Open source

L'**open source** désigne des logiciels dont le code source est librement accessible, permettant à tout le monde de le modifier, l'améliorer et le redistribuer. Ce modèle favorise la collaboration et l'innovation, avec des exemples comme **Linux** ou **Firefox**.

- OpenProject
  - Fonctionnalités entreprise
  - Mise à jour
- Ollama
  - Installation
  - Mise à jour
  - Modèle personnalisé
  - Prompt personnalisé
  - Désinstallation
  - Créer un modèle
  - Supprimer un modèle
  - Utilisation de l'API
- Open-WebUI
  - Installation avec pip
  - Installation avec docker
  - Mise à jour
- Docker
  - Installation (debian)
  - Mise à jour (debian)
  - Purge
- Debian
  - Commandes communes

    - Mise à jour
    - Étendre une partition sur Debian (MBR) sans perte de données
- AMP
    - Installation
- Proxmox
    - Update package database error
    - Helper-script
    - Reverse proxy avec VPS frontend
- Apache Guacamole
    - Mise en place d'une connexion SSH via clé RSA
- Grafana
    - Mise en place de node-exporter
    - Mise en place de prometheus
    - Mise en place de grafana
- Affine
    - Mise à jour (AFFiNE Self-Hosted)
    - Configuration Affine AI (OpenAI Override)
- Wazuh

OpenProject

OpenProject est une plateforme de gestion de projet open-source qui permet de planifier, suivre et collaborer sur des projets. Elle offre des fonctionnalités comme la gestion des tâches, des diagrammes de Gantt, le suivi des bugs, et la gestion des sprints pour les équipes agiles. Son caractère open-source permet de la personnaliser et de l'adapter aux besoins des utilisateurs.

Fonctionnalités entreprise

> Source : OpenProject Enterprise mode for free · GitHub

Solution :

- Remplace le fichier `/opt/openproject/app/models/enterprise_token.rb` dans le code source par le fichier enterprise_token.txt (version texte) ou enterprise_token.rb (version .rb)
- Assure-toi également de REDÉMARRER OpenProject après avoir remplacé le fichier. Il n'indique pas que le mode entreprise est activé dans les paramètres, mais toutes les fonctionnalités du mode entreprise, telles que les tableaux KanBan, sont activées.

> **Version du cœur :** OpenProject 14.6.3
> **Version de PostgreSQL :** 13.16

Mise à jour :

Si vous souhaitez mettre à jour OpenProject, il est recommandé de retirer le fichier et de remettre le fichier d'origine enterprise_token_origin.rb avant de procéder à la mise à jour. Si vous choisissez de procéder à la mise à jour sans retirer le fichier, assurez-vous ensuite d'exécuter la commande suivante :
`sudo openproject reconfigure`
Cela permettra de reconfigurer OpenProject et de restaurer son bon fonctionnement (ne modifiez rien lors de la reconfiguration pour conserver vos données). Après cette opération, il vous faudra remettre le fichier `enterprise_token.rb` manuellement.

Notez également qu'en cas de développement spécifique ou de modifications apportées au logiciel, la mise à jour peut entraîner la suppression ou rendre obsolètes vos ajouts.

> Avertissement concernant l'utilisation des fonctionnalités Enterprise payantes via un contournement

L'utilisation de fonctionnalités Enterprise payantes d'OpenProject activées par un contournement (par exemple, le remplacement de fichiers dans le code source) est strictement réservée à des fins de test ou d'évaluation dans un environnement non productif. **Cette méthode de contournement n'est pas autorisée par OpenProject** et peut entraîner des risques juridiques

et techniques.

Si vous utilisez ces fonctionnalités dans un environnement de production, **vous risquez de rencontrer des problèmes de stabilité ou de compatibilité**, et **OpenProject se réserve le droit de refuser tout support technique** en raison de l'exploitation de ces fonctionnalités via un contournement. L'utilisation non conforme peut également entraîner des sanctions, notamment en cas de non-respect des conditions d'utilisation d'OpenProject.

Nous vous conseillons vivement de souscrire à une licence officielle si vous souhaitez bénéficier des fonctionnalités Enterprise dans un cadre de production et recevoir un support dédié.

Mise à jour

Source : Upgrading your OpenProject installation

Voici les étapes pour mettre à jour OpenProject en utilisant les informations de la documentation officielle.

1. **Sauvegarde** :

   ```
   sudo openproject run backup
   ```

   Sauvegarde ton installation actuelle pour éviter toute perte de données.
2. **Mise à jour du dépôt** :

   ```
   sudo apt-get update
   ```

   Mets à jour ton dépôt de paquets pour obtenir les dernières versions disponibles.
3. **Mise à jour d'OpenProject** :

   ```
   sudo apt-get install --only-upgrade openproject
   ```

   Installe la dernière version d'OpenProject.
4. **Configuration** :

   ```
   sudo openproject configure
   ```

   Exécute la commande de configuration pour appliquer les nouvelles modifications.
5. **Redémarrage des services** :

   ```
   sudo openproject run restart
   ```

   Redémarre les services pour appliquer les changements.
6. **Vérification** : Accède à ton instance OpenProject via le navigateur pour vérifier que tout fonctionne correctement.

Voici la commande compilé :

```
sudo openproject run backup && sudo apt-get update && sudo apt-get install --only-upgrade openproject && sudo openproject configure && sudo openproject run restart
```

Concernant les fonctionnalités entreprise, si elles sont activées sur votre instance OpenProject, merci de vous référer à la section *Mise à jour* du chapitre *Fonctionnalités entreprise*.

Mise à jour pour des versions majeur :

Les versions majeur du logiciel sur le noyau Linux nécessite un changement dans le fichier de mise à jour des paquets.

Voici la ligne à change :

Modification du fichier :

```
sudo nano /etc/apt/sources.list.d/openproject.list
```

Changer $VERSION$ par la version majeur (e.g. .../stable/15/debian...)

```
deb https://dl.packager.io/srv/deb/opf/openproject/stable/$VERSION$/debian 12 main
```

Enregistrer et quitter : Après avoir modifié l'URL, appuyez sur `Ctrl + O` pour enregistrer le fichier, puis `Ctrl + X` pour quitter l'éditeur.

Ollama

Ollama est une plateforme open-source qui permet de créer, exécuter et partager des modèles de langage de grande taille localement sur des systèmes macOS et Linux1. Elle offre une interface en ligne de commande simple pour gérer ces modèles et inclut une bibliothèque de modèles pré-construits

Installation

Source : GitHub - ollama/ollama: Get up and running with Llama 3.2, Mistral, Gemma 2, and other large language models.

> **Version :** 0.4.5

Mise à jour du système :

```
sudo apt update && sudo apt upgrade -y && sudo apt full-upgrade -y && sudo apt autoremove -y && sudo apt clean
```

Installer CURL :

```
sudo apt-get install curl
```

Récupération de la source :

```
curl -fsSL https://ollama.com/install.sh | sh
```

Ajouter Ollama en service de démarrage :

Créez un utilisateur pour Ollama :

```
sudo useradd -r -s /bin/false -U -m -d /usr/share/ollama ollama
```

Ajoutez votre utilisateur actuel au groupe ollama :

```
sudo usermod -a -G ollama $(whoami)
```

Créez un fichier de service dans `/etc/systemd/system/ollama.service` :

```
[Unit]
Description=Ollama Service
After=network-online.target

[Service]
```

```
ExecStart=/usr/bin/ollama serve
User=ollama
Group=ollama
Restart=always
RestartSec=3
Environment="PATH=$PATH"

[Install]
WantedBy=default.target
```

Autre exemple de fichier :

```
[Unit]
Description=Ollama AI Server
After=network.target

[Service]
Environment="HOME=/root"
Environment="OLLAMA_HOST=0.0.0.0:11434"
ExecStart=/usr/local/bin/ollama serve
Restart=always
User=root

[Install]
WantedBy=multi-user.target
```

> **Attention** ! Pour la ligne `ExecStart=/usr/bin/ollama serve`, vous devez correctement localiser l'installation de votre ollama, exemple `ExecStart=/usr/local/bin/ollama serve`.

Ensuite, démarrez le service :

```
sudo systemctl daemon-reload
sudo systemctl enable ollama
```

Voir les logs :

```
journalctl -e -u ollama
```

Mise à jour

Pour mettre à jour Ollama, vous pouvez exécuter à nouveau la commande suivante :

```
curl -fsSL https://ollama.com/install.sh | sh
```

Cela réinstalle le script d'installation d'Ollama, et peut **écraser des fichiers existants** ou entraîner la perte de configurations personnalisées. Assurez-vous de **sauvegarder vos données et configurations importantes** avant de procéder à une mise à jour pour éviter toute perte de données.

Modèle personnalisé

Importation depuis GGUF

Ollama prend en charge l'importation de modèles au format **GGUF** via un fichier **Modelfile**.

Créer un fichier Modelfile :

Créez un fichier nommé `Modelfile`, qui doit contenir une instruction `FROM` avec le chemin local du modèle que vous souhaitez importer.

```
FROM ./vicuna-33b.Q4_0.gguf
```

Cela indique à Ollama de charger le modèle spécifié depuis le fichier local `vicuna-33b.Q4_0.gguf`.

Créer le modèle dans Ollama :

Une fois le fichier `Modelfile` créé, utilisez la commande suivante pour que le modèle soit intégré dans Ollama :

```
ollama create example -f Modelfile
```

Cette commande utilise le fichier `Modelfile` pour créer un modèle dans Ollama sous le nom "example".

Exécuter le modèle :

Enfin, vous pouvez exécuter le modèle avec la commande suivante :

```
ollama run example
```

Cela lance le modèle que vous venez de créer, ici appelé "example", pour qu'il commence à traiter des demandes ou des tâches.

Prompt personnalisé

Les modèles de la bibliothèque Ollama peuvent être personnalisés avec un **prompt**. Par exemple, pour personnaliser le modèle **llama3.2** :

Télécharger le modèle :

Tout d'abord, vous devez télécharger le modèle avec la commande suivante :

```
ollama pull llama3.2
```

Cela permet de récupérer le modèle **llama3.2** depuis la bibliothèque Ollama.

Créer un fichier Modelfile :

Ensuite, créez un fichier `Modelfile` contenant les instructions de personnalisation. Par exemple :

```
FROM llama3.2

# définir la température à 1 [plus élevé = plus créatif, plus bas = plus cohérent]
PARAMETER temperature 1

# définir le message système
SYSTEM """
You are Mario from Super Mario Bros. Answer as Mario, the assistant, only.
"""
```

- `FROM llama3.2` indique que vous souhaitez utiliser le modèle **llama3.2** comme base.
- La ligne `PARAMETER temperature 1` permet de régler la "température" du modèle, un paramètre qui influence la créativité du modèle. Une température élevée (comme 1) rend le modèle plus créatif, tandis qu'une température plus basse le rend plus cohérent.
- La section `SYSTEM` permet de personnaliser le message système, dans cet exemple, vous demandez au modèle de répondre en tant que **Mario**. uniquement.

Créer et exécuter le modèle :

Après avoir créé le fichier `Modelfile`, vous pouvez créer et exécuter le modèle personnalisé en utilisant ces commandes :

```
ollama create mario -f ./Modelfile
ollama run mario
```

La première commande crée un modèle nommé **Mario** à partir du fichier `Modelfile`, et la deuxième commande lance ce modèle.

Exemple d'interaction avec le modèle personnalisé :

```
>>> hi
Hello! It's your friend Mario.
```

Désinstallation

Supprimez le service Ollama :

```
sudo systemctl stop ollama
sudo systemctl disable ollama
sudo rm /etc/systemd/system/ollama.service
```

Supprimez le binaire `ollama` de votre répertoire `bin` (cela peut être `/usr/local/bin`, `/usr/bin`, ou `/bin`).

```
sudo rm $(which ollama)
```

Supprimez les modèles téléchargés ainsi que l'utilisateur et le groupe ollama créés pour le service :

```
sudo rm -r /usr/share/ollama
sudo userdel ollama
sudo groupdel ollama
```

Cela entraînera la **suppression définitive des fichiers d'Ollama**, y compris les modèles téléchargés et l'utilisateur associé. Assurez-vous de **sauvegarder vos données et configurations** avant de procéder à la désinstallation, car cette action est irréversible et pourrait supprimer des informations importantes.

Créer un modèle

La commande `ollama create` est utilisée pour créer un modèle à partir d'un fichier **Modelfile**.

```
ollama create mymodel -f ./Modelfile
```

Cette commande crée un modèle nommé **mymodel** en utilisant les instructions définies dans le fichier `Modelfile`.

Télécharger un modèle :

Pour télécharger un modèle, utilisez la commande suivante :

```
ollama pull llama3.2
```

Cette commande télécharge le modèle **llama3.2** depuis la bibliothèque Ollama. Elle peut également être utilisée pour **mettre à jour un modèle local**. Seules les différences (diff) entre le modèle local et la version la plus récente seront téléchargées.

Supprimer un modèle :

Si vous souhaitez supprimer un modèle, vous pouvez utiliser la commande suivante :

```
ollama rm llama3.2
```

Cela supprimera le modèle **llama3.2** de votre machine.

Copier un modèle :

Pour copier un modèle, utilisez la commande :

```
ollama cp llama3.2 my-model
```

Cela crée une copie du modèle **llama3.2** sous le nom **my-model**.

Entrée multilignes :

Pour entrer plusieurs lignes de texte, vous pouvez entourer le texte avec des guillemets triples (`"""`), comme suit :

```
>>> """Hello, ... world! ... """
```

Cela permet de saisir un texte réparti sur plusieurs lignes. Par exemple, cette entrée pourrait produire la sortie suivante :

```
I'm a basic program that prints the famous "Hello, world!" message to the console.
```

Modèles multimodaux :

Les modèles multimodaux permettent d'interagir avec des fichiers autres que du texte, comme des images. Par exemple, pour analyser une image avec un modèle multimodal, utilisez la commande :

```
ollama run llava "What's in this image? /Users/jmorgan/Desktop/smile.png"
```

Cela pourrait donner la réponse suivante :

```
The image features a yellow smiley face, which is likely the central focus of the picture.
```

Passer le prompt en argument :

Vous pouvez aussi passer un prompt directement en argument à la commande `ollama run`. Par exemple :

```
$ ollama run llama3.2 "Summarize this file: $(cat README.md)"
```

Cela permet de résumer le contenu d'un fichier, comme le fichier `README.md`. Le modèle peut générer un résumé tel que :

```
Ollama is a lightweight, extensible framework for building and running language models on the local machine. It provides a simple API for creating, running, and managing models, as well as a library of pre-built models that can be easily used in a variety of applications.
```

Afficher les informations sur un modèle :

Pour afficher les informations détaillées d'un modèle, utilisez la commande suivante :

```
ollama show llama3.2
```

Cela affiche des informations sur le modèle **llama3.2**, telles que sa version et ses paramètres.

Lister les modèles sur votre ordinateur :

Pour voir la liste de tous les modèles installés sur votre machine, utilisez cette commande :

```
ollama list
```

Lister les modèles actuellement chargés :

Pour voir quels modèles sont actuellement chargés en mémoire, utilisez la commande :

```
ollama ps
```

Arrêter un modèle en cours d'exécution :

Si vous souhaitez arrêter un modèle qui est en cours d'exécution, utilisez la commande suivante :

```
ollama stop llama3.2
```

Cela arrêtera le modèle **llama3.2** en cours d'exécution.

Démarrer Ollama :

Si vous voulez démarrer Ollama sans utiliser l'application de bureau, vous pouvez utiliser la commande suivante :

```
ollama serve
```

Cela démarre Ollama en mode serveur, permettant ainsi d'interagir avec les modèles via une API sans l'interface graphique.

Supprimer un modèle

Supprimer un modèle dans la list :

```
ollama rm llama3.2
```

Utilisation de l'API

? 1. Interroger l’API locale d’Ollama

Par défaut, l’API écoute sur :

```
http://localhost:11434
```

Exemple de requête API :

```
curl http://localhost:11434/api/generate \
  -d '{
        "model": "llama3",
        "prompt": "Bonjour, comment vas-tu ?"
      }'
```

? 2. Utilisation de l’API Ollama depuis Docker

Si vous exécutez des applications dans un **conteneur Docker** qui doivent appeler l’API Ollama située sur l’hôte, utilisez l’adresse suivante :

```
http://host.docker.internal:11434
```

Exemple depuis un conteneur :

```
curl http://host.docker.internal:11434/api/generate \
  -d '{"model": "llama3", "prompt": "Test depuis docker"}'
```

Remarque importante

- `host.docker.internal` fonctionne **automatiquement sur Docker Desktop (Windows / macOS)**.
- Sur Linux, Docker ne fournit **pas nativement** ce DNS.\ S'il n'existe pas, vous pouvez forcer la résolution en ajoutant dans votre commande `docker run` :

```
docker run --add-host=host.docker.internal:host-gateway ...
```

Ainsi, l'accès depuis Docker fonctionnera comme prévu.

? 3. Exemple simple depuis un script Python

```
import requests

payload = {
    "model": "llama3",
    "prompt": "Bonjour depuis Python !"
}

r = requests.post("http://localhost:11434/api/generate", json=payload)
print(r.text)
```

Open-WebUI

Interface IA conviviale (prend en charge Ollama, l’API OpenAI, ...)

Installation avec pip

Source : GitHub - open-webui/open-webui: User-friendly AI Interface (Supports Ollama, OpenAI API, ...)

ATTENTION !

L'installation avec `pip` rend la mise à jour du logiciel complexe. Faite cette installation pour des tests, ou du développement spécifique et non une utilisation en production.

Installation via Python pip

Open WebUI peut être installé via pip, l'installateur de paquets Python. Avant de procéder, assurez-vous d'utiliser Python 3.11 pour éviter les problèmes de compatibilité.

Installer Open WebUI :

Ouvrez votre terminal et exécutez la commande suivante pour installer Open WebUI :

```
pip install open-webui
```

Exécuter Open WebUI :

Une fois l'installation terminée, vous pouvez démarrer Open WebUI en exécutant la commande suivante :

```
open-webui serve
```

Cela démarrera le serveur Open WebUI, que vous pourrez accéder à l'adresse suivante : http://localhost:8080.

Erreur possible :

Erreur lors de l'installation de `open-webui` via `pip` :

```
root@srv-ia:/home/ia# pip install open-webui
error: externally-managed-environment

× This environment is externally managed
```

```
└─> To install Python packages system-wide, try apt install
    python3-xyz, where xyz is the package you are trying to
    install.

    If you wish to install a non-Debian-packaged Python package,
    create a virtual environment using python3 -m venv path/to/venv.
    Then use path/to/venv/bin/python and path/to/venv/bin/pip. Make
    sure you have python3-full installed.

    If you wish to install a non-Debian packaged Python application,
    it may be easiest to use pipx install xyz, which will manage a
    virtual environment for you. Make sure you have pipx installed.

    See /usr/share/doc/python3.11/README.venv for more information.

note: If you believe this is a mistake, please contact your Python installation or OS
distribution provider. You can override this, at the risk of breaking your Python installation
or OS, by passing --break-system-packages.
hint: See PEP 668 for the detailed specification.
```

Utiliser un environnement virtuel

La meilleure pratique pour installer des paquets Python sans affecter l'environnement système est de créer un **environnement virtuel**. Cela permet d'avoir une installation isolée pour vos paquets Python.

Étapes pour créer un environnement virtuel :

1. **Installez** `python3-venv` (si ce n'est pas déjà fait) :

   ```
   sudo apt install python3-venv
   ```

2. **Créez un environnement virtuel** :
   Dans le répertoire où vous souhaitez installer vos paquets, créez un environnement virtuel :

   ```
   python3 -m venv mon_venv
   ```

   Cette commande crée un dossier `mon_venv` qui contient une installation isolée de Python.
3. **Activez l'environnement virtuel** :

```
source mon_venv/bin/activate
```

Vous verrez probablement un changement dans l'invite de commande, indiquant que l'environnement virtuel est activé (par exemple, `(mon_venv)` au début de la ligne).

4. **Installez `open-webui` dans l'environnement virtuel** :
   Une fois l'environnement activé, vous pouvez installer `open-webui` comme d'habitude :

   ```
   pip install open-webui
   ```

5. **Lancez l'application avec `nohup` et redirigez la sortie vers un fichier de log, puis lancez-le en arrière-plan :**

   ```
   nohup open-webui serve > open-webui.log 2>&1 &
   ```

   - `nohup` permet au processus de ne pas être tué lorsqu'une session se termine.
   - `> open-webui.log 2>&1` redirige les sorties standard (stdout) et les erreurs (stderr) vers un fichier `open-webui.log`.
   - `&` place le processus en arrière-plan.

   Vous pouvez quitter la session SSH ou votre terminal normalement, l'application continuera de fonctionner.
   Pour voir si le processus est toujours en cours d'exécution :

   ```
   ps aux | grep open-webui
   ```

6. **Désactivez l'environnement virtuel** quand vous avez terminé :

```
deactivate
```

Cela vous ramènera à votre environnement système Python.

Installation avec docker

Avant de faire l'installation de Open-WebUI, merci de vous référer à l'installation de docker.

Installation avec la configuration par défaut

Si Ollama est sur votre ordinateur, utilisez cette commande :

```
docker run -d -p 3000:8080 --add-host=host.docker.internal:host-gateway -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:main
```

Si Ollama est sur un serveur différent, utilisez cette commande :

Pour vous connecter à Ollama sur un autre serveur, changez l'URL OLLAMA_BASE_URL par l'URL du serveur :

```
docker run -d -p 3000:8080 -e OLLAMA_BASE_URL=https://example.com -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:main
```

Pour exécuter Open WebUI avec le support GPU Nvidia, utilisez cette commande :

```
docker run -d -p 3000:8080 --gpus all --add-host=host.docker.internal:host-gateway -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:cuda
```

Installation pour une utilisation uniquement avec l'API OpenAI

Si vous utilisez uniquement l'API OpenAI, utilisez cette commande :

```
docker run -d -p 3000:8080 -e OPENAI_API_KEY=your_secret_key -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:main
```

Installation d'Open WebUI avec le support intégré d'Ollama
Cette méthode d'installation utilise une image de conteneur unique qui regroupe Open WebUI avec Ollama, permettant une configuration simplifiée avec une seule commande. Choisissez la commande appropriée en fonction de votre configuration matérielle :

Avec le support GPU : Utilisez les ressources GPU en exécutant la commande suivante :

```
docker run -d -p 3000:8080 --gpus=all -v ollama:/root/.ollama -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:ollama
```

Pour un usage uniquement avec le CPU : Si vous n'utilisez pas de GPU, utilisez cette commande à la place :

```
docker run -d -p 3000:8080 -v ollama:/root/.ollama -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:ollama
```

Les deux commandes facilitent une installation intégrée et sans tracas de Open WebUI et Ollama, garantissant une mise en place rapide de l'ensemble.

Après l'installation, vous pouvez accéder à Open WebUI à l'adresse http://localhost:3000. Profitez-en ! □□

Erreur de connexion Docker

Si vous rencontrez une erreur de connexion lorsque vous essayez d'accéder à Ollama, cela peut être dû au fait que le conteneur WebUI Docker ne peut pas communiquer avec le serveur Ollama qui fonctionne sur votre hôte. Voici comment résoudre cela :

Ajustez les paramètres réseau □□ : Utilisez le flag `--network=host` dans votre commande Docker. Cela relie directement votre conteneur au réseau de votre hôte.

Changez le port : N'oubliez pas que le port interne passe de 3000 à 8080.

Exemple de commande Docker :

```
docker run -d --network=host -v open-webui:/app/backend/data -e OLLAMA_BASE_URL=http://127.0.0.1:11434 --name open-webui --restart always ghcr.io/open-webui/open-webui:main
```

□□ Après avoir exécuté cette commande, votre WebUI devrait être disponible à l'adresse http://localhost:8080.

Mise à jour

Mise à jour vers Open WebUI tout en conservant vos données

Si vous souhaitez mettre à jour vers la nouvelle image tout en migrant tous vos paramètres précédents tels que les conversations, les invites, les documents, etc.

Vous pouvez effectuer les étapes suivantes :

```
docker rm -f ollama-webui
docker pull ghcr.io/open-webui/open-webui:main
# Crée un nouveau volume et utilise un conteneur temporaire pour copier d'un volume à un autre
selon https://github.com/moby/moby/issues/31154#issuecomment-360531460
docker volume create --name open-webui
docker run --rm -v ollama-webui:/from -v open-webui:/to alpine ash -c "cd /from ; cp -av .
/to"
[insérez la commande équivalente que vous avez utilisée pour installer avec le nouveau nom de
l'image Docker]
```

Une fois que vous avez vérifié que toutes les données ont été migrées, vous pouvez supprimer l'ancien volume avec la commande suivante :

```
docker volume rm ollama-webui
```

Passage depuis un dépôt Git local
Si vous veniez d'une installation Git où vous avez utilisé `docker compose up` dans le répertoire du projet, vos volumes auront un préfixe avec le nom du dossier. Par conséquent, si votre chemin OpenWebUI était : /home/myserver/ollama-webui/, les volumes seraient nommés "ollama-webui_open-webui" et "ollama-webui_ollama".

Pour copier le contenu vers une installation Docker conventionnelle, vous pouvez exécuter des commandes de migration similaires. **Dans notre cas, les commandes seraient les suivantes :**

```
docker rm -f open-webui
docker rm -f ollama
docker pull ghcr.io/open-webui/open-webui:main
docker pull ghcr.io/open-webui/open-webui:ollama
docker volume create --name open-webui
docker volume create --name ollama
```

```
docker run --rm -v ollama-webui_open-webui:/from -v open-webui:/to alpine ash -c "cd /from ; cp -av . /to"
docker run --rm -v ollama-webui_ollama:/from -v ollama:/to alpine ash -c "cd /from ; cp -av . /to"
```

Selon que vous ayez installé Ollama ou que vous ayez déjà les mêmes noms de volumes en place, certaines des commandes pourraient générer des erreurs, mais elles peuvent généralement être ignorées en toute sécurité car nous procédons à un écrasement.

Ensuite, lancez les deux conteneurs comme d'habitude, comme décrit dans le guide de démarrage :

```
docker run -d -p 3000:8080 --add-host=host.docker.internal:host-gateway -v open-webui:/app/backend/data --name open-webui --restart always ghcr.io/open-webui/open-webui:main
```

Une fois que vous avez vérifié que toutes les données ont été migrées, vous pouvez supprimer l'ancien volume avec la commande `docker volume rm` mentionnée ci-dessus.

Docker

Docker est une plateforme permettant de lancer certaines applications dans des conteneurs logiciels lancée en 2013.

Installation (debian)

Mettre à jour votre système

Avant d'installer Docker, il est recommandé de mettre à jour votre système Debian pour s'assurer que tous les paquets sont à jour.

```
sudo apt update && sudo apt upgrade -y
```

Installer les dépendances nécessaires

Docker nécessite quelques outils comme `apt-transport-https`, `ca-certificates`, `curl`, `software-properties-common` et `gnupg2`. Installez-les en utilisant la commande suivante :

```
sudo apt install apt-transport-https ca-certificates curl software-properties-common gnupg2 -y
```

Ajouter la clé GPG officielle de Docker

Docker signe ses paquets avec une clé GPG pour garantir leur authenticité. Vous devez ajouter cette clé pour pouvoir installer Docker depuis les dépôts officiels.

```
curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --dearmor -o
/usr/share/keyrings/docker-archive-keyring.gpg
```

Ajouter le dépôt Docker

Ensuite, vous devez ajouter le dépôt officiel de Docker à votre liste de sources APT. Exécutez cette commande pour ajouter le dépôt stable :

```
echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg]
https://download.docker.com/linux/debian $(lsb_release -cs) stable" | sudo tee
/etc/apt/sources.list.d/docker.list > /dev/null
```

Mettre à jour les sources

Une fois le dépôt Docker ajouté, vous devez mettre à jour la liste des paquets disponibles pour que le système reconnaisse les paquets Docker :

```
sudo apt update
```

Installer Docker

Vous pouvez maintenant installer Docker. Utilisez la commande suivante pour installer Docker CE (Community Edition) :

```
sudo apt install docker-ce docker-ce-cli containerd.io -y
```

Vérifier l'installation

Une fois Docker installé, vérifiez qu'il fonctionne correctement en exécutant la commande suivante pour voir la version de Docker installée :

```
docker --version
```

Démarrer et activer le service Docker

Par défaut, le service Docker devrait démarrer automatiquement après l'installation. Si ce n'est pas le cas, vous pouvez le démarrer manuellement et l'activer pour qu'il démarre automatiquement au démarrage de votre VM :

```
sudo systemctl start docker
sudo systemctl enable docker
```

Vérifier que Docker fonctionne correctement

Pour vérifier que Docker fonctionne correctement, exécutez la commande suivante, qui va télécharger et exécuter un conteneur test :

```
sudo docker run hello-world
```

Si tout est correctement installé, vous devriez voir un message vous indiquant que Docker fonctionne correctement.

Ajouter votre utilisateur au groupe Docker (facultatif)

Si vous souhaitez exécuter des commandes Docker sans avoir à utiliser `sudo` à chaque fois, vous pouvez ajouter votre utilisateur au groupe `docker` :

```
sudo usermod -aG docker $USER
```

Pour que ce changement prenne effet, vous devez vous déconnecter puis vous reconnecter à votre session ou exécuter la commande suivante :

```
newgrp docker
```

Mise à jour (debian)

Mettre à jour la liste des dépôts

La première étape consiste à mettre à jour la liste des dépôts pour s'assurer que vous avez les dernières informations sur les paquets disponibles.

```
sudo apt update
```

Vérifier les versions disponibles

Avant de mettre à jour Docker, vous pouvez vérifier si une nouvelle version est disponible. Pour cela, utilisez la commande suivante :

```
apt list --upgradable
```

Cela affichera la liste des paquets pouvant être mis à jour, y compris Docker si une nouvelle version est disponible.

Mettre à jour Docker

Si Docker est inclus dans la liste des paquets à mettre à jour, vous pouvez procéder à la mise à jour avec cette commande :

```
sudo apt upgrade docker-ce docker-ce-cli containerd.io
```

Cela mettra à jour Docker et ses composants principaux.

(Facultatif) Mettre à jour tous les paquets du système

Si vous voulez mettre à jour l'ensemble des paquets de votre système, y compris Docker et d'autres dépendances, utilisez la commande suivante :

```
sudo apt upgrade -y
```

Cela mettra à jour tous les paquets installés, y compris Docker, si une nouvelle version est disponible dans les dépôts.

Vérifier la version de Docker

Une fois la mise à jour terminée, vous pouvez vérifier que Docker a bien été mis à jour en exécutant :

```
docker --version
```

Cela vous affichera la version actuelle de Docker installée.

Redémarrer le service Docker (si nécessaire)

Bien que Docker mette généralement à jour ses composants sans redémarrage, il est parfois nécessaire de redémarrer le service Docker pour que les modifications prennent effet correctement :

```
sudo systemctl restart docker
```

Vérifier le fonctionnement de Docker

Pour vous assurer que Docker fonctionne correctement après la mise à jour, vous pouvez exécuter un conteneur de test :

```
sudo docker run hello-world
```

Si Docker fonctionne correctement, le message "Hello from Docker!" s'affichera.

Nettoyage (Facultatif)

Enfin, pour garder votre système propre, vous pouvez supprimer les anciennes versions de paquets qui ne sont plus nécessaires en exécutant :

```
sudo apt autoremove
```

Cela supprimera les dépendances inutiles et les anciens paquets qui ne sont plus utilisés.

Docker

Purge

Cette procédure permet de supprimer l'intégralité des données de Docker (conteneurs, images, volumes, réseaux et cache) pour retrouver un environnement "vierge", tout en conservant l'installation de Docker Engine.

---

1. Arrêt et suppression des conteneurs

La première étape consiste à stopper tous les processus en cours pour libérer les ressources.

- **Arrêter tous les conteneurs :**

```
docker stop $(docker ps -aq)
```

- **Supprimer tous les conteneurs :**

```
docker rm $(docker ps -aq)
```

---

2. Suppression des images

Pour libérer l'espace disque occupé par les systèmes de fichiers superposés.

- **Supprimer toutes les images (utilisées ou non) :**

```
docker rmi -f $(docker images -aq)
```

---

3. Suppression des volumes et réseaux

Les volumes contiennent vos données persistantes et les réseaux vos configurations de ponts personnalisées.

- **Supprimer tous les volumes :**

```
docker volume rm $(docker volume ls -q)
```

- **Supprimer les réseaux personnalisés :** *(Note : Les réseaux par défaut comme '*bridge*', '*host*' et '*none*' ne seront pas supprimés)*

```
docker network rm $(docker network ls -q)
```

---

4. Nettoyage du cache de construction (Build Kit)

Docker conserve des traces des étapes de construction pour accélérer les futurs `docker build`. Pour une purge totale, il faut les vider.

- **Supprimer le cache de build :**

```
docker builder prune -a -f
```

---

5. La commande "All-in-One" (Le Grand Ménage)

Si vous souhaitez exécuter l'intégralité de ces étapes en une seule ligne de commande sécurisée (elle vérifie d'abord s'il y a des éléments à supprimer pour éviter les erreurs de syntaxe), utilisez celle-ci :

```
docker stop $(docker ps -aq) 2>/dev/null; docker system prune -a --volumes -f
```

Détail de la commande All-in-One :

- `docker stop ...` : Arrête proprement tous les conteneurs.
- `2>/dev/null` : Masque les erreurs si aucun conteneur n'est en cours.
- `docker system prune` : La commande native de Docker pour le nettoyage.

- `-a` : Supprime toutes les images inutilisées, pas seulement les "dangling" (sans nom).
- `--volumes` : Force la suppression des volumes (non inclus par défaut dans le prune).
- `-f` : Force l'exécution sans demander de confirmation (mode automatique).

> **Attention :** Cette opération est irréversible. Toutes les données contenues dans les bases de données (volumes) et les configurations spécifiques seront perdues.

Debian

Debian est un système d’exploitation libre et gratuit, principalement basé sur Linux. Créé en 1993, il est connu pour sa stabilité, sa sécurité et son large choix de logiciels. Debian supporte de nombreuses architectures matérielles et sert souvent de base à d’autres distributions comme Ubuntu.

Debian

Commandes communes

Utiliser la commande `ss`

tag : <voir port>

La commande `ss` est un outil moderne et rapide pour afficher les connexions réseau. Vous pouvez l'utiliser pour lister les ports ouverts.

Pour afficher tous les ports ouverts et les services qui les utilisent :

```
sudo ss -tuln
```

- `-t` : Affiche les connexions TCP.
- `-u` : Affiche les connexions UDP.
- `-l` : Affiche les ports à l'écoute (Listening).
- `-n` : Affiche les adresses et les ports sous forme numérique (sans résolution de noms).

Exemple de sortie :

```
Netid  State      Recv-Q Send-Q Local Address:Port   Peer Address:Port
tcp    LISTEN     0      128          *:22              *:*
tcp    LISTEN     0      128       127.0.0.1:631        *:*
udp    UNCONN     0      0        0.0.0.0:123           *:*
```

Cela vous donne une vue d'ensemble des ports ouverts et des adresses locales auxquelles ils sont liés.

Exemple pour un port spécifique (par exemple, le port 8080) :

```
sudo ss -tuln | grep ':8080'
```

Cette commande affiche les connexions TCP et UDP en écoute et filtre celles qui utilisent le port 8080.

Si vous souhaitez également afficher le PID, ajoutez l'option `-p` :

```
sudo ss -tulnp | grep ':8080'
```

Utiliser la commande `ip -c a`

tag : <voir ip, voir adresse ip>

`ip -c a` (ou **`ip -c addr`**) affiche toutes les interfaces réseau et leurs informations, avec une sortie colorée pour améliorer la lisibilité.

```
ip -c a
```

Exemple de sortie de `ip -c a` :

Voici un exemple de ce à quoi pourrait ressembler la sortie d'une commande `ip -c a` :

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever

2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen
1000
    inet 192.168.1.10/24 brd 192.168.1.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::f816:3eff:fe2a:0bfa/64 scope link
       valid_lft forever preferred_lft forever
```

- **`lo`** est l'interface de boucle locale (loopback), avec l'adresse `127.0.0.1`.
- **`eth0`** est une interface Ethernet avec une adresse IPv4 (`192.168.1.10`) et une adresse IPv6 (`fe80::...`).

Avec l'option `-c`, ces informations sont colorées pour les rendre plus faciles à analyser visuellement.

En résumé :

- **`ip`** : Utilitaire pour gérer les interfaces réseau, les routes, les adresses IP, etc.
- **`-c`** : Active la coloration de la sortie.
- **`a`** : Affiche les adresses réseau et les interfaces.

La commande `ip -c a` est donc un moyen simple et rapide de visualiser l'état des interfaces réseau de votre système avec une sortie colorée pour une lecture plus facile.

Debian

Mise à jour

? Commande sans risque (mise à jour des index uniquement)

```
sudo apt update
```

> Met à jour la liste des paquets disponibles sans installer quoi que ce soit. Aucun changement sur le système.

?? Commande plus complète (mise à jour des paquets installés)

```
sudo apt update && sudo apt upgrade -y
```

> Met à jour la liste des paquets **et installe les mises à jour disponibles** pour les paquets déjà installés. Le `-y` permet de valider automatiquement les changements.

? Commande ultra complète (mise à jour + nettoyage)

```
sudo apt update && sudo apt full-upgrade -y && sudo apt autoremove --purge -
y && sudo apt autoclean
```

> En plus de la mise à jour : `full-upgrade` gère les changements de dépendances (peut installer ou supprimer des paquets). `autoremove --purge` supprime les paquets inutiles et leurs fichiers de config. `autoclean` nettoie les fichiers .deb obsolètes.

Debian

Étendre une partition sur Debian (MBR) sans perte de données

Ce guide explique comment étendre une partition / (root) sous Debian lorsque le disque virtuel a été agrandi dans l'hyperviseur, mais que la partition principale est bloquée par une partition étendue contenant le swap (`sda2` + `sda5`).

Fonctionne pour :

- VMware, Proxmox, Hyper-V, VirtualBox
- Debian 10 → Debian 13+
- Disques MBR (label `dos`)

? Avant de commencer

Vérifier que le disque virtuel a déjà été agrandi dans l'hyperviseur :

```
lsblk
```

Vous devez voir par ex. :

sda 50G └─sda1 23G

1?? Désactiver le swap existant

```
swapoff -a
```

2?? Modifier la table de partitions avec fdisk

Lancer `fdisk` :

```
fdisk /dev/sda
```

Dans `fdisk`, effectuer exactement les actions suivantes :

1. Supprimer l’ancienne swap (sda5)

```
d
5
```

2. Supprimer la partition étendue (sda2)

```
d
2
```

3. Supprimer la partition root (sda1)

⚠ Sans danger tant que le secteur de début est conservé.

```
d
1
```

4. Recréer /dev/sda1 sur tout le disque

```
n
p
1
2048 ← le même secteur de début qu’avant
<Entrée> ← fin par défaut = fin du disque
```

5. Remettre le drapeau boot

```
a
1
```

6. Sauvegarder et quitter

```
w
```

Un redémarrage peut être nécessaire :

```
reboot
```

3?? Étendre le système de fichiers ext4

Une fois la machine revenue :

```
resize2fs /dev/sda1
```

Le système de fichiers s’adapte alors à la nouvelle taille (50 Go dans cet exemple).

Vérifier :

```
df -h
```

4?? Créer un swap via un fichier (recommandé)

Un swapfile est plus simple et plus flexible qu’une partition swap.

Création du fichier swap :

```
fallocate -l 1G /swapfile && chmod 600 /swapfile && mkswap /swapfile && swapon /swapfile
```

Rendre le swap permanent

Ajouter dans `/etc/fstab` :

```
echo '/swapfile none swap sw 0 0' >> /etc/fstab
```

ou

```
/swapfile none swap sw 0 0
```

5?? Vérifications finales

```
lsblk && df -h && free -h
```

Vous devriez voir :

- `/dev/sda1` = ~50 Go monté sur `/`
- `/swapfile` actif

? Résultat final

Votre système Debian utilise maintenant toute la taille du disque agrandi, sans perte de données, avec un swap propre et moderne via fichier.

? Bonus : résolution du problème “commandes sbin introuvables”

Si des commandes comme `fdisk`, `parted`, `ip`, `reboot`... n’étaient plus accessibles :

Ajouter ceci dans `/root/.bashrc` :

```
export PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

Puis :

```
source /root/.bashrc
```

AMP

L'outil **AMP** (Application Management Panel) par **CubeCoders** est un panneau de contrôle web auto-hébergé, conçu principalement pour la gestion de serveurs de jeux

Installation

Proxmox

Proxmox est une plateforme de **virtualisation open-source** qui permet de gérer des environnements virtuels.

Proxmox

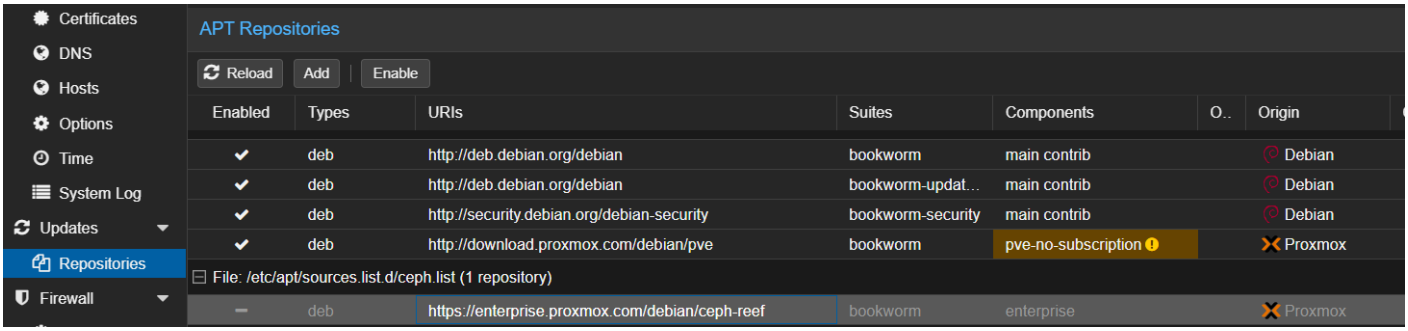
Update package database error

? Erreur rencontrée :

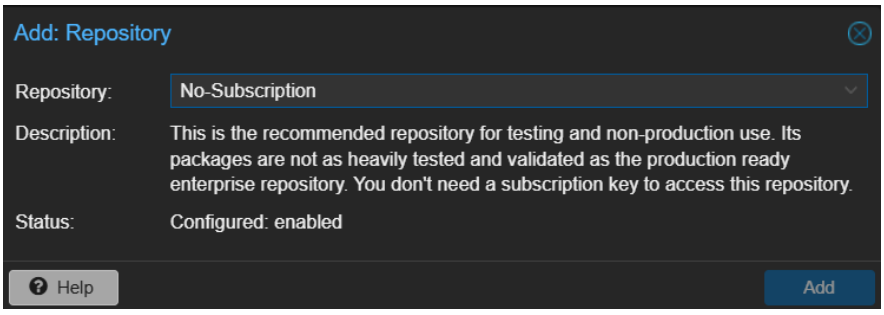
Code d’erreur : `TASK ERROR: command 'apt-get update' failed: exit code 100`

? Solution :

- Sélectionnez l'hôte concerné.
- Allez dans l'onglet **Mises à jour** puis **Dépôts**.
- Repérez le dépôt **Enterprise** et **désactivez-le**.
- Si un second dépôt **Enterprise** est présent, désactivez-le également.



Cliquez sur **Ajouter**, puis changez le dépôt en **"No Subscription"**, et ajoutez-le.



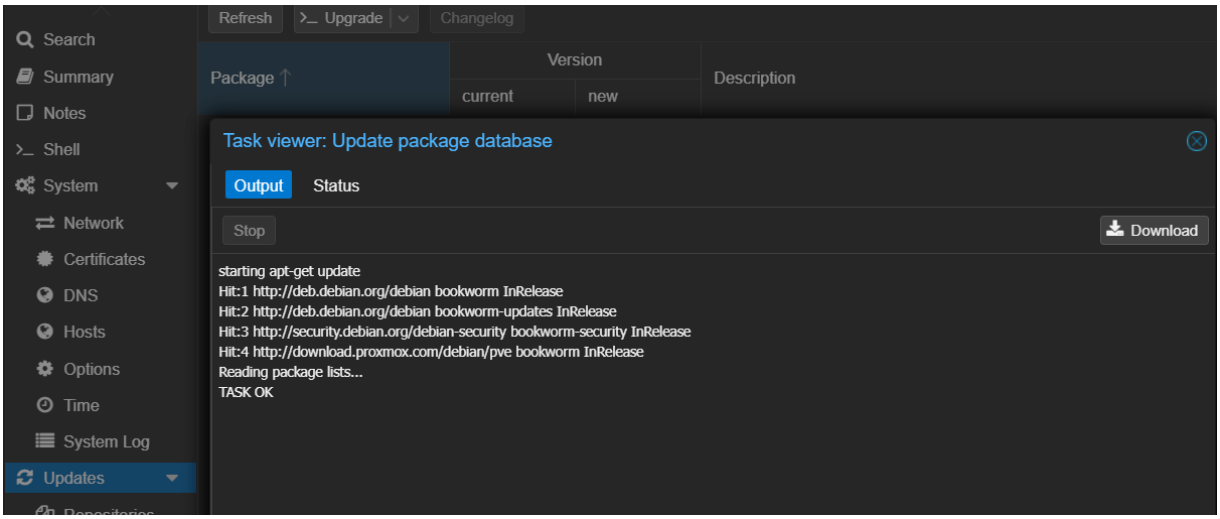
? Vérification :

- Vous devriez voir une **coche verte** et le message :
  "*You get updates for Proxmox VE*".

? Finalisation :

- Retournez dans l’onglet **Mises à jour**.
- Cliquez sur **Actualiser**.

- La tâche devrait maintenant se compléter **sans erreur**.



Helper-script

> Source : https://community-scripts.github.io/ProxmoxVE/

Proxmox VE Helper-Scripts :

- **Collection d'outils** : Ces scripts sont conçus pour simplifier la configuration et la gestion de l'environnement Proxmox Virtual Environment (VE).
- **Plus de 300 scripts** : Le dépôt communautaire propose une vaste gamme de scripts pour aider à gérer les machines virtuelles, les conteneurs, les clusters hautement disponibles, le stockage et les réseaux.
- **Interface conviviale** : Les utilisateurs peuvent créer des conteneurs Linux ou des machines virtuelles de manière interactive, avec des options pour des configurations simples ou avancées.

Ces outils sont très utiles pour optimiser l'expérience de gestion de Proxmox VE.

Reverse proxy avec VPS frontend

? Objectif

- Utiliser un VPS OVH comme passerelle d'accès
- Tunnel des services locaux via FRP (Fast Reverse Proxy)
- Reverse proxy HTTPS avec Caddy sur le VPS
- Accès via des sous-domaines personnalisés

? Prérequis

- Un VPS avec Ubuntu/Debian
- Un nom de domaine chez OVH avec DNS pointés vers le VPS
- Un serveur local (Proxmox) avec :
  - APP1 sur 192.168.X.X:8080
  - APP2 sur 192.168.X.X:8080/exemple
  - PROXMOX sur 192.168.X.X:8662

> Différents cas d'usage sont donnés ici. APP1 étant une application classique sans configuration particulière. APP2 est une application utilisant une redirection vers un lien spécifique, ici le dossier "exemple". Et enfin une configuration adapté pour PROXMOX.

? Partie 1 : Installer FRP

Sur le VPS (serveur FRP)

```
cd /opt
wget https://github.com/fatedier/frp/releases/download/v0.58.0/frp_0.58.0_linux_amd64.tar.gz
tar -xzf frp_0.58.0_linux_amd64.tar.gz
mv frp_0.58.0_linux_amd64 frp
nano /opt/frp/frps.ini
```

Contenu :

```
[common]
bind_port = 6000
```

Lancer manuellement (ou voir plus bas pour le service) :

```
/opt/frp/frps -c /opt/frp/frps.ini
```

Créer le service systemd :

```
sudo nano /etc/systemd/system/frps.service
```

Contenu :

```
[Unit]
Description=FRP Server
After=network.target

[Service]
ExecStart=/opt/frp/frps -c /opt/frp/frps.ini
Restart=always
User=user
LimitNOFILE=65535

[Install]
WantedBy=multi-user.target
```

Puis :

```
sudo systemctl daemon-reexec
sudo systemctl enable --now frps
```

Sur le serveur local (client FRP)

```
cd /opt
wget https://github.com/fatedier/frp/releases/download/v0.58.0/frp_0.58.0_linux_amd64.tar.gz
tar -xzf frp_0.58.0_linux_amd64.tar.gz
mv frp_0.58.0_linux_amd64 frp
nano /opt/frp/frpc.ini
```

Contenu :

```
[common]
server_addr = IP_DU_VPS
server_port = 6000

[APP1]
type = tcp
local_ip = 192.186.X.X
local_port = 8080
remote_port = 9000

[APP2]
type = tcp
local_ip = 192.168.X.X
local_port = 8080
remote_port = 9001

[PROXMOX]
type = tcp
local_ip = 192.168.X.X
local_port = 8662
remote_port = 9002
```

> Si vous réalisez une modification/ajout dans le fichier ci-dessus, pensez bien à redémarrer FRPC `systemctl restart frpc`

Créer le service FRP client :

```
sudo nano /etc/systemd/system/frpc.service
```

Contenu :

```
[Unit]
Description=FRP Client
After=network.target

[Service]
ExecStart=/opt/frp/frpc -c /opt/frp/frpc.ini
Restart=always
User=user
LimitNOFILE=65535
```

```
[Install]
WantedBy=multi-user.target
```

Et activer :

```
sudo systemctl daemon-reexec
sudo systemctl enable --now frpc
```

? Partie 2 : Installer Caddy sur le VPS

```
sudo apt install -y debian-keyring debian-archive-keyring curl
curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/gpg.key' | sudo gpg --dearmor -o
/usr/share/keyrings/caddy.gpg
curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/debian.deb.txt' | sudo tee
/etc/apt/sources.list.d/caddy.list
sudo apt update
sudo apt install caddy
```

?? Partie 3 : Configuration du Caddyfile (côté VPS)

```
sudo nano /etc/caddy/Caddyfile
```

Contenu recommandé :

> Ce contenu comporte différents types de configuration. Prenez le plus adapté à votre cas d'usage.

```
proxmox.domaine.com {
    reverse_proxy https://localhost:9000 {
        transport http {
            tls_insecure_skip_verify
            versions h1
        }
        header_up Host {host}
        header_up X-Real-IP {http.request.remote}
        header_up X-Forwarded-For {http.request.remote}
        header_up X-Forwarded-Proto {http.request.scheme}
    }
```

```
}

app1.domaine.com {
    reverse_proxy localhost:6001
}

app2.domaine.com {
    @exemple_path path /*
    rewrite @exemple_path /exemple{path}

    reverse_proxy localhost:9002 {
        transport http {
            versions h1
        }
        header_up Host {host}
        header_up X-Forwarded-Proto https
    }
}
```

> Si vous réalisez une modification/ajout dans le fichier ci-dessus, pensez bien à redémarrer CADDY `systemctl reload caddy`

Puis :

```
sudo systemctl reload caddy
```

? Test des services

- Proxmox : https://proxmox.domaine.com
- APP1 : https://app1.domaine.com
- APP2 : https://app2.domaine.com (mais affiché sans /exemple)

Tester en navigation privée pour éviter les cookies HTTPS précédents.

? Et voilà !

Tu as maintenant une exposition sécurisée de tes services locaux via un VPS, sans exposer ton IP personnelle, avec HTTPS et Caddy.

Apache Guacamole

Apache Guacamole est une passerelle de bureau à distance qui permet d'accéder à des machines distantes sans avoir besoin d'installer de client.

Mise en place d'une connexion SSH via clé RSA

? Prérequis

- Un serveur **Apache Guacamole** opérationnel
- Une machine cible avec un serveur SSH actif
- Une **clé RSA** (privée + publique)
- Un utilisateur SSH valide sur la machine cible

?? 1. Génération d’une clé RSA

Sur la machine qui initie la connexion (par ex. Guacamole) :

```
ssh-keygen -t rsa -b 4096 -m PEM -f ~/.ssh/guac_key
```

 Option `-m PEM` nécessaire : Guacamole n'accepte pas le format OpenSSH.

- `guac_key` : clé **privée** (à coller dans Guacamole)
- `guac_key.pub` : clé **publique** (à mettre sur la machine cible)
- **Ne mettez pas de passphrase** pour éviter des problèmes

? 2. Déploiement de la clé publique

Sur le serveur (où se trouve sshd) :

```
mkdir -p ~/.ssh
chmod 700 ~/.ssh

cat guac_key.pub >> ~/.ssh/authorized_keys
chmod 600 ~/.ssh/authorized_keys
```

?? 3. Ajustement du serveur SSH (si nécessaire)

Certaines distros récentes désactivent `ssh-rsa` par défaut.

Modifier `/etc/ssh/sshd_config` et ajouter :

```
PubkeyAcceptedAlgorithms +ssh-rsa
HostKeyAlgorithms +ssh-rsa
```

Puis redémarrer le service SSH :

```
sudo systemctl restart sshd
```

?? 4. Configuration dans l'interface Guacamole

1. Aller dans **Connexions → Nouvelle connexion**
2. Choisir le **protocole SSH**
3. Remplir les champs suivants :
   - **Nom d'hôte** : IP ou domaine de la cible
   - **Port** : généralement `22`
   - **Nom d'utilisateur** : ex. `root`
   - **Clé privée** : coller le contenu du fichier `guac_key`
   - **Phrase de passe** : laisser vide (si clé sans mot de passe)

? 5. Tester la connexion

Cliquez sur la connexion SSH : si tout est bien configuré, le terminal s'ouvre depuis le navigateur.

? Dépannage : "Public key authentication failed"

Si l'erreur suivante apparaît dans les logs `guacd` :

```
Public key authentication failed: Username/PublicKey combination invalid
```

Vérifications à faire :

- Clé privée au format **PEM**
- Clé publique bien dans `~/.ssh/authorized_keys`
- Fichier `sshd_config` contient :

```
PubkeyAcceptedAlgorithms +ssh-rsa
HostKeyAlgorithms +ssh-rsa
```

- Permissions correctes :
  - `~/.ssh` → `700`
  - `~/.ssh/authorized_keys` → `600`

? Astuce

Pour suivre les logs en direct :

```
journalctl -u guacd -f
```

>

Source : Documentation Guacamole Manuel OpenSSH

Grafana

Grafana est une plateforme de visualisation de données **open source** lancée en **2013**. Elle permet aux utilisateurs de :

- **Créer des tableaux de bord dynamiques** et modulables.
- **Visualiser** des métriques, des journaux et des traces provenant de diverses sources de données.
- **Configurer des alertes** pour surveiller les performances et la santé des systèmes.

Mise en place de node-exporter

Mise en place de Node Exporter pour la supervision avec Prometheus et Grafana

Ce guide explique comment installer et configurer **Node Exporter** sur une machine (VPS ou serveur Proxmox) afin d'exposer ses métriques à un serveur **Prometheus**, qui les affichera ensuite dans **Grafana**.

? Objectif

Obtenir les statistiques système de vos machines :

- CPU, RAM, disques, réseau
- Température, uptime, charge

? Prérequis

- Un serveur Prometheus fonctionnel
- Une machine cible sur laquelle on souhaite installer Node Exporter (ex : VPS, Proxmox Host)

?? Installation de Node Exporter

1. Se connecter à la machine cible (ex : via SSH)

2. Télécharger Node Exporter

```
curl -LO https://github.com/prometheus/node_exporter/releases/latest/download/node_exporter-1.8.1.linux-amd64.tar.gz
tar -xvzf node_exporter-1.8.1.linux-amd64.tar.gz
cd node_exporter-1.8.1.linux-amd64
```

> Si la commande `curl -LO` ne fonctionne pas, utiliser `wget`. Une commande fonctionnelle est présente ci-dessous. Merci de simplement modifier la version de l'exemple (1.8.1) vers la dernière version.

>

> `wget https://github.com/prometheus/node_exporter/releases/download/v1.8.1/node_exporter-1.8.1.linux-amd64.tar.gz`

3. Installer le binaire

```
sudo cp node_exporter /usr/local/bin/
```

4. Créer un utilisateur spécial pour exécuter Node Exporter

```
sudo useradd -rs /bin/false node_exporter
```

5. Créer le service systemd

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

[Service]
User=node_exporter
ExecStart=/usr/local/bin/node_exporter

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

6. Activer et démarrer le service

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
sudo systemctl enable --now node_exporter
```

7. Vérifier qu'il fonctionne

```
curl http://localhost:9100/metrics
```

Vous devez voir une longue liste de métriques Prometheus.

? Configuration de Prometheus (sur le serveur Prometheus)

1. Modifier le fichier `prometheus.yml`

Ajoutez la cible dans la section `scrape_configs` :

```
- job_name: 'nom_de_la_machine'
    static_configs:
      - targets: ['IP_DE_LA_MACHINE:9100']
```

2. Redémarrer Prometheus

```
sudo systemctl restart prometheus
```

3. Vérifier l'interface Web de Prometheus

Naviguez vers `http://IP_DU_SERVEUR_PROMETHEUS:9090/targets` Vous devez voir la cible apparaître en **UP**.

? Affichage dans Grafana

1. Aller dans Grafana > Configuration > Data Sources

- Ajouter ou sélectionner la source **Prometheus**

2. Aller dans Dashboards > Import

- Entrer l'ID : `1860`
- Sélectionner la source Prometheus
- Importer le dashboard **Node Exporter Full**

3. Profiter des statistiques temps réel □□

?? Sécurité (optionnel)

- Pour accéder à Node Exporter via Internet, utilisez un tunnel WireGuard, FRP ou un reverse proxy avec auth.
- Sinon, limitez l'accès au port 9100 au seul serveur Prometheus via `iptables` ou `ufw`.

? Notes finales

- Node Exporter ne collecte **pas de logs**. Pour les logs (fail2ban, journald...), utilisez **Promtail** avec **Loki**.
- Vous pouvez installer Node Exporter sur **autant de machines que nécessaire**, Prometheus gèrera toutes les cibles.

C'est tout ! Vous avez maintenant un système de monitoring robuste avec Node Exporter + Prometheus + Grafana.

Grafana

Mise en place de prometheus

Mise en place de Prometheus pour la supervision

Ce guide explique comment installer et configurer **Prometheus** sur une machine dédiée (VM ou LXC), afin de collecter les métriques d'autres machines (VPS, Proxmox, etc.) via des agents comme **node_exporter**.

> Vous pouvez aussi utilisé Helper-Script pour l'installation :
>
> **Prometheus PVE Exporter :** https://community-scripts.github.io/ProxmoxVE/scripts?id=prometheus-pve-exporter
>
> **Prometheus :** https://community-scripts.github.io/ProxmoxVE/scripts?id=prometheus

? Objectif

Mettre en place un serveur **Prometheus** pour collecter des métriques depuis plusieurs machines :

- VPS avec node_exporter
- Proxmox exporter (exporteur dédié pour Proxmox VE)
- Machines locales avec node_exporter

? Prérequis

- Une VM ou LXC dédiée à Prometheus (Debian/Ubuntu recommandé)
- Accès root ou sudo

?? Installation de Prometheus

1. Mise à jour et création d'utilisateur

```
sudo apt update && sudo apt install -y curl tar
sudo useradd --no-create-home --shell /bin/false prometheus
```

2. Téléchargement

```
cd /tmp
curl -LO https://github.com/prometheus/prometheus/releases/latest/download/prometheus-2.52.0.linux-amd64.tar.gz
```

```
tar xvf prometheus-2.52.0.linux-amd64.tar.gz
cd prometheus-2.52.0.linux-amd64
```

> Si la commande `curl -LO` ne fonctionne pas, utilisé `wget`

3. Déplacement des fichiers

```
sudo mkdir -p /etc/prometheus /var/lib/prometheus
sudo cp prometheus promtool /usr/local/bin/
sudo cp -r consoles/ console_libraries/ /etc/prometheus/
sudo cp prometheus.yml /etc/prometheus/
```

4. Droits

```
sudo chown -R prometheus:prometheus /etc/prometheus /var/lib/prometheus
sudo chown prometheus:prometheus /usr/local/bin/prometheus /usr/local/bin/promtool
```

? Création du service systemd

```
sudo tee /etc/systemd/system/prometheus.service > /dev/null <<EOF
[Unit]
Description=Prometheus
Wants=network-online.target
After=network-online.target

[Service]
User=prometheus
Group=prometheus
Type=simple
ExecStart=/usr/local/bin/prometheus \
  --config.file=/etc/prometheus/prometheus.yml \
  --storage.tsdb.path=/var/lib/prometheus/

[Install]
WantedBy=multi-user.target
EOF
```

?? Configuration de `/etc/prometheus/prometheus.yml`

Voici un exemple **valide** pour plusieurs machines :

```
global:
  scrape_interval: 15s

scrape_configs:
  - job_name: 'nom_de_la_machine'
    static_configs:
      - targets: ['IP_DE_VOTRE_MACHINE:9090']

  - job_name: 'nom_de_la_machine'
    static_configs:
      - targets: ['IP_DE_VOTRE_MACHINE:9221']

  - job_name: 'nom_de_la_machine'
    static_configs:
      - targets: ['IP_DE_VOTRE_MACHINE:9100']

  - job_name: 'nom_de_la_machine'
    static_configs:
      - targets: ['IP_DE_VOTRE_MACHINE:9100']
```

? Lancement de Prometheus

1. Activer le service

```
sudo systemctl daemon-reexec
sudo systemctl daemon-reload
sudo systemctl enable --now prometheus
```

2. Vérification

```
systemctl status prometheus
```

3. Interface Web

Accédez à :

```
http://IP_DU_SERVEUR_PROMETHEUS:9090
```

Puis allez dans : `Status > Targets`

Vous devez voir toutes les cibles listées comme **UP**.

? Connexion à Grafana

1. Sur Grafana (VM différente), aller dans **Configuration > Data Sources**
2. Cliquer sur **Add Data Source** > choisir **Prometheus**
3. Mettre comme URL : `http://IP_DU_SERVEUR_PROMETHEUS:9090`
4. Cliquer sur **Save & Test**

? Notes utiles

- Le fichier `prometheus.yml` doit avoir une **seule section** `scrape_configs:`
- Chaque `job_name` correspond à une cible identifiée dans Grafana
- Si le service ne démarre pas, vérifiez les erreurs YAML :

```
/usr/local/bin/prometheus --config.file=/etc/prometheus/prometheus.yml
```

- Exemple d'erreur courante :

```
yaml: unmarshal errors: field scrape_configs already set
```

=> Ça signifie que `scrape_configs:` est déclaré deux fois

□□ Prometheus est maintenant prêt à collecter des métriques sur tous vos serveurs □

Mise en place de grafana

Mise en place de Grafana pour visualiser les métriques Prometheus

Ce guide explique comment installer et configurer **Grafana** sur une VM dédiée (ou LXC), afin d'afficher les métriques collectées par Prometheus, comme celles de **Node Exporter**, **Proxmox Exporter**, etc.

> Vous pouvez également l'installer de manière simple et rapide avec l'aide de Helper-Script :
> https://community-scripts.github.io/ProxmoxVE/scripts?id=grafana

---

? Objectif

- Visualiser en temps réel les métriques de vos machines avec des dashboards interactifs
- Créer des alertes ou graphiques personnalisés via Prometheus

---

? Prérequis

- Une VM dédiée pour Grafana (Debian/Ubuntu recommandé)
- Un serveur Prometheus fonctionnel accessible via IP

?? Installation de Grafana (mode natif, sans Docker)

1. Ajouter le dépôt officiel Grafana

```
sudo apt-get install -y software-properties-common
sudo add-apt-repository "deb https://packages.grafana.com/oss/deb stable main"

wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -
```

2. Installer Grafana

```
sudo apt update
sudo apt install grafana -y
```

3. Activer et démarrer Grafana

```
sudo systemctl daemon-reload
sudo systemctl enable --now grafana-server
```

4. Accès à l'interface web

```
http://IP_DE_LA_VM_GRAFANA:3000
```

Identifiants par défaut :

- **admin / admin** (puis changement de mot de passe)

? Connexion à Prometheus

1. Aller dans Grafana > Configuration > Data Sources

- Cliquer sur **Add Data Source**
- Choisir **Prometheus**

2. Configuration de la source

- **URL** : `http://IP_DU_SERVEUR_PROMETHEUS:9090`
- Laisser les autres options par défaut
- Cliquer sur **Save & Test**
- Si tout est bon : "Data source is working"

? Importer des dashboards

Dashboard Node Exporter (métriques système)

- Aller dans **Dashboards > Import**
- Entrer l'ID : `1860`
- Choisir la data source Prometheus
- Cliquer sur **Import**

Dashboard Proxmox Exporter

- Aller dans **Dashboards > Import**
- Entrer l'ID : `11074`
- Importer

Dashboard Loki (si Promtail est installé)

- ID recommandé : `13639` (journald, fail2ban, syslog)

?? Bonnes pratiques

- Organisez vos dashboards par machine ou par rôle (ex : VPS, proxmox, backup...)
- Créez des "folders" pour chaque groupe
- Activez l'authentification externe (LDAP, OAuth) si Grafana est exposé publiquement

---

? Sécurité (optionnel)

- Si vous exposez Grafana sur Internet, pensez à le placer derrière un reverse proxy (ex : Caddy, Nginx) avec HTTPS et mot de passe
- Sinon, limitez l'accès au port 3000 avec un pare-feu ou WireGuard

---

? Notes utiles

- Les sources de données Grafana peuvent inclure Prometheus, Loki, InfluxDB, etc.
- Vous pouvez aussi créer des alertes dans Grafana (Alerts v2)
- Les dashboards peuvent être exportés et sauvegardés au format JSON

□□ Grafana est maintenant prêt à afficher vos métriques Prometheus sur une interface puissante et personnalisable □

Affine

AFFiNE est une plateforme de travail tout-en-un qui fusionne les documents, les tableaux blancs et les bases de données.

Mise à jour (AFFiNE Self-Hosted)

? Étape 0 — Sauvegarde (très important)

Avant toute chose :

- Sauvegarde ton fichier `compose.yml`
- Sauvegarde tes données importantes (par exemple : volumes Docker, fichiers de config, base de données, etc.)

> En cas de souci pendant la mise à jour, tu pourras revenir en arrière sans perte.

? Étape 1 — Télécharger le nouveau `compose.yml`

1. Va sur la page officielle pour récupérer le **dernier fichier** `compose.yml`.
2. Remplace ton fichier existant (`compose.yml`) par celui que tu viens de télécharger.

⚠ **Attention** :
Si tu avais modifié ton ancien fichier (`ports`, `volumes`, variables d'environnement, etc.), **copie ces changements** dans le nouveau fichier.

? Étape 2 — Récupérer la dernière image Docker

Dans le terminal (dans le dossier où se trouve ton `compose.yml`) :

```
docker compose -f compose.yml pull
```

> Cette commande va télécharger la nouvelle version de l'image AFFiNE depuis Docker Hub.

□□ Étape 3 — Redémarrer le service

Toujours dans le terminal :

```
docker compose -f compose.yml up
```

>

> Cela va arrêter l’ancienne version (si elle tourne) et démarrer AFFiNE avec la **dernière version**.

? Mise à jour terminée !

Tu as maintenant la **dernière version de AFFiNE** en local, avec tous tes réglages conservés (si tu les as bien remis dans le nouveau fichier).

Configuration Affine AI (OpenAI Override)

1. Contexte et Problématique

Par défaut, les versions self-hosted d'Affine (v0.18+) sont configurées pour utiliser des modèles Google Gemini spécifiques. L'objectif est de forcer l'utilisation d'OpenAI (GPT-4o) pour tous les scénarios IA en contournant les limitations hardcodées du backend.

2. Configuration Infrastructure (Docker Compose)

La variable d'environnement `selfhosted=true` est **obligatoire** pour débloquer le menu d'administration IA et autoriser les API tiers.

Modification du fichier `docker-compose.yml`

Modifier le service `affine` pour ajouter les variables et fixer le montage du fichier de configuration dans un répertoire neutre (`/opt`) pour éviter les problèmes de permissions `root` vs `node`.

```
services:
  affine:
    image: ghcr.io/toeverything/affine:stable
    environment:
      # ... variables existantes (DB, Redis) ...
      - AFFINE_COPILOT_ENABLED=true

      # CRITIQUE : Active le support des API tiers et l'interface Admin AI
      - selfhosted=true

      # Force le chemin de configuration (évite les erreurs de dossier home)
      - AFFINE_CONFIG_PATH=/opt/affine_config.json

    volumes:
```

```
      - ${UPLOAD_LOCATION}:/root/.affine/self-host/storage
      # Montage dans /opt pour contourner les restrictions de permissions /root
      - ${CONFIG_LOCATION}:/opt/affine_config.json
```

> *Note : Assurez-vous que `${CONFIG_LOCATION}` dans le fichier `.env` pointe bien vers votre fichier json sur l'hôte.*

3. Configuration Initiale (Fichier JSON)

Ce fichier sert uniquement à déclarer le fournisseur. La configuration fine des modèles se fera via l'interface web.

Fichier sur l'hôte : `config.json`

```
{
  "copilot": {
    "enabled": true,
    "providers": {
      "openai": {
        "apiKey": "sk-XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
      }
    },
    "embedding": {
      "provider": "openai",
      "model": "text-embedding-3-small",
      "apiKey": "sk-XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
    }
  }
}
```

> *Note : Laisser les permissions du fichier à `644` pour qu'il soit lisible par le conteneur.*

4. Configuration Finale (Interface Web Admin)

C'est la méthode recommandée ("Plan B") pour assurer la persistance et mapper correctement les modèles internes d'Affine vers OpenAI.

1. Se connecter à l'instance Affine.

2. Accéder à l'URL : `https://votre-domaine-affine.com/admin` (ou via le menu **Settings > Admin Panel**).
3. Aller dans l'onglet **AI** (ou Intelligence).
4. Activer l'option **Enable Copilot**.

Mapping des Scénarios (Surcharge)

Dans la section de configuration des scénarios (Custom Models), coller le JSON suivant. **Important :** Cette configuration redirige toutes les requêtes internes (qui demandent par défaut `gemini-2.5-flash`) vers `gpt-4o`.

```
{
  "scenarios": {
    "chat": "gpt-4o",
    "chat_stream": "gpt-4o",
    "writing": "gpt-4o",
    "translation": "gpt-4o",
    "summary": "gpt-4o",
    "brainstorm": "gpt-4o",
    "coding": "gpt-4o",
    "check_grammar": "gpt-4o",
    "quick_decision_making": "gpt-4o",
    "quick_text_generation": "gpt-4o",
    "complex_text_generation": "gpt-4o",
    "polish_and_summarize": "gpt-4o",
    "audio_transcribing": "whisper-1",
    "image": "dall-e-3"
  },
  "override_enabled": true
}
```

Vérification OpenAI Provider

Juste en dessous, dans la section **OpenAI Provider**, vérifier que la clé API est bien détectée (masquée). Si nécessaire, forcer la configuration :

```
{
  "apiKey": "sk-XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
}
```

5. Dépannage Rapide

Si l'IA ne répond pas (Erreur 500 ou "No provider available") :

1. Vérifier les logs : `docker compose logs -f affine`
2. Si erreur `ZodError` : Le fichier JSON est mal formé ou le modèle retourné par l'API n'est pas une liste valide.
3. Si erreur `Permission denied` : Vérifier que le montage Docker n'est pas dans `/root` côté conteneur.
4. Si les menus Admin sont invisibles : Vérifier que `selfhosted=true` est bien chargé dans les variables d'environnement.

Wazuh

Wazuh : Votre SIEM & XDR Open Source

Wazuh est une plateforme de securite gratuite et open-source offrant une visibilite unifiee sur la securite des endpoints et du cloud. Elle combine les fonctions de SIEM et de XDR.

Pourquoi utiliser Wazuh ?

- **Detection d intrusions :** Analyse des logs en temps reel.
- **Verification d integrite (FIM) :** Surveille les changements sur les fichiers systeme.
- **Analyse des vulnerabilites :** Inventaire des applications et correlation CVE.

Architecture et Fonctionnement

- **Wazuh Manager :** Le coeur du systeme.
- **Wazuh Indexer :** Moteur de recherche et d analyse.
- **Wazuh Dashboard :** L interface graphique de visualisation.
- **Agents Wazuh :** Programmes installes sur les machines surveillees.

> Documentation officielle : **documentation.wazuh.com**