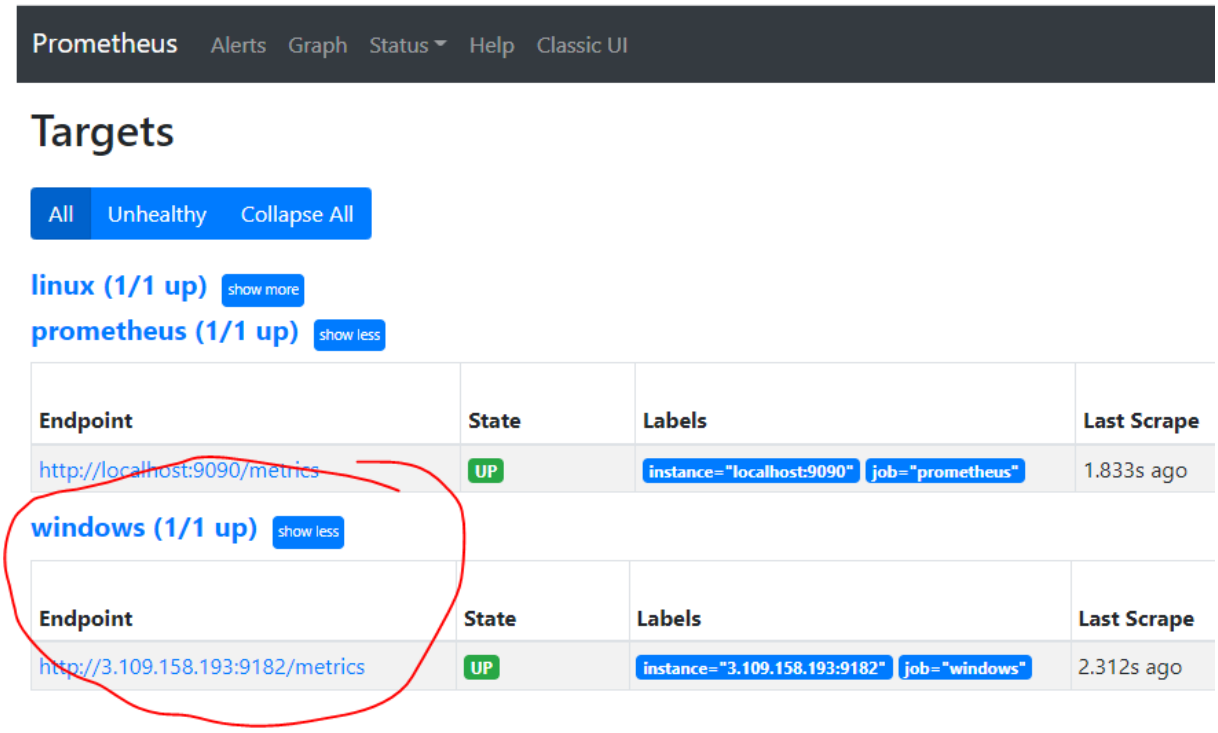
```
- job_name: windows
  static_configs:
  - targets: ['3.109.158.193:9182']
```

```
# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scr
  - job_name: "prometheus"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["localhost:9090"]

  - job_name: linux
    static_configs:
      - targets: ['localhost:9100']
  - job_name: windows
    static_configs:
      - targets: ['3.109.158.193:9182']
```

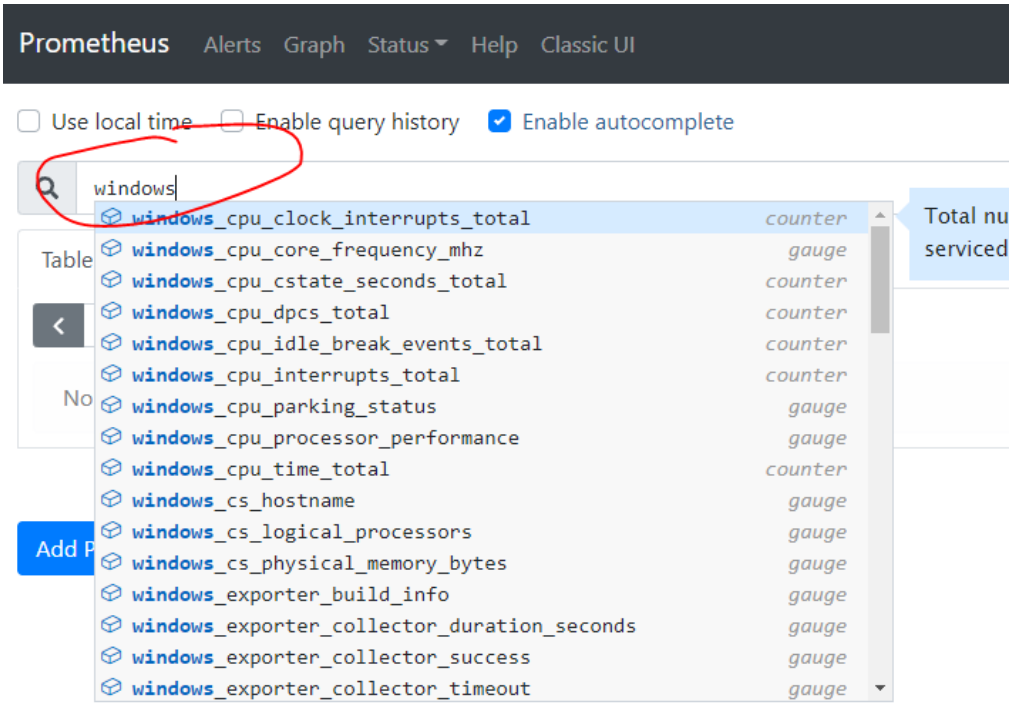
Étape 4 : Redémarrer ou recharger un Prometheus

```
sudo systemctl restart prometheus
```

Étape 5 : Valider qu'une cible a été ajoutée dans Promethus



Étape 6 : Valider une métrique Windows



Linux exporter

Nous allons voir comment reliée des environnements Linux à Prometheus/Grafana

Étape 1 : Téléchargement de Node Exporter

```
wget https://github.com/prometheus/node_exporter/releases/download/v1.2.2/node_exporter-1.2.2.linux-amd64.tar.gz
```

Étape 2 : Extraction de l'archive
```
tar xvfz node_exporter-1.2.2.linux-amd64.tar.gz
```

Étape 3 : Déplacement vers le répertoire d'installation
```
cd node_exporter-1.2.2.linux-amd64
```

Étape 4 : Déplacement de l'exécutable vers /usr/local/bin
```
sudo cp node_exporter /usr/local/bin
```

Étape 5 : Création d'un utilisateur pour Node Exporter
```
sudo useradd -rs /bin/false node_exporter
```

Étape 6 : des permissions appropriées
```
sudo chown node_exporter:node_exporter /usr/local/bin/node_exporter
```

Étape 7 : Création d'un service systemd pour Node Exporter

```
sudo tee /etc/systemd/system/node_exporter.service > /dev/null <<EOL
[Unit]
Description=Node Exporter
Wants=network-online.target
After=network-online.target

[Service]
User=node_exporter
Group=node_exporter
Type=simple
ExecStart=/usr/local/bin/node_exporter

[Install]
WantedBy=default.target
EOL
```

Étape 8 : Redémarrage du démon systemd et activation du démarrage automatique au démarrage
```
sudo systemctl daemon-reload
sudo systemctl start node_exporter
sudo systemctl enable node_exporter
```

Maintenant on les métriques sont accessible à http://localhost:9100/metric

Étape 9 : Configurez Promethus promethus.yml pour la cible

```
- job_name linux
   static_configs:
     - targets: ['3.109.158.193:910']
```

```
# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scr
  - job_name: "prometheus"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["localhost:9090"]

  - job_name: linux
    static_configs:
      - targets: ['localhost:9100']
  - job_name: windows
    static_configs:
      - targets: ['3.109.158.193:9182']
```

Solutions rendre les Dashboard accessible

Authentification anonyme :
On rend Grafana accessible sans aucune connexion requise en activant l'accès anonyme dans le fichier de configuration.
sudo nano /etc/grafana/grafana.ini
on active cette authentifications et on enlève « **;** »
[auth.anonymous]
enabled = true

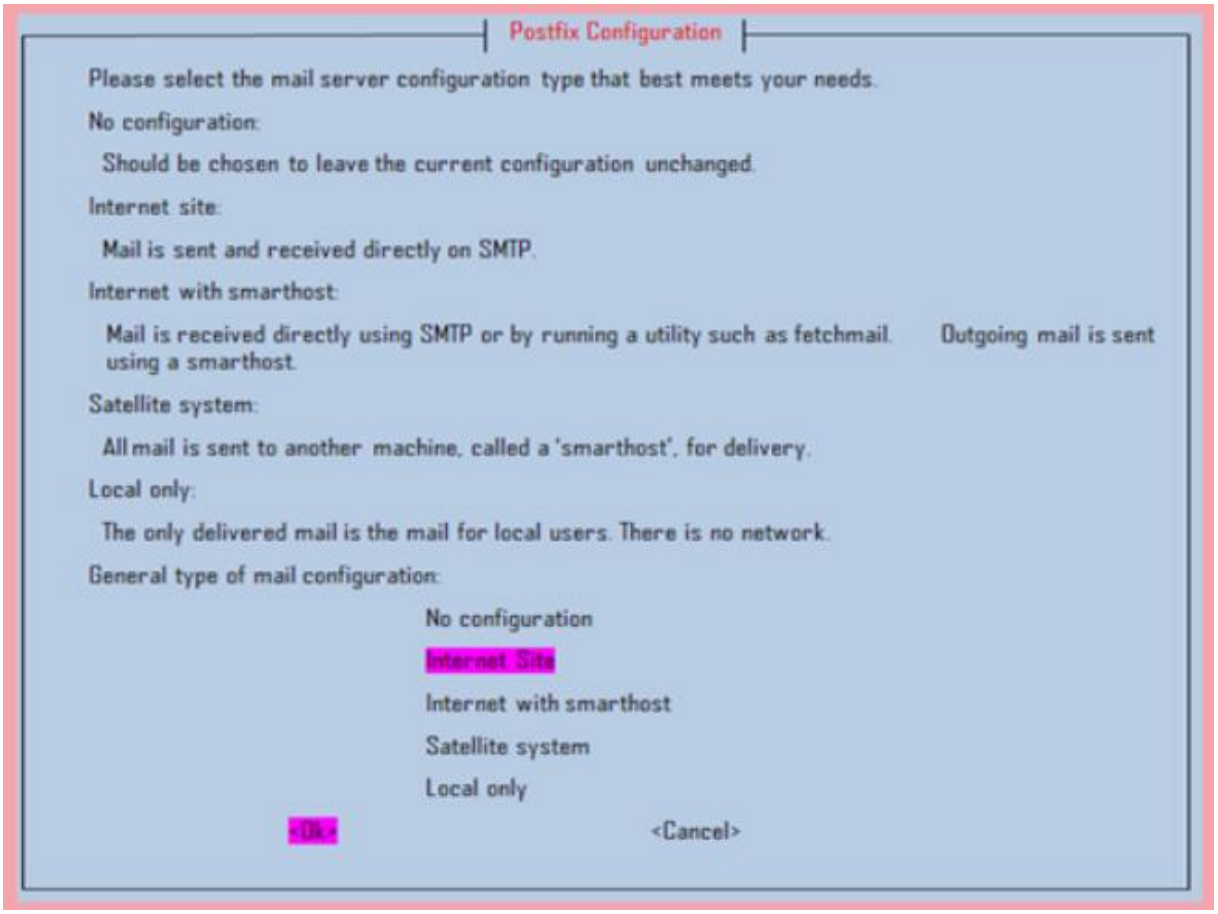
C. Installations serveur SMTP

Pour installer un serveur SMTP sur Ubuntu, vous pouvez utiliser Postfix, qui est un agent de transfert de courrier (Mail Transfer Agent - MTA) populaire. Voici comment vous pouvez l'installer et le configurer :

Étape 1 : Mettre à Jour l'Index des Paquets :
sudo apt update

Étape 2 : Installer Postfix :
sudo apt install postfix

Pendant le processus d'installation, on vous demandera de choisir le type de configuration du serveur. Vous pouvez choisir "Site Internet" puis saisir votre nom de domaine.



Étape 3 : Configurer Postfix
sudo dpkg-reconfigure postfix

Étape 4 : Démarre Postfix
sudo systemctl start postfix

Étape 5 : Activer le Démarrage Automatique de Postfix au Démarrage
sudo systemctl enable postfix

Étape 6 : Vérifier l'État de Postfix
sudo systemctl status postfix

Étape 7 : Vérifie le port écoute 25 (le port SMTP par défaut) en utilisant la commande netstat :
sudo netstat -tuln | grep 25

Avec ces étapes, Postfix devrait être installé et en cours d'exécution sur votre serveur Ubuntu. Vous devrez peut-être le configurer davantage en fonction de vos besoins spécifiques, tels que la configuration de comptes de messagerie, d'alias ou de mesures de sécurité supplémentaires.

Pour envoyer et recevoir des e-mails depuis le serveur en cours d'exécution exact, vous devez configurer Postfix. Pour ce faire, le serveur sera l'hôte local et Postfix devra être configuré pour écouter exclusivement sur l'interface de bouclage. Les serveurs utilisent cette interface comme réseau virtuel pour communiquer entre eux.
Modifiez le fichier de configuration principal de Postfix (**main.cf**), situé dans **etc/postfix** , pour apporter les modifications nécessaires.

sudo cp/etc/postfix/main.cf/etc/postfix/main.cf.backup

Maintenant, exécutez la commande ci-dessous pour accéder au fichier.
sudo nano/etc/postfix/main.cf

Ensuite, recherchez les lignes suivantes :
/etc/postfix/main.cf

```
. . .
mailbox_size_limit = 0
recipient_delimiter = +
inet_interfaces = all
. . .
```

Pour continuer, définissez la valeur du paramètre *inet_interfaces* sur *loopback-only* :
/etc/postfix/main.cf

```
. . .
mailbox_size_limit = 0
recipient_delimiter = +
inet_interfaces = loopback-only
. . .
```

Enregistrez et fermez le fichier lorsque vous avez terminé. Vous pouvez maintenant appliquer les modifications en démarrant le serveur Postfix. Alors, exécutez la commande suivante pour l'activer :

```
sudo systemctl enable postfix
sudo systemctl start postfix
```

Testez le serveur SMTP

Maintenant que vous avez installé et configuré avec succès le serveur SMTP Ubuntu, il est logique de tester si Postfix fonctionne correctement. Pour ce faire, vous pouvez utiliser telnet pour obtenir la confirmation.

```
sudo telnet Your IP
```

Si vous voyez que l'hôte local est connecté, utilisez la commande suivante pour installer les packages d'envoi de courrier de Postfix.

```
sudo apt install bsd-mailx
```

Une fois l'installation terminée, exécutez la commande ci-dessous pour utiliser l'outil qui vous permet de terminer le processus d'installation du serveur SMTP Ubuntu.

```
sudo mailx -r test@operavps -s ‘’SUBJECT’’ [YOUR-EMAIL-ADDRESS]
```

Remplacez votre adresse e-mail réelle pour faire un vrai test.

Configurer le pare-feu

Étant donné que le pare-feu peut restreindre Postfix, utilisez les commandes ci-dessous pour autoriser les fonctions Postfix dans le pare-feu.

```
sudo ufw allow ‘’Postfix’’
sudo ufw allow ‘’Postfix SMTPS’’
sudo ufw allow ‘’Postfix Submission’’
```

Vous avez terminé et maintenant un serveur de messagerie d'envoi uniquement alimenté par Postfix est prêt à être utilisé.

Activer le cryptage SMTP

Dans cette étape, vous êtes prêt à activer le cryptage SMTP. Pour ce faire, vous pouvez demander un certificat TLS gratuit à Let's Encrypt pour votre domaine. Enfin, vous configurerez Postfix pour l'utiliser lors de l'envoi d'e-mails.
Étant donné qu'Ubuntu inclut Certbot dans ses référentiels de packages par défaut, exécutez :

```
sudo apt install certbot
```

Tapez **Y** et appuyez sur Entrée pour confirmer lorsque vous y êtes invité. Vous avez installé *ufw* , le pare-feu simple, dans le cadre de la configuration préliminaire du serveur décrite dans les conditions préalables. Pour que la vérification du domaine soit effectuée, vous devez le configurer pour accepter le port HTTP 80. Pour l'activer, exécutez la commande suivante :

```
sudo ufw allow 80
```

Une fois que le résultat indique que le port est ouvert, vous êtes prêt à obtenir un certificat en exécutant la commande ci-dessous :

```
sudo certbot certonly --standalone --rsa-key-size 4096 --agree-tos --preferred-challenges http -d your_domain
```

N'oubliez pas de remplacer your-domainpar le vôtre avant d'exécuter la commande. Lorsque vous y êtes invité, entrez votre adresse e-mail.
Jusqu'à présent, votre certificat et votre fichier de clé privée ont été enregistrés sous /etc/letsencrypt/live/your_domain.
Voilà, vous avez votre certificat. Ouvrez main.cf pour le modifier :

```
sudo nano /etc/postfix/main.cf
```

Ensuite, trouvez la partie suivante :
/etc/postfix/main.cf

```
# TLS parameters
smtpd_tls_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
smtpd_tls_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
smtpd_tls_security_level=may

smtp_tls_CApath=/etc/ssl/certs
smtp_tls_security_level=may
smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
```

Pour mettre à jour vos paramètres TLS pour Postfix, modifiez-les comme suit et remplacez-les your-domainpar votre domaine.
/etc/postfix/main.cf

```
# TLS parameters
smtpd_tls_cert_file=/etc/letsencrypt/live/your_domain/fullchain.pem
smtpd_tls_key_file=/etc/letsencrypt/live/your_domain/privkey.pem
smtpd_tls_security_level=may

smtp_tls_CApath=/etc/ssl/certs
smtp_tls_security_level=may
smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
```

Enregistrez et fermez le fichier lorsque vous avez terminé. Pour appliquer les modifications que vous avez apportées, utilisez la commande ci-dessous pour redémarrer Postfix.

```
sudo systemctl restart postfix
```

D. Script

Script PowerShell

Collecte des informations sur les VMs

```
$vmArray = @()
Get-VM | ForEach-Object {
    $vmInfo = @{
        "VMName" = $_.Name
        "VMId" = $_.Id
        "State" = $_.State
        "CPUUsage" = $_.CPUUsage
        "MemoryAssigned" = $_.MemoryAssigned / 1MB
        "Uptime" = $_.Uptime.TotalSeconds
        "Status" = if ($_.State -eq 'Running') { 1 } else { 0 }
    }
    $vmArray += New-Object PSObject -Property $vmInfo
}
```

Convertir en JSON et écrire dans la sortie standard
$vmArray | ConvertTo-Json -Compress -AsArray

Détails du script :
Initialisation d'un tableau vide
$vmArray = @()
Ceci initialise un tableau vide nommé $vmArray pour stocker les informations sur chaque VM.

Récupération et traitement des informations des VM

Get-VM : Récupère une liste des VM sur le système.
ForEach-Object : Itère à travers chaque VM.
À l'intérieur de la boucle, il crée une table de hachage $vmInfo contenant les propriétés suivantes :
"**VMName":** Nom de la VM.
"VMId": Identifiant unique de la VM.
"State": État actuel de la VM (par ex., Running, Stopped).
"CPUUsage": Utilisation actuelle du CPU par la VM.
"MemoryAssigned": Mémoire attribuée à la VM, convertie de bytes en mégaoctets.
"Uptime": Temps de fonctionnement total de la VM en secondes.
"Status": Un statut personnalisé où 1 indique que la VM est en cours d'exécution et 0 indique qu'elle ne l'est pas.
New-Object PSObject -Property $vmInfo : convertit la table de hachage en un objet PowerShell et l'ajoute au $vmArray.

Conversion en JSON et sortie

$vmArray | ConvertTo-Json -Compress -AsArray
ConvertTo-Json -Compress -AsArray

Convertit le tableau d'objets VM en une chaîne JSON compressée. La chaîne JSON est ensuite écrite dans la sortie standard.

Script python

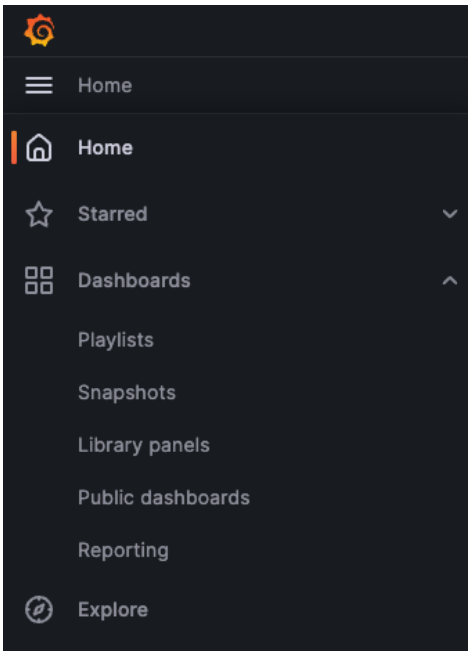
```
# Importation des modules requis
from prometheus_client import start_http_server, Gauge  # Importation de classes et de fonctions spécifiques depuis le module prometheus_client
import time  # Module pour la gestion du temps
import subprocess  # Module pour exécuter des processus externes
import json  # Module pour travailler avec des données JSON
import os  # Module pour interagir avec le système d'exploitation
import socket  # Module pour accéder aux fonctionnalités du réseau

# Configuration des paramètres
PORT = 8000  # Port sur lequel le serveur Prometheus écoutera
SCRIPT_PATH = r"C:\Users\Administrateur\Documents\status.ps1"  # Chemin vers le script PowerShell
UPDATE_INTERVAL = 15  # Fréquence de mise à jour des métriques
HOST_NAME = socket.getfqdn()  # Obtient le nom d'hôte de la machine locale

# Définition des métriques Prometheus
VM_UPTIME = Gauge("vm_uptime_seconds", "Uptime des VMs en secondes", ["vmid"])  # Métrique pour l'uptime des VMs
VM_CPU = Gauge("vm_cpu_usage_percent", "Utilisation CPU des VMs en pourcentage", ["vmid"])  # Métrique pour l'utilisation CPU des VMs
VM_STATUS = Gauge("vm_status", "Status actuel de la VM", ["vmid"])  # Métrique pour le statut actuel des VMs

# Variable pour stocker les VMID précédentes
previous_vmids = set()

# Fonction pour exécuter le script PowerShell et récupérer la sortie
def runPowershellScript(script_path):
    try:
        p = subprocess.run(["pwsh", "-File", script_path], capture_output=True, text=True)
        p.check_returncode()  # Vérifie si le processus s'est terminé avec succès
        return p.stdout  # Renvoie la sortie du script PowerShell
    except subprocess.CalledProcessError as e:
        print(f"Erreur lors de l'exécution du script PowerShell: {e.stderr}")
        raise  # Renvoie l'erreur pour obtenir une trace de la pile complète
```

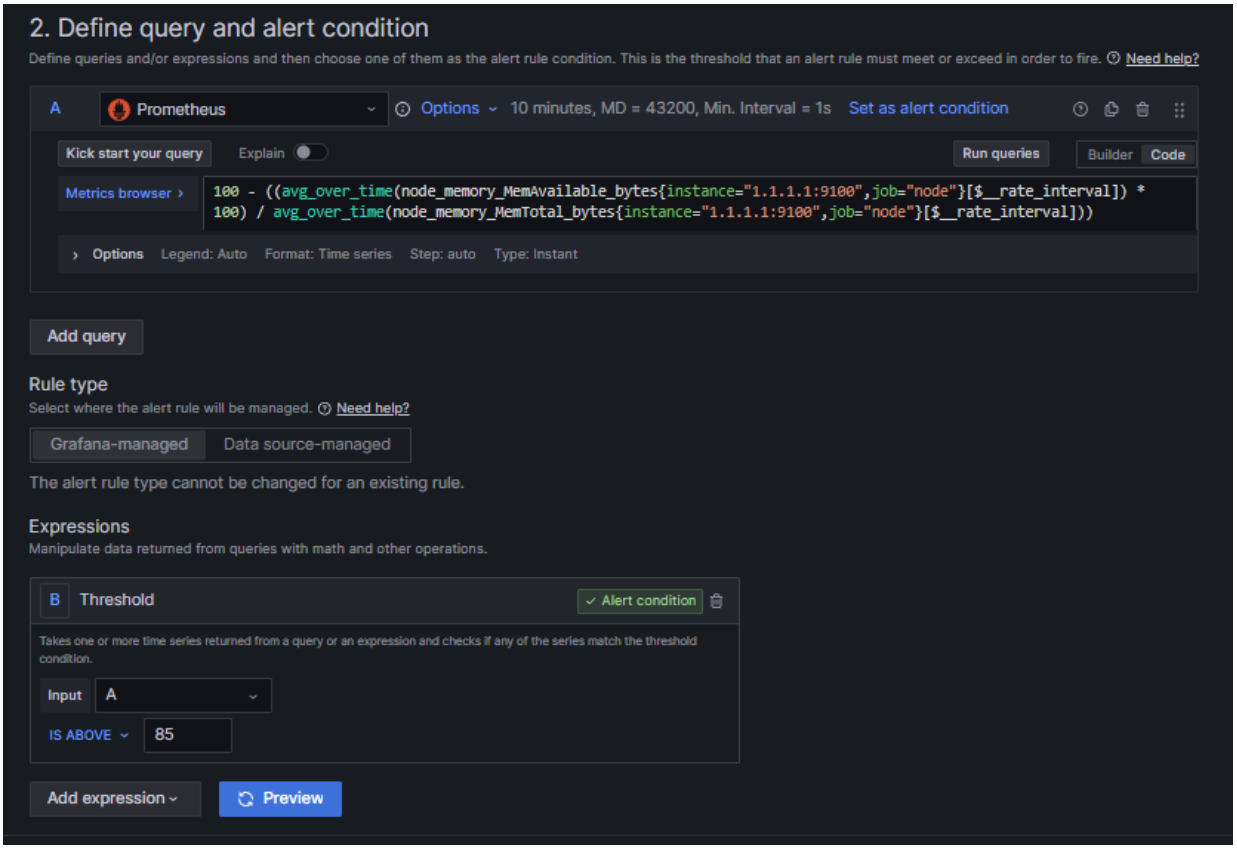
```
# Collecte des informations sur les VMs à partir du script PowerShell
def getVMInfo():
    try:
        # Exécute le script PowerShell pour obtenir les informations sur les VMs
        pwshOutput = runPowershellScript(SCRIPT_PATH)
        vms = json.loads(pwshOutput)  # Convertit la sortie JSON en un objet Python
        return vms  # Renvoie les informations sur les VMs
    except json.JSONDecodeError as e:
        print(f"Erreur lors de l'analyse du JSON: {e}")
        raise  # Renvoie l'erreur pour obtenir une trace de la pile complète

# Mise à jour des métriques Prometheus
def updateMetrics():
    global previous_vmids  # Utilisation de la variable globale previous_vmids pour suivre les VMIDs précédentes
    try:
        current_vmids = set()  # Crée un ensemble pour stocker les VMID actuelles

        # Récupère les informations sur les VMs
        vms = getVMInfo()

        # Met à jour les métriques pour les VMs actuelles
        for vm in vms:
            vmid = vm["VMId"]  # Obtient le VMID de la VM en cours de traitement
            current_vmids.add(vmid)  # Ajoute le VMID actuel à l'ensemble des VMID actuelles
            VM_STATUS.labels(vmid=vmid).set(1 if vm["Status"] else 0)  # Met à jour la métrique de statut de la VM
            VM_UPTIME.labels(vmid=vmid).set(vm["Uptime"])  # Met à jour la métrique d'uptime de la VM
            VM_CPU.labels(vmid=vmid).set(vm["CPUUsage"])  # Met à jour la métrique d'utilisation CPU de la VM

        # Supprime les métriques des VMs qui ont été supprimées depuis la dernière itération
        for previous_vmid in (previous_vmids - current_vmids):
            VM_STATUS.remove(previous_vmid)  # Supprime la métrique de statut de la VM
            VM_UPTIME.remove(previous_vmid)  # Supprime la métrique d'uptime de la VM
            VM_CPU.remove(previous_vmid)  # Supprime la métrique d'utilisation CPU de la VM

    except Exception as e:
        print(f"Une erreur est survenue lors de la mise à jour des métriques: {e}")
        raise  # Renvoie l'erreur pour obtenir une trace de la pile complète

    # Stocke les VMID actuelles pour la prochaine itération
    previous_vmids = current_vmids.copy()

# Point d'entrée du script
if __name__ == '__main__':
    # Démarre le serveur Prometheus
    start_http_server(PORT)
    print(f"Serveur Prometheus démarré sur le port {PORT}.")

    # Boucle principale
    while True:
        updateMetrics()  # Met à jour les métriques Prometheus
        time.sleep(UPDATE_INTERVAL)  # Attend l'intervalle de mise à jour avant la prochaine itération
```

Conclusion :

Ces scripts permettent de surveiller les VMs en temps réel, d'exposer des métriques précises et structurées pour l'analyse et l'alerte, et de les rendre accessibles à Prometheus pour une surveillance continue

E. Fonctionnalités Grafana





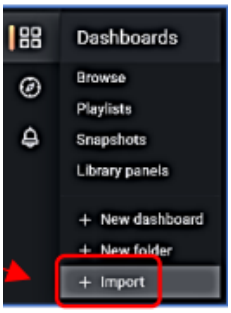
Fonctionnalités de Grafana :

1. **Visualisation de données** : Permet de créer et afficher des graphiques interactifs pour analyser des données en temps réel.
2. **Sources de données** : Connecte Grafana à diverses bases de données et services pour agréguer et visualiser les données.
3. **Tableaux de bord** : Offre des tableaux de bord personnalisables pour surveiller et analyser les données à partir de multiples sources.
4. **Alertes** : Configure des alertes basées sur des seuils spécifiques pour recevoir des notifications en temps réel.
5. **Exploration de données** : Facilite l'analyse rapide et approfondie des données sans avoir besoin de tableaux de bord prédéfinis.
6. **Authentification et sécurité** : Intègre des systèmes d'authentification et de contrôle d'accès pour sécuriser l'accès aux données et tableaux de bord.
7. **Annotations** : Permet d'ajouter des annotations aux graphiques pour marquer des événements ou des notes importantes.
8. **Plugins et intégrations** : Étend les fonctionnalités de Grafana avec des plugins pour de nouvelles sources de données et panneaux de visualisation.
9. **Gestion des utilisateurs et des équipes** : Gère les utilisateurs et les équipes avec des permissions spécifiques pour accéder et modifier les tableaux de bord.
10. **Multitenant** : Supporte plusieurs organisations distinctes avec isolation des données et des configurations.
11. **Enregistrements et révisions** : Suivi et gestion des modifications apportées aux tableaux de bord avec la possibilité de revenir à des versions antérieures.
12. **Notifications avancées** : Permet des notifications conditionnelles et l'intégration avec des systèmes de gestion des incidents.
13. **API et automation** : Fournit une API RESTful pour automatiser la gestion des tableaux de bord, sources de données et alertes.
14. **Support des variables** : Utilise des variables pour créer des tableaux de bord dynamiques et réutilisables, filtrant les données selon les besoins.
15. **Thèmes personnalisés** : Personnalise l'apparence de Grafana avec des thèmes, incluant des thèmes clairs et sombres.

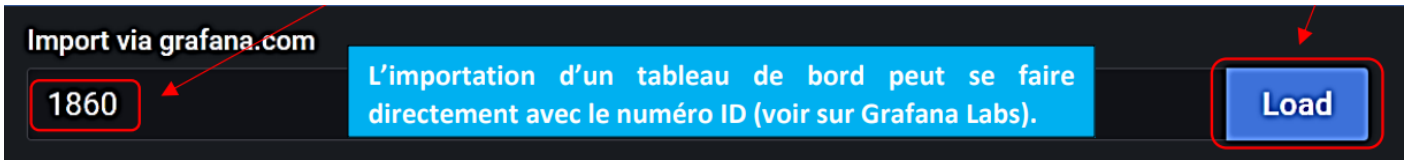
F. Dashboard

Une fois que Node exporter a été installé en tant que service dans Prometheus, l'intérêt est de pouvoir réaliser des tableaux de bord clairs, modernes, détaillés (et ludiques) dans Grafana. Sur Grafana Labs, de nombreux modèles de tableaux de bord sont proposés dans la rubrique « Nodes Dashboard ». Dans Grafana Labs choisissez un modèle de Dashboard et repérez son « ID » : ici, nous allons importer le tableau de bord « Node exporter Full » qui possède l'ID 1860 :

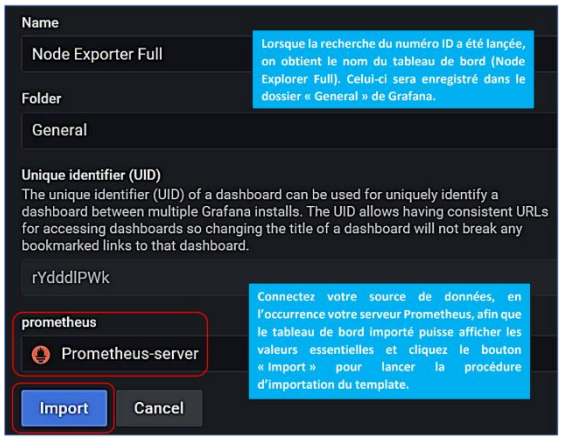
- Connectez-vous à l'interface Grafana depuis votre navigateur
- Cliquez, dans le menu de gauche, sur « Dashboard » - « Import »



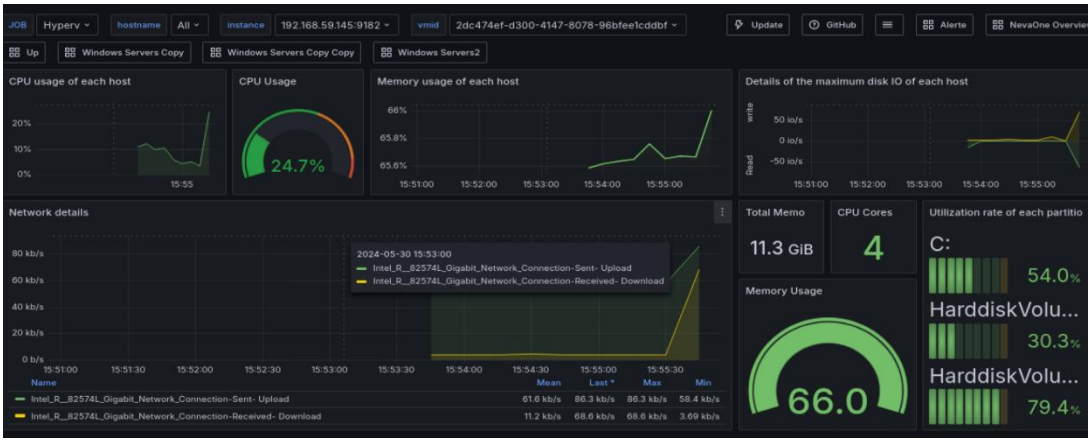
Saisissez le numéro ID qui correspond au modèle de Dashboard



Indiquez la source des données (votre serveur Prometheus) et cliquez le bouton « Import » :



Le tableau de bord est importé et affiche l'ensemble des mesures envoyées par Prometheus :



Installer Polystat :

```
grafana-cli plugins install grafana-polystat-panel
```

Redémarrer Grafana

Après l'installation du plugin, redémarrez Grafana pour que le nouveau plugin soit pris en compte :

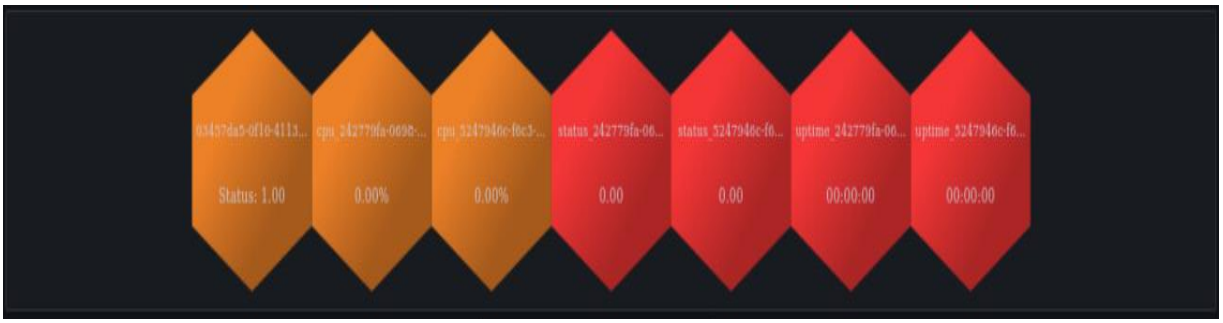
```
sudo systemctl restart grafana-server
```

Vérification de l'installation

Une fois Grafana redémarré, connectez-vous à votre interface Grafana via votre navigateur. Dans le menu de configuration des plugins pour vérifier que Polystat est bien installé :
Connectez-vous à Grafana.
Cliquez sur l'icône d'engrenage (Configuration) dans la barre latérale.
Sélectionnez "Plugins".
Recherchez "Polystat Panel" pour vérifier son installation.

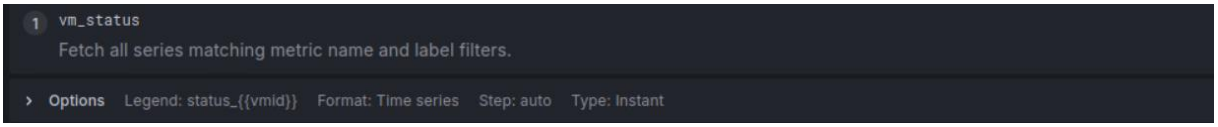
Ajouter un panneau Polystat à votre tableau de bord

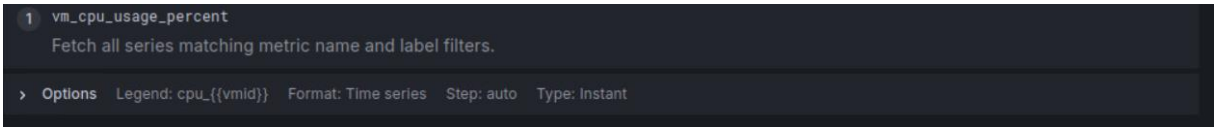
Pour ajouter un panneau Polystat à votre tableau de bord Grafana :
Ouvrez ou créez un tableau de bord.
Cliquez sur "Add Panel".
On sélectionnez "Polystat" dans la liste des types de panneaux disponibles.
Configurez le panneau en fonction de nos besoins.
Nous pouvons voir de nombreuses cellules représentant les VM dans HyperV

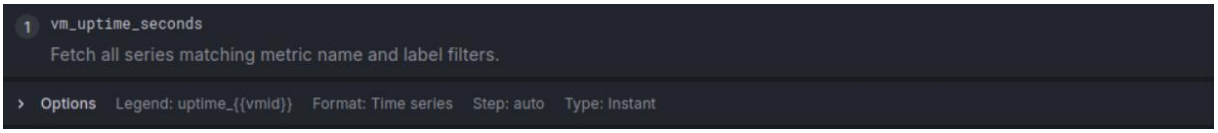


Commandes :

Ces commandes affichent le statut, le pourcentage de CPU et le temps depuis qu'elles sont démarrés dans Hyper v et chaque cellule est représentatifs des commandes ci-dessous pour chaque VM. Dans les options on met en format Time est la valeur dans l'instant qui nous permet de connaître l'état à chaque instant.

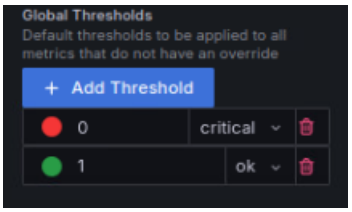






Seuil :

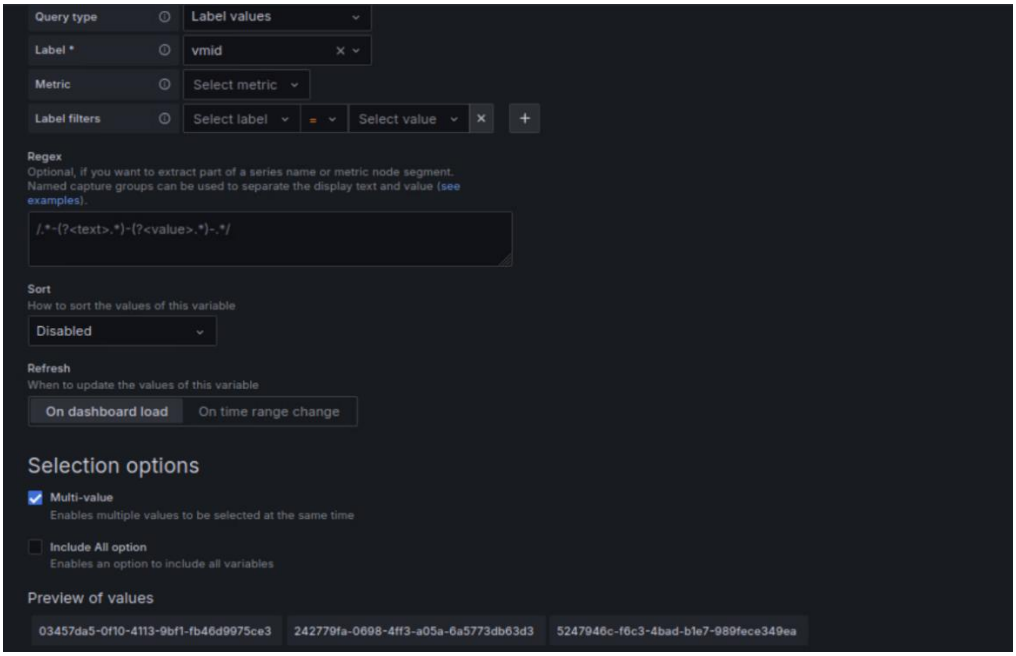
J'ai créé un seuil en fonctions de l'état des VM , rouge si elle est critique vert fonctionnel



Objectif est d'afficher une cellule qui afficheras le statuts, CPU et temps comme si dessous :

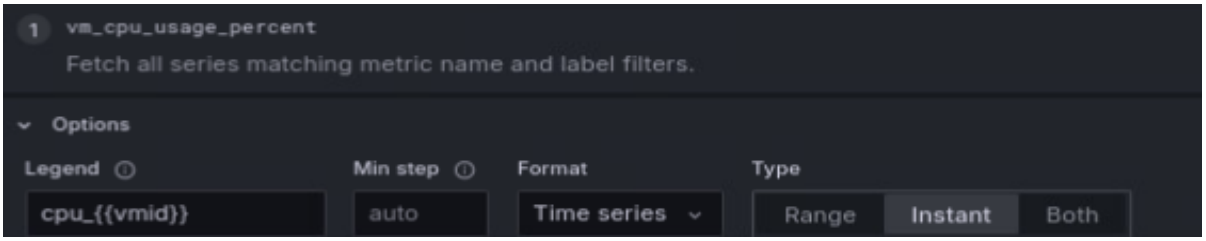
<u>Variable :</u> On va créer une variable Vmid

La fonction label values dans Grafana permet de récupérer les valeurs distinctes associées à une étiquette (label) spécifique, nous utilisons une étiquette appelée vmid pour identifier les machines virtuelles. En utilisant la fonction label values(vmid), on obtient une liste des identifiants uniques des machines virtuelles présentes dans votre source de données. Cette liste peut être utilisée pour être associée a des composites



<u>Personnaliser des métriques :</u>

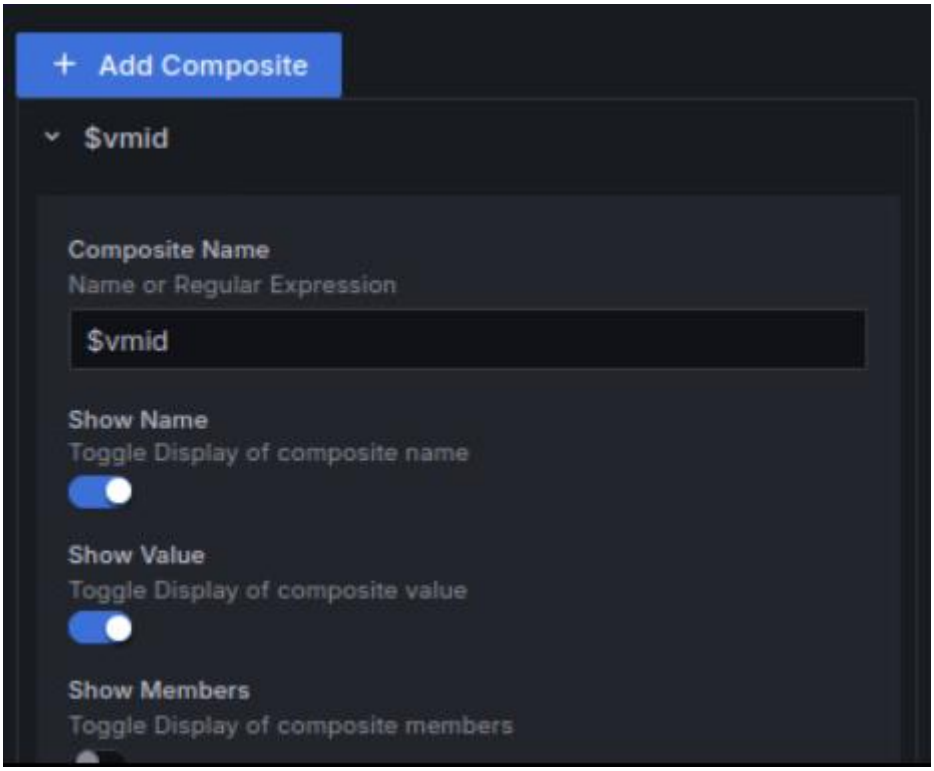
Dans le champ de personnalisation de la légende, ajoutez "commande_{{vmid}}". Cela va inclure le "vmid" dans la légende, ce qui vous permettra de distinguer les différentes séries de données par leur ID de machine virtuelle.



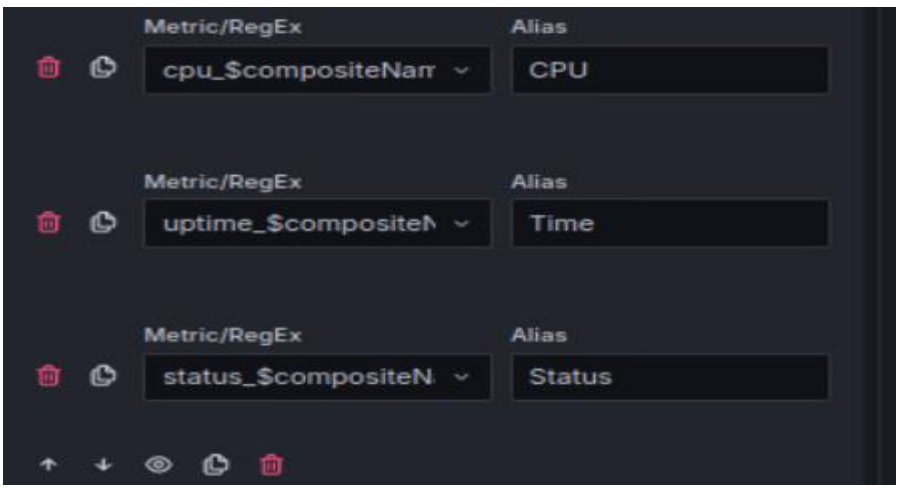
Composite :

J'ai créé un composite permettent de regrouper plusieurs panneaux dans une seule vue, ce qui est utile lorsque vous avez besoin de surveiller plusieurs métriques ou sources de données connexes en même temps.

J'ai nommé le composite "vmid" dans Grafana qui affiche les métriques "vm status", "vm cpu", et "vm uptime" pour chaque machine virtuelle, avec des légendes personnalisées incluant l'ID de la machine virtuelle (vmid). Cela vous permettra d'avoir une vue d'ensemble de l'état et des performances de vos machines virtuelles sur un seul tableau de bord.

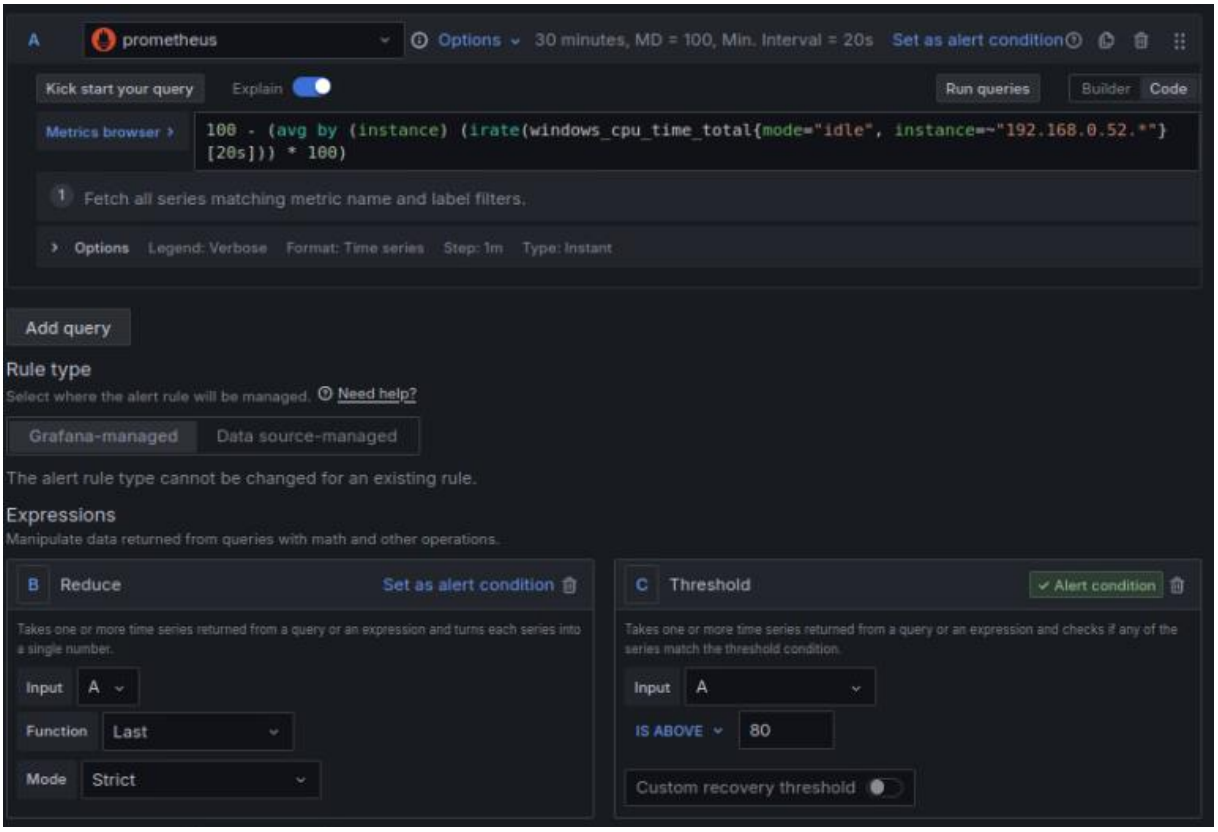


On active les options nom et les valeurs d'une série de données qui permettra d'afficher des bulles d'informations



En combinant ces métriques dans un panneau composite dans Grafana, vous pouvez obtenir une vue d'ensemble de l'utilisation du CPU, des durées d'exécution et de l'état opérationnel de vos machines virtuelles.

Alertes
J'ai créé 3 types d'alarmes pour les CPU RAM et Disk que quand on atteint 80% des ressources on reçoit une alerte par e-mail
Je vous ci-dessous exemple d'une création d'une règle pour le CPU que quand le seuil retourné est supérieur à 80% il lance une alerte par e-mail.



IV. Problèmes rencontrés

A. Permissions certificat SSL

Problèmes rencontrés: Failed to send test alert.: could not load cert or key file: open /etc/ssl/private/ssl-cert-snakeoil.key: permission denied

J'ai assuré que le fichier de certificat et de clé ont des permissions appropriées.

```
ls -l /etc/ssl/certs/ssl-cert-snakeoil.pem
ls -l /etc/ssl/private/ssl-cert-snakeoil.key
```

Les résultats devraient ressembler à ceci :

```
-rw-r--r-- 1 root root ... ssl-cert-snakeoil.pem
-rw-r--r-- 1 root ssl-cert ... ssl-cert-snakeoil.key
```

Ajouter l'utilisateur Grafana au groupe ssl-cert :

```
sudo usermod -aG ssl-cert grafana
```

Cela ajoutera l'utilisateur grafana au groupe ssl-cert, lui permettant ainsi d'accéder au fichier de clé privée. Modifier les permissions du fichier de clé privée pour inclure le groupe ssl-cert :

```
sudo chown root:ssl-cert /etc/ssl/private/ssl-cert-snakeoil.key
```

Cela modifiera le propriétaire du fichier de clé privée pour qu'il soit root et le groupe pour qu'il soit ssl-cert. Autoriser le groupe ssl-cert à lire la clé privée :

```
sudo chmod 640 /etc/ssl/private/ssl-cert-snakeoil.key
```

Cela accordera des permissions de lecture et d'écriture au propriétaire (root) et des permissions de lecture au groupe (ssl-cert). Les autres utilisateurs n'auront aucune permission.

Après avoir effectué ces modifications, redémarrez le service Grafana pour appliquer les changements :

```
sudo systemctl restart grafana-server
```

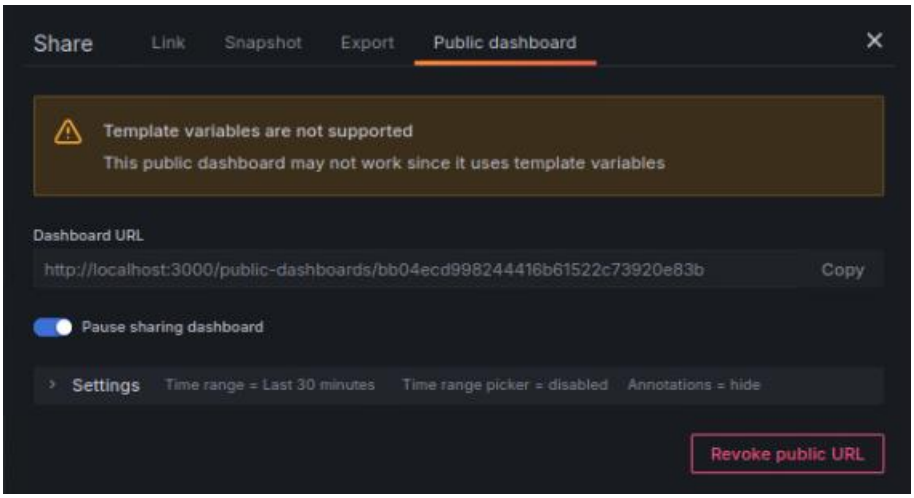
Configuration le Grafana SMTP :

```
sudo nano /etc/grafana/grafana.ini
```

```
[smtp]
enabled = true
host = mail.neva-one.fr:587
user = grafana@neva-one.fr
password = Grafana@67
# If the password contains # or ; you have to wrap it with triple quotes. Ex """#password;"""
password = adN9toiQpHFi$DAh
cert_file = /etc/ssl/certs/ssl-cert-snakeoil.pem
key_file = /etc/ssl/private/ssl-cert-snakeoil.key
skip_verify = true
from_address = grafana@neva-one.fr
from_name = Grafana
# EHLO identity in SMTP dialog (defaults to instance_name)
;ehlo_identity = dashboard.example.com
# SMTP startTLS policy (defaults to 'OpportunisticStartTLS')
;startTLS_policy = NoStartTLS

[smtp.static_headers]
# Include custom static headers in all outgoing emails
;Foo-Header = bar
;Foo = bar

[emails]
;welcome_email_on_sign_up = false
;templates_pattern = emails/*.html, emails/*.txt
;content_types = text/html
```

Authentification anonyme :

Un des objectifs est de partager le Dashboard de façons sécurisée, mais Grafana bloque les partages de Dashboard due aux variables modèles sinon la valeur retournée serait statique.



Pour rendre Grafana accessible sans aucune connexion requise en activant l'accès anonyme dans le fichier de configuration.

/etc/grafana/grafana.ini

[auth.anonymous]
enabled = true

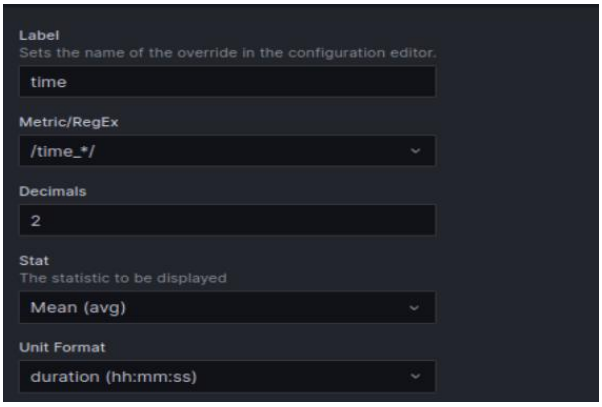
# Organization name that should be used for unauthenticated users
org_name = Main Org.

# Role for unauthenticated users, other valid values are `Editor` and `Admin`
org_role = Viewer

Activer l'authentification anonyme expose vos tableaux de bord à tout le monde sans restriction. J'ai limité les utilisateurs anonymes au rôle de "Viewer", ce qui ne leur permet que de visualiser les tableaux de bord, et de n'apporter aucune modification.

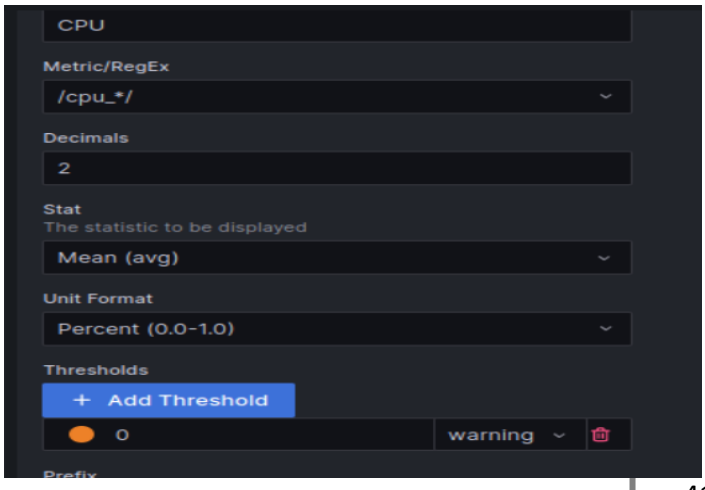
Valeurs retournées :

Valeurs retournées en Ko j'ai cherché comment retourner ces valeurs en temps. Override m'a permis de retourner une donnée en une valeurs en temps (hhh :mm :ss)



Seuils CPU personnalisés :

Créer un seuil personnalisé au CPU car quand il se mettait en veille cela rendait les cellules composites rouges et cela peut porter à confusions. Pour régler cela j'ai créé un seuil personnalisé quand la valeur retournée pour indiquer quelle est en veille est la cellule devient Orange et que sa valeur soit en pourcentage



Projet2 :

III. Gestions de projet

A. Description de la situation initiale

Chez NevaOne, la gestion des mots de passe pour les différentes infrastructures clients était dispersée et peu sécurisée. Il est devenu nécessaire de mettre en place une solution centralisée pour stocker, partager et sécuriser les mots de passe. Vaultwarden, un gestionnaire de mots de passe auto-hébergé, a été identifié comme une solution viable pour répondre à ces besoins.

B. Le contexte et la problématique

Le principal défi était de garantir une gestion sécurisée des mots de passe tout en permettant un accès contrôlé et partagé entre les différents utilisateurs internes de NevaOne. Le contexte incluait des considérations de sécurité, d'efficacité opérationnelle, et de facilité d'utilisation pour les utilisateurs finaux.

C. L'analyse fonctionnelle des besoins

Les besoins fonctionnels identifiés comprenaient :
Sécurité : Utilisation de chiffrement pour protéger les mots de passe.
Accessibilité : Accès facile et sécurisé via une interface web.
Partage : Capacité de partager des mots de passe avec des utilisateurs ou des groupes spécifiques.
Gestion des utilisateurs : Création et gestion de comptes utilisateurs individuels.
Organisation : Structuration des mots de passe par client et par infrastructure.
Audit : Suivi des accès et modifications pour assurer la conformité et la sécurité.

D. Positionnement (responsabilité confiée, autonomie, part de pilotage)

Le projet m'a été confié avec une autonomie relative, sous la supervision de mon tuteur Alexandre Drubbels et Alexis Koestel.

Tâches à Réaliser	Stagiaire	Tuteurs
1. Préparation de l'environnement	R	I, A, C
Accéder au lab pour les tests	R	I, A, C
Créer les serveurs nécessaires	R	I, A, C
2. Installation de Docker	R	I, A, C
Installer Docker sur le serveur	R	I, A, C
Vérifier la compatibilité	R	I, A, C
3. Installation de Docker Compose	R	I, A, C
Installer Docker Compose	R	I, A, C
Assurer la compatibilité des versions	R	I, A, C
4. Création du fichier docker-compose.yml	R	I, A, C
Définir les services et configurations	R	I, A, C

5. Création du répertoire de données	R	I, A, C
Créer un répertoire dédié	R	I, A, C
Configurer les permissions	R	I, A, C
6. Démarrage du serveur Vaultwarden	R	I, A, C
Utiliser Docker Compose	R	I, A, C
Vérifier l'accessibilité	R	I, A, C
7. Configuration initiale de Vaultwarden	R	I, A, C
Configurer les paramètres de sécurité	R	I, A, C
8. Configuration du serveur web (Nginx)	R	I, A, C
Configurer Nginx	R	I, A, C
Mettre en place les règles de sécurité	R	I, A, C
9. Sécurité et accès	R	I, A, C
Configurer l'accès sécurisé	R	I, A, C
Mettre en place l'authentification	R	I, A, C

10. Documentation et formation	R	I, A, C
Documenter les étapes et processus	R	I, A, C
Former les équipes et utilisateurs	R	I, A, C
11. Tests et validation	R	I, A, C
Effectuer des tests de fonctionnalité	R	I, A, C
Valider le bon fonctionnement	R	I, A, C
12. Mise en production	R	I, A, C
Déployer en production	R	I, A, C
Suivi initial pour résolution des problèmes	R	I, A, C
13. Suivi et amélioration continue	R	I, A, C
Surveiller performances et efficacité	R	I, A, C
Recueillir les retours et améliorer	R	I, A, C

R (réalisateur) : personne qui réalise la tâche
A (approbateur) : personne qui supervise et approuve la réalisation de la tâche.
C (consulté) : personne qui conseille en fonction de son domaine de compétence.
I (informé) : personne qui est informé des progrès réalisés

E. Objectifs fixés

Objectifs					
1.Installer et configurer Vaultwarden sur un serveur Docker.	**1.Créer des comptes utilisateurs** pour chaque membre de l'équipe.	**1.Organiser les mots de passe** par client et par infrastructure.	**1.Mettre en place des partages sécurisés** entre les utilisateurs.	**1.Documenter les procédures** et former les utilisateurs internes.	**1.Assurer la sécurité des données** par des configurations appropriées et des tests rigoureux.

F. Contraintes

Les contraintes du projet incluaient :
Sécurité : Assurer le chiffrement et la protection des données sensibles.
Compatibilité : Intégration avec l'infrastructure existante.
Temps : Respect des délais pour chaque phase du projet.
Ressources : Utilisation efficace des ressources matérielles et humaines disponibles.

G. Moyens et ressources

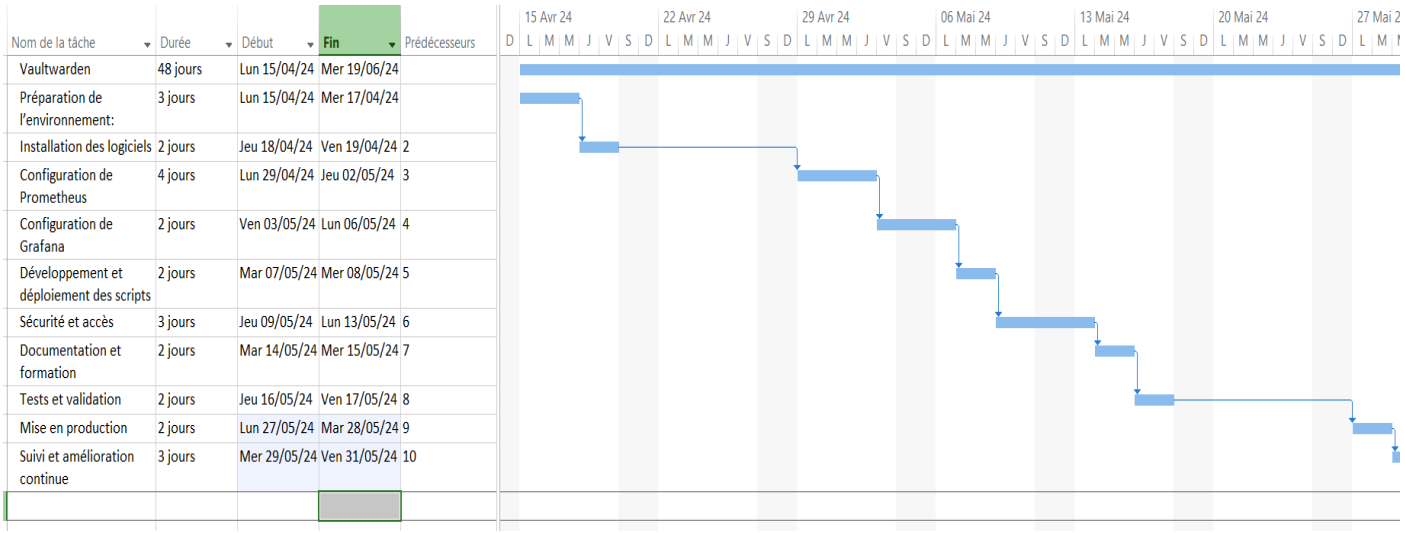
Les ressources mobilisées pour ce projet comprenaient :
Matériel : Serveur pour l'hébergement de Vaultwarden.
Logiciel : Docker et Docker Compose pour la gestion des conteneurs.
Humain :Moi-même ainsi que mes tuteurs.
Documentaire : Utilisation des guides et documentations de Vaultwarden.

H. Comparatif de solutions

Critère	Vaultwarden (Bitwarden RS)	Passbolt	KeePassXC	Buttercup
Licence	AGPLv3	AGPLv3	GPL v2.0	MIT
Hébergement	Auto-hébergement principalement	Auto-hébergement, options cloud hébergé	Installation locale, stockage local	Cloud, synchronisation via Dropbox
Fonctionnalités clés	Stockage sécurisé, synchronisation multi-appareils, partage sécurisé, intégrations	Chiffrement OpenPGP, partage sécurisé, gestion des utilisateurs, intégration LDAP	Chiffrement local, gestion des mots de passe, générateur de mots de passe	Chiffrement AES-256, synchronisation, intégration cloud
Sécurité	Chiffrement AES-256 côté client	Chiffrement OpenPGP	Chiffrement AES-256, sécurité renforcée	Chiffrement AES-256, sécurité renforcée
Support et communauté	Communauté active, support robuste	Communauté active, support disponible	Communauté active, documentation complète	Communauté active, support disponible
Contrôle des données	Contrôle total avec l'auto-hébergement	Contrôle avec l'auto-hébergement, clés publiques/privées	Contrôle total avec stockage local	Contrôle avec stockage local, intégration cloud
Intégration LDAP/2FA	Oui	Oui	Oui	Oui

I. Description des tâches

Les tâches principales du projet étaient organisées comme suit :
Préparation de l'environnement : Mettre à jour les paquets et préparer le serveur pour l'installation de Docker.
Installation de Docker : Installer Docker pour la gestion des conteneurs.
Installation de Docker Compose : Installer Docker Compose pour orchestrer les conteneurs.
Création du fichier docker-compose.yml : Écrire et configurer le fichier nécessaire au déploiement de Vaultwarden.
Création du répertoire de données : Créer le répertoire vw-data pour stocker les données persistantes.
Démarrage du serveur Vaultwarden : Lancer le conteneur Vaultwarden via Docker Compose.
Configuration initiale de Vaultwarden : Accéder à l'interface web pour la configuration initiale et la création du compte administrateur.
Configuration du serveur web (Nginx) : Mettre en place un serveur Nginx pour sécuriser l'accès avec SSL.
Sécurité et accès : Mettre en place des politiques de sécurité et des accès utilisateurs.
Documentation et formation : Rédiger la documentation utilisateur et former les utilisateurs internes.
Tests et validation : Tester les fonctionnalités et valider la configuration.
Mise en production : Déployer Vaultwarden pour une utilisation quotidienne.
Suivi et amélioration continue : Assurer la maintenance et améliorer les configurations selon les retours des utilisateurs.



Nom de la tâche	Durée	Début	Fin	Prédécesseurs
Vaultwarden	48 jours	Lun 15/04/24	Mer 19/06/24	
Préparation de l'environnement:	3 jours	Lun 15/04/24	Mer 17/04/24	
Installation des logiciels	2 jours	Jeu 18/04/24	Ven 19/04/24	2
Configuration de Prometheus	4 jours	Lun 29/04/24	Jeu 02/05/24	3
Configuration de Grafana	2 jours	Ven 03/05/24	Lun 06/05/24	4
Développement et déploiement des scripts	2 jours	Mar 07/05/24	Mer 08/05/24	5
Sécurité et accès	3 jours	Jeu 09/05/24	Lun 13/05/24	6
Documentation et formation	2 jours	Mar 14/05/24	Mer 15/05/24	7
Tests et validation	2 jours	Jeu 16/05/24	Ven 17/05/24	8
Mise en production	2 jours	Lun 27/05/24	Mar 28/05/24	9
Suivi et amélioration continue	3 jours	Mer 29/05/24	Ven 31/05/24	10

IV. Projet Vaultwarden

A. Installations Docker

Étape 1: Installation de Docker
sudo apt update
sudo apt install docker.io
Étape 2: Installation de Docker Compose

sudo apt install docker-compose
Étape 3: Création d'un fichier docker-compose.yml
Créez un fichier docker-compose.yml dans un répertoire de votre choix et ajoutez le contenu suivant :

```
  GNU nano 6.2                          docker-compose.yml
version: '3'

services:
  vaultwarden:
    image: vaultwarden/server:latest
    container_name: vaultwarden
    restart: always
    environment:
      WEBSOCKET_ENABLED: "true"  # Enable WebSocket notifications.
      DOMAIN: "https://192.168.0.54"  # Remplacez par l'adresse IP de votre machine
    volumes:
      - ./vw-data:/data
    ports:
      - "8080:80"  # Mappez le port 8080 de l'hôte au port 80 du conteneur
```

Ce fichier définit un service Vaultwarden qui utilise l'image vaultwarden/server:latest, expose le port 80 et utilise un volume pour stocker les données Vaultwarden.

Étape 4: Création du répertoire de données
Créez un répertoire nommé vw-data dans le même répertoire que votre fichier docker-compose.yml. Cela permettra de stocker les données de Vaultwarden de manière persistante :

mkdir vw-data

Étape 5: Démarrage du serveur Vaultwarden
Dans le répertoire contenant votre fichier docker-compose.yml, lancez la commande suivante pour démarrer le serveur Vaultwarden :

docker-compose up -d

Cela va télécharger l'image Docker de Vaultwarden et démarrer le conteneur. Vous pourrez alors accéder à votre serveur Vaultwarden en utilisant l'adresse IP de votre machine Ubuntu.

B. Installations Nginx

Configuration Nginx et Docker :
Accéder à l'interface de Vaultwarden, l'accès s'effectue en HTTP, donc les communications ne sont pas chiffrées. Jinstalle ngix sert de médiateur pour gérer les requêtes et sécuriser les communications avec Vaultwarden Nginx sert de médiateur pour gérer les requêtes et sécuriser les communications

```
sudo apt update
sudo apt install nginx
```

Générer un certificat auto-signé :

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/vaultwarden.key -out /etc/ssl/certs/vaultwarden.crt
```

Configurer Nginx
Créez un fichier de configuration pour Vaultwarden dans Nginx :

`sudo nano /etc/nginx/sites-available/vaultwarden` Ajoutez-y le contenu suivant :

```
server {
    listen 80;
    server_name 192.168.0.54;

    location / {
        return 301 https://$host$request_uri;
    }
}

server {
    listen 443 ssl;
    server_name 192.168.0.54;

    ssl_certificate /etc/ssl/certs/vaultwarden.crt;
    ssl_certificate_key /etc/ssl/private/vaultwarden.key;

    location / {
        proxy_pass http://localhost:8080;  # Proxy to Vaultwarden on port 8080
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;
    }
}
```

Créer un lien symbolique en Unix/Linux

```
sudo ln -s /etc/nginx/sites-available/vaultwarden /etc/nginx/sites-enabled/
```

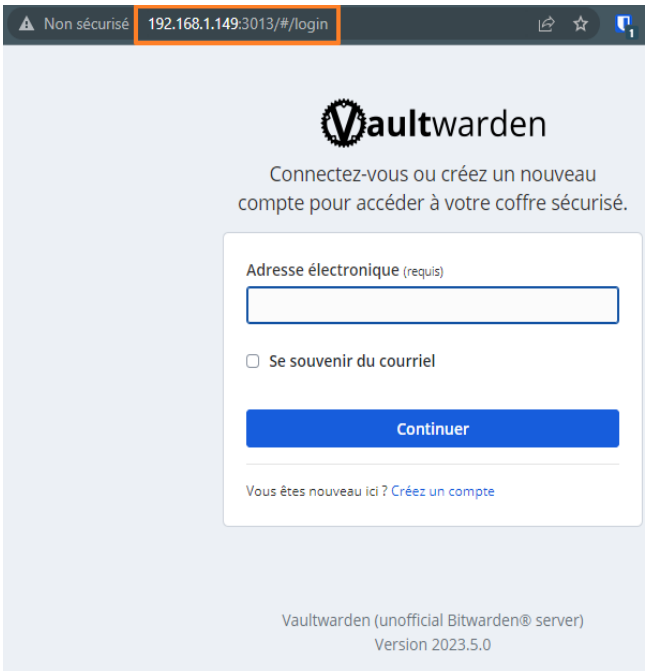
Testez la configuration de Nginx pour vous assurer qu'il n'y a pas d'erreurs :

```
sudo nginx -t
```

Redémarrez Nginx pour appliquer les changements :

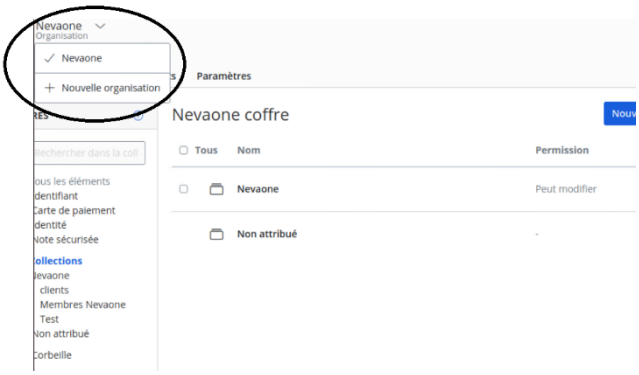
```
sudo systemctl restart nginx
```

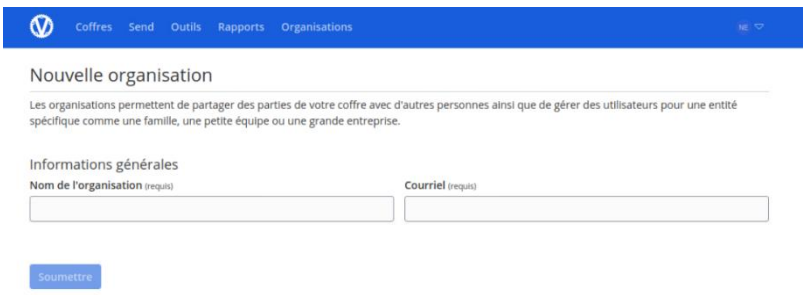
C. Découvertes de Vaultwarden



Jai crée un compte
Je crée une nouvelle organisation **Nevaone**

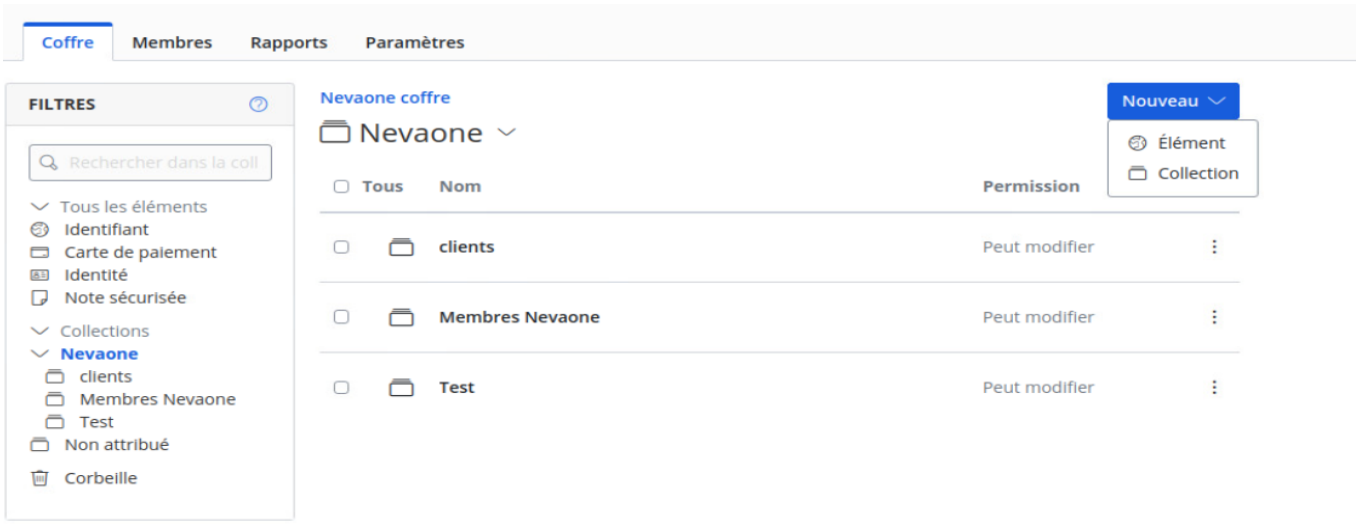
Créer une organisation



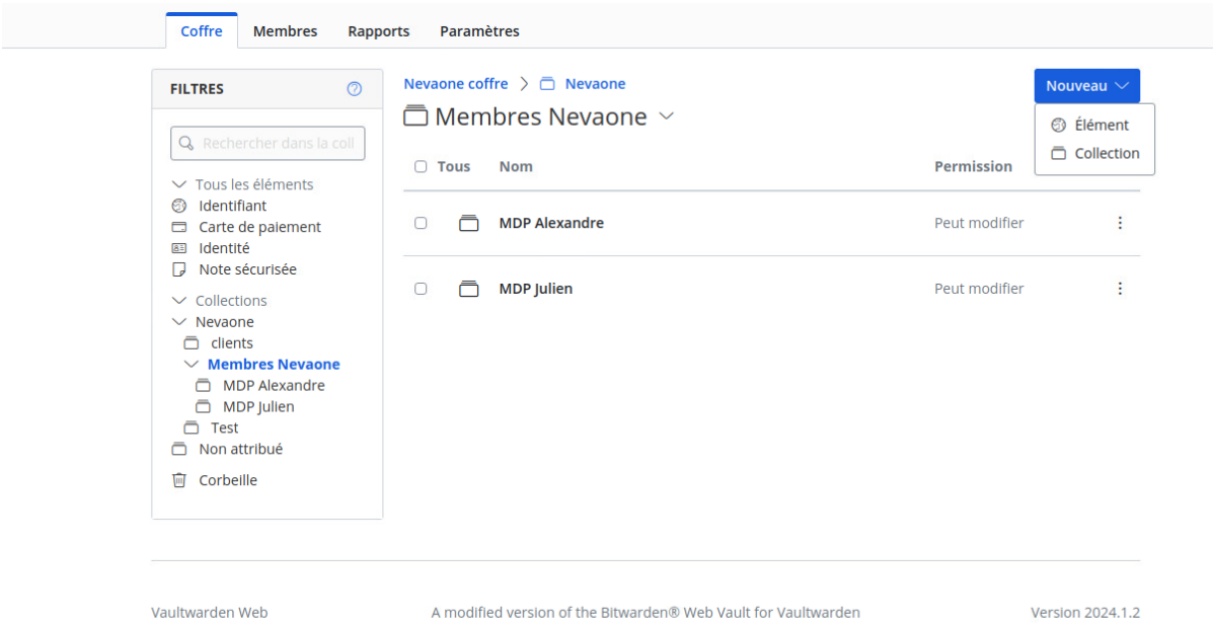


Créer une Collections
Pour créer des comptes individuels au membres de NevaOne et pour client jai crée une collections(Dossier) Nevaone avec plusieurs collections afin de les séparers et d'organiser les Mdp **Nouveu> Elément**

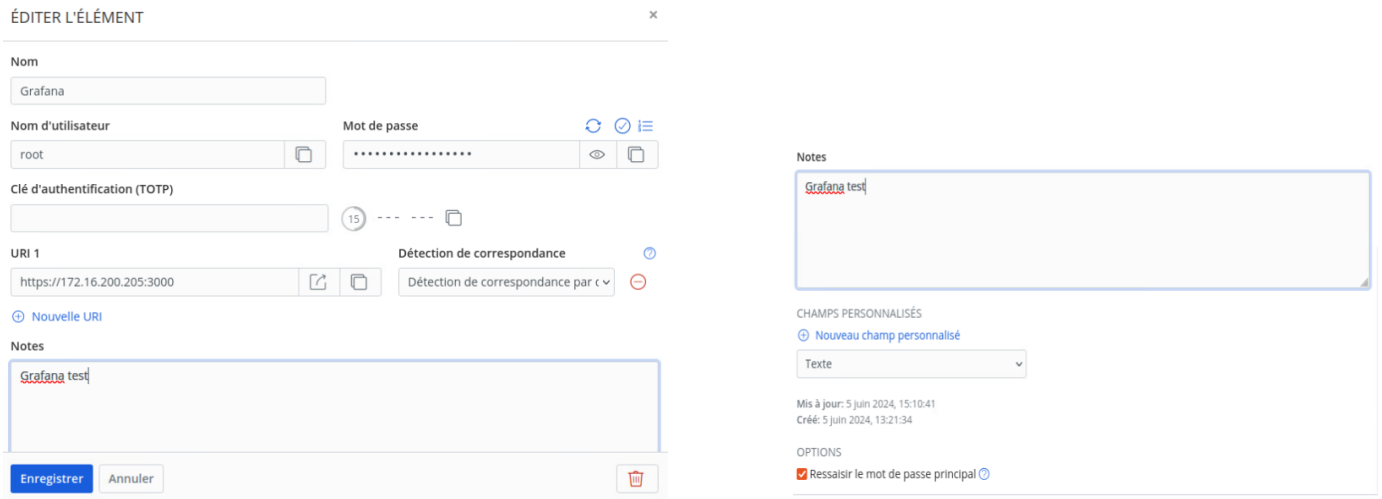
Créer un Identiefiant



Pour créer un compte a chaque membre de Nevaone ou Clients ou Ajoute un **Nouveu> Elément**

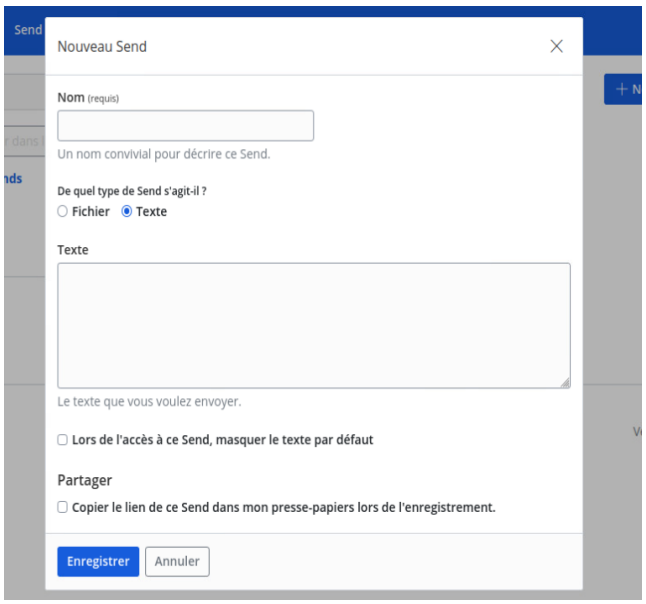


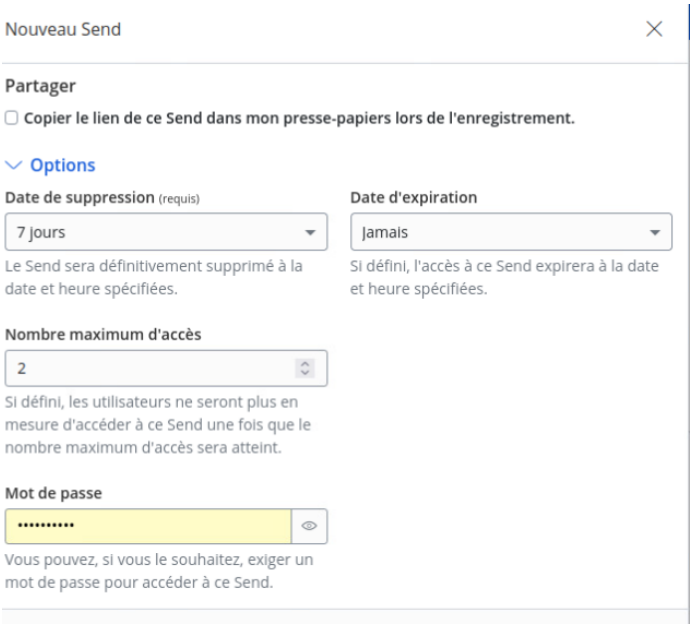
Le type ici il s'agit >Identiefiant on inscris le **Nom de la Plateforme id** le **MDP le lien de la plate fome** et des **notes ,** pour sécuriser cela quand on quitte et on reviens sur la fiche on doit mettre le **MDP Principal** pour cela on active **>Options**



Partage de fichier ou de MDP

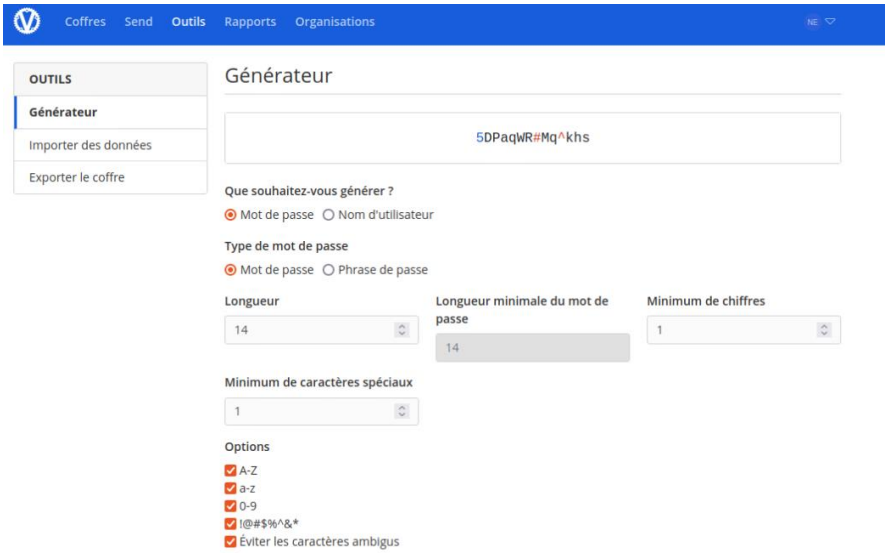
Valutwarden propose une option de partage de fichier ou de texte via un lien avec les options on peut limiter le nombre de personne qui ont accès au lien, la durée avant expirations du lien et un MDP pour sécuriser et avoir accès.





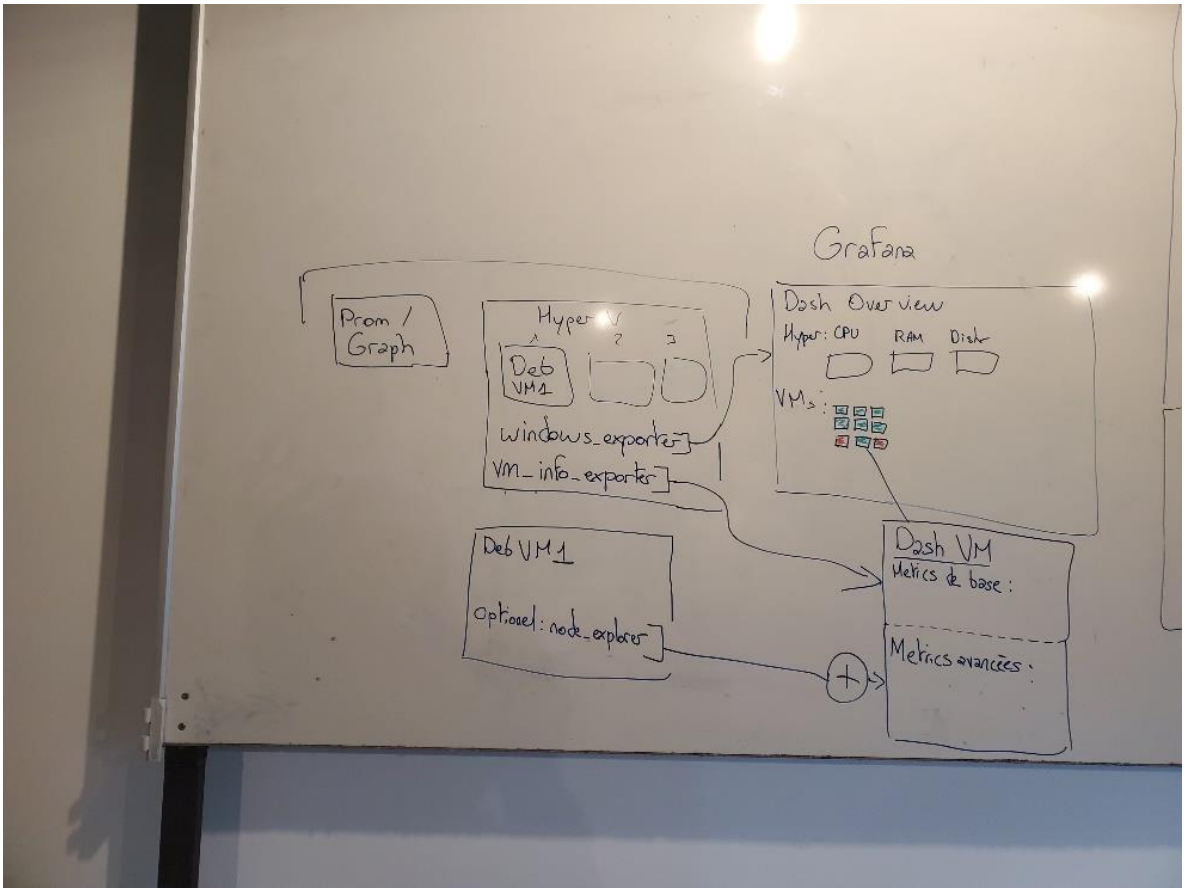
Générateurs de MDP

Il ya un outils de géneration de MDP et ID on proposant plusieurs options nombres de caractères , caractères utilisés.



Rapport :
Vaulwarden propose un service proposons les fonctionnalités suivantes pour la sécurités afin de vérifier les liens, les MDP et les emails

Annexe : Photos prise en entreprise



Annexe : Mots clés, liens et définitions

Monitoring informatique : Processus de surveillance continue des systèmes informatiques, des réseaux, des applications et des services pour détecter et répondre de manière proactive aux problèmes, optimiser les performances et garantir la disponibilité des infrastructures.
Métriques : Données quantitatives représentant divers aspects de l'état d'un système, d'une application ou d'un processus. Utilisées pour mesurer et suivre la performance, la santé et la disponibilité des ressources informatiques.

Instance : Dans Prometheus, une instance est généralement une combinaison de l'adresse IP et du port d'un service ou d'un nœud qui expose des métriques. Par exemple, "192.168.1.5 :9100" serait une instance. L'étiquette instance permet d'identifier et de différencier les métriques provenant de différentes sources ou hôtes.

Label (Étiquette): Paires clé-valeur attachées aux métriques qui permettent de les distinguer les unes des autres. Elles fournissent des dimensions supplémentaires pour l'identification, la classification et le filtrage dans les requêtes PromQL.

Prometheus : Système de monitoring et d'alerte qui collecte et stocke ses métriques sous forme de séries temporelles. Il fonctionne en récupérant les métriques d'Endpoint configurés à intervalles réguliers et permet de définir des alertes basées sur des requêtes complexes.

Grafana : Outil de visualisation des données qui permet de créer des tableaux de bord dynamiques et hautement personnalisables. Il peut se connecter à diverses sources de données, notamment Prometheus, pour afficher les données en temps réel.

Exporter : Composant logiciel qui expose les métriques d'un système pour qu'elles puissent être collectées par Prometheus. Par exemple, Node Exporter pour les serveurs Linux et Windows Exporter pour les serveurs Windows.

https://grafana.com/docs/grafana/latest/setup-grafana/installation/debian/
https://youtu.be/ecspEOdtPDc?si=pG390QY2wHDppcsV
https://grafana.com/blog/2019/04/02/grafana-plugin-tutorial-polystat-panel-part-1/#composites
https://grafana-com.translate.goog/grafana/plugins/grafana-polystat-panel/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=sc

Grafana open source documentation | Grafana documentation

Overview | Prometheus

PROMETHEUS / GRAFANA : tutos et formation sur l'observabilité

Windows Server Monitoring using Prometheus and WMI Exporter

Guide to the Prometheus Blackbox Exporter : OpsRamp

Conclusion :

Au terme de ce stage chez NevaOne, j'ai pu participer activement à des projets d'envergure qui ont contribué à l'amélioration des services de l'entreprise. Basée à Goxwiller, NevaOne se distingue par sa capacité à offrir des solutions numériques complètes et intégrées, allant de la sécurité informatique à l'hébergement, en passant par le support aux utilisateurs et la surveillance des systèmes informatiques.
Mon stage s'est principalement concentré sur deux projets majeurs : l'implémentation de systèmes de monitoring avec Grafana et Prometheus, et la mise en place de Vaultwarden pour la gestion sécurisée des mots de passe. Ces projets ont été cruciaux pour optimiser la surveillance des infrastructures clients et sécuriser les informations sensibles.

Projet Grafana et Prometheus :

Le premier projet portait sur la mise en place d'un système de monitoring avancé. L'intégration de Grafana et Prometheus a permis de surveiller en temps réel les performances et la disponibilité des infrastructures clients. Ce système a couvert une large gamme d'aspects, tels que la surveillance des serveurs Linux et Windows, la gestion des équipements FortiGate et la surveillance des connexions internet. J'ai configuré des alertes, créé des dashboards personnalisés et intégré des exporters pour une meilleure visibilité des données.

Projet Vaultwarden :

Le second projet concernait la gestion des mots de passe avec Vaultwarden. Face à la dispersion des informations sensibles, il était essentiel de centraliser et de sécuriser les mots de passe utilisés pour la gestion des infrastructures clients. J'ai installé et configuré Vaultwarden sur un serveur Docker, mis en place une organisation pour entreprise en interne et pour les clients et par infrastructure de plus, j'ai documenté les procédures.

Réflexions et apprentissages :

Ces projets m'ont permis de concrétiser mes connaissances théoriques en administration système et en sécurité informatique, tout en développant de nouvelles compétences pratiques. La mise en place de ces systèmes a non seulement renforcé la sécurité et l'efficacité des opérations chez NevaOne, mais m'a également offert une expérience précieuse dans la gestion de projets techniques.
Travailler chez NevaOne a été une expérience très positive, marquée par un environnement professionnel dynamique et collaboratif. J'ai pu évoluer dans un cadre où l'innovation et l'excellence sont au cœur des préoccupations, ce qui m'a permis de grandir tant sur le plan professionnel que personnel.
En conclusion, ce stage a été une étape déterminante de mon parcours, me permettant d'acquérir une expertise pratique et de contribuer significativement aux projets de l'entreprise.