RAPPORT DE STAGE

BTS SIO solutions d'infrastructure,

systèmes et réseaux (SISR)

Décembre 2025-Janvier 2026

Aziz BENYAMINA-HOUARI

Année scolaire 2025-2026

Sommaire

Remerciements

Je souhaite exprimer ma profonde gratitude à **Elie Annestay**, Développeur chez J&A Business Consulting. Je le remercie tout particulièrement pour m'avoir confié l'intégralité de mes missions et pour m'avoir guidé avec expertise tout au long de ce stage. Son accompagnement technique et ses conseils ont été essentiels à la réussite de mes travaux.

Je remercie également **Alia JELEDI**, Directrice de publication, pour m'avoir permis d'intégrer le cabinet et de participer à ces projets d'envergure.

Enfin, je remercie l'ensemble de l'équipe de J&A Business Consulting pour son accueil et son soutien constant.

Introduction

Ce stage de huit semaines a été réalisé dans le cadre de ma formation en BTS SIO, option SISR, au sein de l'entreprise J&A Business Consulting. Il m'a permis de confronter les connaissances théoriques acquises en cours aux réalités du monde professionnel, dans un environnement orienté vers la transformation digitale et la sécurisation des systèmes d'information, où les enjeux de fiabilité, de disponibilité et de sécurité sont primordiaux.

En tant que stagiaire, j'ai participé à différentes missions liées à l'administration des systèmes, ainsi qu'à la cybersécurité. Ces missions ont porté notamment sur la sécurisation d'infrastructures existantes, le durcissement de serveurs Linux, la gestion des accès, la configuration de services réseau et la mise en place d'outils de supervision et de sécurité. J'ai également été amené à intervenir sur des environnements conteneurisés et à participer à la documentation technique des solutions mises en œuvre.

Ce stage m'a permis de développer des compétences pratiques en administration systèmes et réseaux, de mieux comprendre les enjeux de la cybersécurité en entreprise et de confirmer mon intérêt pour les métiers liés à l'exploitation et à la sécurisation des infrastructures informatiques.

Présentation de l'entreprise

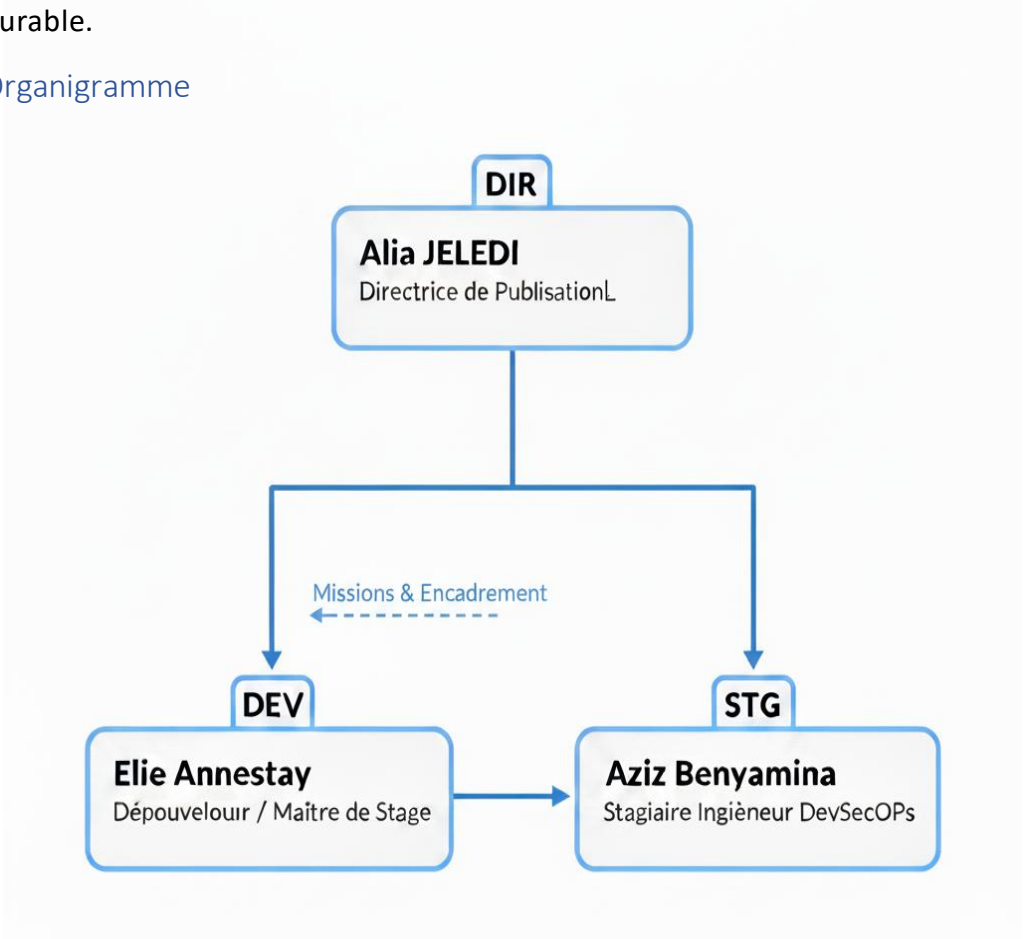
Nom de la société : J&Abusiness consulting, SIRET :91803775500017
Adresse du siège social : 124 rue de la Pompe 75116 Paris

Directeur: Alia JELEDI

J&A Business Consulting est une entreprise spécialisée dans le conseil en transformation digitale et en management stratégique. Elle accompagne ses clients dans la structuration, le pilotage et l'optimisation de leurs projets, en proposant des solutions innovantes adaptées à leurs besoins. L'entreprise intervient principalement auprès de startups et de PME, en les aidant à relever les défis liés à la digitalisation des processus et à l'intégration de nouvelles technologies.

Basée en France et disposant d'une expertise internationale, J&A Business Consulting combine conseil stratégique et savoir-faire technique pour garantir la réussite et la sécurité des projets de ses clients. La société met également un accent particulier sur la fiabilité et la pérennité des systèmes d'information, afin d'assurer un accompagnement complet et durable.

Organigramme



Descriptif de mon quotidien

Pendant mon stage chez J&A Business Consulting, mon quotidien était principalement centré sur la sécurisation des serveurs et la rédaction de documentation technique. J'ai pu mettre en pratique mes connaissances théoriques en participant à des missions concrètes, tout en découvrant le fonctionnement d'un service informatique dans un contexte professionnel.

Chaque jour, je travaillais sur le durcissement des serveurs, en appliquant des mesures de sécurité pour protéger les systèmes et les données de l'entreprise. Je réalisais également des documents et guides destinés à expliquer les procédures mises en place, ce qui permettait à l'équipe de suivre et de maintenir les bonnes pratiques de sécurité.

En plus de mes interventions sur les serveurs, j'ai effectué quelques jours de télétravail, ce qui m'a appris à gérer mon temps et à rester autonome dans la réalisation des tâches confiées. Cette organisation m'a permis de développer mes compétences techniques et ma rigueur, tout en comprenant l'importance de la documentation et du respect des procédures dans un environnement professionnel sécurisé.

Activités / Manipulations pendant les stages

1. Sécurisation et Durcissement — Serveur Debian 13 (Trixie)

Linux Hardening Best Practices

Ma première mission chez J&A Business Consulting a consisté à sécuriser les serveurs Linux. J'ai appliqué différentes mesures de durcissement, comme la mise à jour des paquets, la gestion des utilisateurs et la sécurisation des accès SSH avec des clés et des permissions strictes.

Pour faciliter la reproduction de ces configurations, j'ai créé un script d'automatisation regroupant toutes les étapes du durcissement.

Durcissement d'un serveur Debian

1.1 Introduction

L'objectif était de passer d'une installation par défaut à un niveau **production**, résistant aux attaques courantes (brute force, spoofing, rootkits).

1.2 Gestion des accès et identités

- **Création d'un utilisateur `deploy` avec droits sudo** et suppression de l'usage direct du root :

```
sudo useradd -m -s /bin/bash deploy
sudo passwd deploy
sudo usermod -aG sudo deploy
```

- **Configuration SSH** : port 5956, root désactivé, authentification par clé uniquement, restriction des options :

```
sudo nano /etc/ssh/sshd_config
# Port 5956
# PermitRootLogin no
# PasswordAuthentication no
# PubkeyAuthentication yes
# MaxAuthTries 3, MaxSessions 2
sudo systemctl restart ssh
```

- **Création du dossier SSH avec permissions strictes** :

```
mkdir -p /home/deploy/.ssh
chmod 700 /home/deploy/.ssh
chmod 600 /home/deploy/.ssh/authorized_keys
chown -R deploy:deploy /home/deploy/.ssh
```

1.3 Sécurité réseau et pare-feu

- **UFW** : politique de refus par défaut, ouverture uniquement des ports nécessaires : SSH (5956),

```
sudo apt install ufw -y
sudo ufw default deny incoming
sudo ufw default allow outgoing
sudo ufw allow 5956/tcp
sudo ufw enable
```

- **Fail2Ban** : bannissement automatique des IP suspectes

```
sudo apt install fail2ban -y
sudo nano /etc/fail2ban/jail.local
# [sshd] enabled=true port=5956 maxretry=3 mode=aggressive bantime=1h
sudo systemctl restart fail2ban
```

- **Correction des DNS** IPv6 défectueux

```
sudo nano /etc/resolvconf/resolv.conf.d/head
# nameserver 1.1.1.1
# nameserver 8.8.8.8
sudo resolvconf -u
sudo systemctl restart networking
```

1.4 Durcissement du noyau et du système

- **Paramètres Sysctl** pour renforcer mémoire et réseau

```
sudo nano /etc/sysctl.d/99-hardening.conf
# net.ipv4.conf.all.rp_filter=1
# kernel.kptr_restrict=2
sudo sysctl -p /etc/sysctl.d/99-hardening.conf
```

- **Blacklisting modules non essentiels**

```
sudo nano /etc/modprobe.d/blacklist-peripherals.conf
# usb-storage, firewire-core
sudo update-initramfs -u
```

- **Permissions critiques et UMASK**

```
sudo chmod 700 /home/deploy
sudo chmod 700 /root
```

1.5 Audit et intégrité

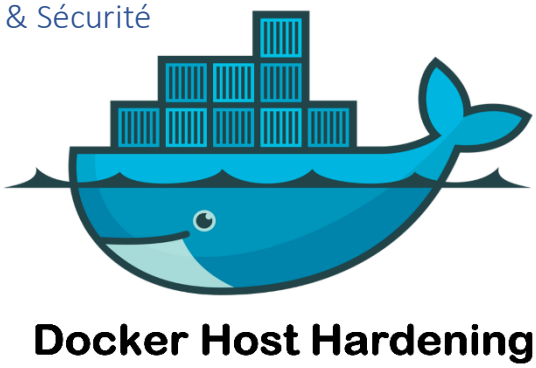
- **AIDE** pour contrôle de l'intégrité des fichiers

```
sudo apt install aide aide-common -y
sudo aide --init
sudo cp /var/lib/aide/aide.db.new /var/lib/aide/aide.db
```

- **Rkhunter et Debsums** pour vérifier les binaires et détecter rootkits

```
sudo apt install rkhunter debsums -y
sudo rkhunter --check
sudo debsums sudo
sudo debsums openssh-server
```

2 Infrastructure Docker & Sécurité



- **Serveur** : Debian 4GB
- **Projet** : Auth Infrastructure (SuperTokens, Nginx, Node.js)

2.1 Configuration du Démon Docker

- Fichier : `/etc/docker/daemon.json`
- Paramètres principaux :

```
{
  "log-driver": "json-file",
  "log-opts": {"max-size": "10m", "max-file": "3"},
  "live-restore": true,
  "no-new-privileges": true
}
```

- Justification : rotation des logs, maintien conteneurs actifs, blocage élévation privilèges.

2.2 Sécurisation des Conteneurs (Docker Compose)

- Limitation des ressources : RAM et CPU selon service.
- Cloisonnement réseau : `public_net` pour services exposés, `internal_net` pour backend et base.
- Privilèges : `no-new-privileges:true`, images Alpine Linux.
- Healthchecks : PostgreSQL (`pg_isready`), Backend (page accueil toutes 30s).

2.3 Gestion des Fichiers et Secrets

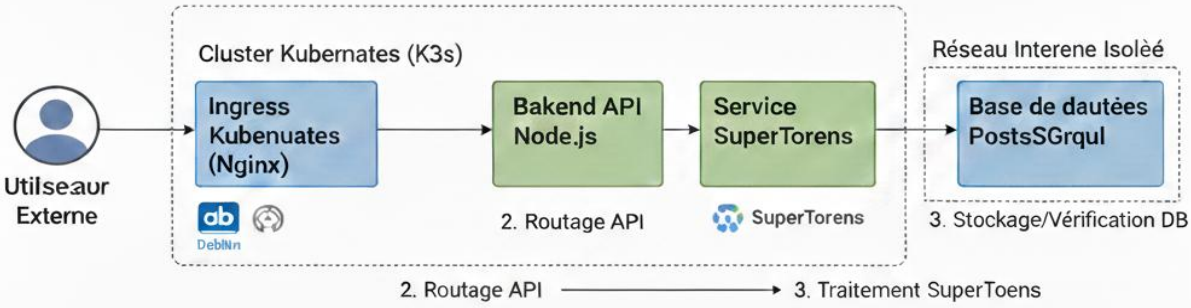
- `.env` déplacé à la racine pour build correct.
- `.dockerignore` ajouté pour exclure `.git`, `node_modules`, `.env`.

2.4 Guide de Maintenance

```
# Mettre à jour l'application
docker compose up -d --build --remove-orphans
# Vérifier état services
docker compose ps
# Nettoyer disque (1x/semaine)
docker system prune -a -f
# Sauvegarder base de données
docker exec -t auth-infrastructure-supertokens-postgresql-1 pg_dumpall -c
-U supertokens_user > backup_$(date +%F).sql
```

3 Déploiement — Platform Auth (Backend Only)

Déployer une plateforme d'authentification basée sur **SuperTokens**, **PostgreSQL** et une **API Backend**, orchestrée par **Kubernetes (K3s)**. Le frontend est volontairement désactivé ; l'Ingress redirige directement vers le backend.



3.1 Installation des outils Kubernetes

Sur un serveur Debian vierge, installation des outils nécessaires à l'administration du cluster.

```
sudo apt-get update && sudo apt-get install -y apt-transport-https ca-cert
ificates curl
```

Installation de kubectl

```
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.29/deb/Release.key | sudo
gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
echo "deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https:/
/pkgs.k8s.io/core:/stable:/v1.29/deb/ /" | sudo tee /etc/apt/sources.list.
d/kubernetes.list
sudo apt-get update && sudo apt-get install -y kubectl
```

Installation de Helm

```
curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/main
/scripts/get-helm-3
chmod 700 get_helm.sh
./get_helm.sh
```

Configuration de l'accès K3s

```
mkdir -p ~/.kube
sudo cp /etc/rancher/k3s/k3s.yaml ~/.kube/config
sudo chown deploy:deploy ~/.kube/config
chmod 600 ~/.kube/config
```

3.2 Gestion des secrets et configuration Helm

Les secrets (mots de passe, clés API) sont définis dans le fichier :

```
/opt/auth_server_tenancy/infrastructure/helm/platform/values-prod.yaml
```

Points clés : - Backend activé - Frontend désactivé - Ingress actif - Secrets isolés via Helm

Le routage doit obligatoirement pointer vers le backend.

3.3 Correction de l'Ingress (erreur 503)

Par défaut, Helm redirige / vers le frontend. Celui-ci étant désactivé, une modification du template Ingress est nécessaire.

Le service ciblé est désormais **le backend (port 3001)** afin d'éviter toute erreur 503.

3.4 Déploiement de la plateforme

```
cd /opt/auth_server_tenancy/infrastructure/helm/platform
kubectl delete ingress platform-prod-ingress -n prod
sudo helm upgrade --install prod . -f values-prod.yaml -n prod --kubeconfi
g /home/deploy/.kube/config
```

Helm crée automatiquement les ressources Kubernetes nécessaires (namespace, pods, services, secrets).

3.4 Correctif PostgreSQL (MD5 / SCRAM)

Un problème d'authentification entre PostgreSQL et SuperTokens a été corrigé.

```
kubectl exec -it platform-prod-postgres-0 -n prod -- sh -c "echo 'host all
platform 10.244.0.0/16 md5' >> /var/lib/postgresql/data/pg_hba.conf"
kubectl delete pod platform-prod-postgres-0 -n prod
kubectl delete pod $(kubectl get pods -n prod | grep supertokens | awk '{p
rint $1}') -n prod
```

3.5 Vérifications finales

```
kubectl get pods -n prod
```

Tous les pods doivent être en état **Running** (Backend, PostgreSQL, SuperTokens).

Test d'accès API :

```
curl http://auth.annestay.com/     Une réponse 404 Not Found provenant d'Ex
press confirme que le routage fonctionne correctement.
```

4 GitLab CE & WireGuard VPN

Ce projet consiste à la mise en place d'une infrastructure DevOps sécurisée sur un serveur **Debian**, intégrant **GitLab CE** pour la gestion du code source et des pipelines CI/CD, ainsi qu'un **VPN WireGuard** pour restreindre l'accès aux services.

L'objectif principal est de **limiter l'exposition réseau** : l'accès à GitLab est autorisé **uniquement via le tunnel VPN**, conformément au principe du moindre privilège.

4.1 Déploiement de GitLab CE (Docker)

Création des volumes persistants

```
mkdir -p /opt/gitlab/{config,logs,data}
```

Lancement du conteneur GitLab

```
docker run -d \
  --hostname gitlab.local \
  --name gitlab \
  --restart always \
  -p 127.0.0.1:80:80 \
  -p 127.0.0.1:443:443 \
  -v /opt/gitlab/config:/etc/gitlab \
  -v /opt/gitlab/logs:/var/log/gitlab \
  -v /opt/gitlab/data:/var/opt/gitlab \
  gitlab/gitlab-ce:latest
```

Récupération du mot de passe root initial

```
docker exec -it gitlab cat /etc/gitlab/initial_root_password
```

4.2 Installation et configuration du VPN WireGuard

```
apt update
apt install wireguard -y
```

Génération des clés serveur

```
wg genkey | tee /etc/wireguard/server.key | wg pubkey >
/etc/wireguard/server.pub
chmod 600 /etc/wireguard/server.key
```

4.3 Configuration de l'interface VPN

Fichier : `/etc/wireguard/wg0.conf`

```
[Interface]
Address = 10.7.0.1/24
ListenPort = 51820
PrivateKey = <SERVER_PRIVATE_KEY>
```

Activation du service

```
systemctl enable wg-quick@wg0
systemctl start wg-quick@wg0
```

4.4 Sécurisation réseau (UFW)

Politique par défaut

```
ufw default deny incoming
ufw default allow outgoing
```

Autorisation du VPN uniquement

```
ufw allow 51820/udp
ufw enable
```

Les ports **80/443** (GitLab) restent inaccessibles depuis Internet et sont utilisables uniquement via le VPN.

Génération des clés client

```
wg genkey | tee client1.key | wg pubkey > client1.pub
```

Déclaration du client côté serveur

```
[Peer]
PublicKey = <CLIENT_PUBLIC_KEY>
AllowedIPs = 10.7.0.2/32
```

4.5 Déploiement du GitLab Runner

Lancement du conteneur Runner

```
docker run -d \
  --name gitlab-runner \
  --restart always \
  -v /var/run/docker.sock:/var/run/docker.sock \  -v /opt/gitlab-runner:/e
tc/gitlab-runner \
  gitlab/gitlab-runner:latest
```

Enregistrement du runner

```
docker exec -it gitlab-runner gitlab-runner register
```

Conclusion

Ce stage m'a permis de consolider et d'approfondir mes compétences techniques en administration systèmes, réseaux et sécurité. J'ai pu intervenir sur des environnements réels en mettant en place des infrastructures sécurisées basées sur Linux, Docker et Kubernetes, tout en respectant les bonnes pratiques professionnelles.

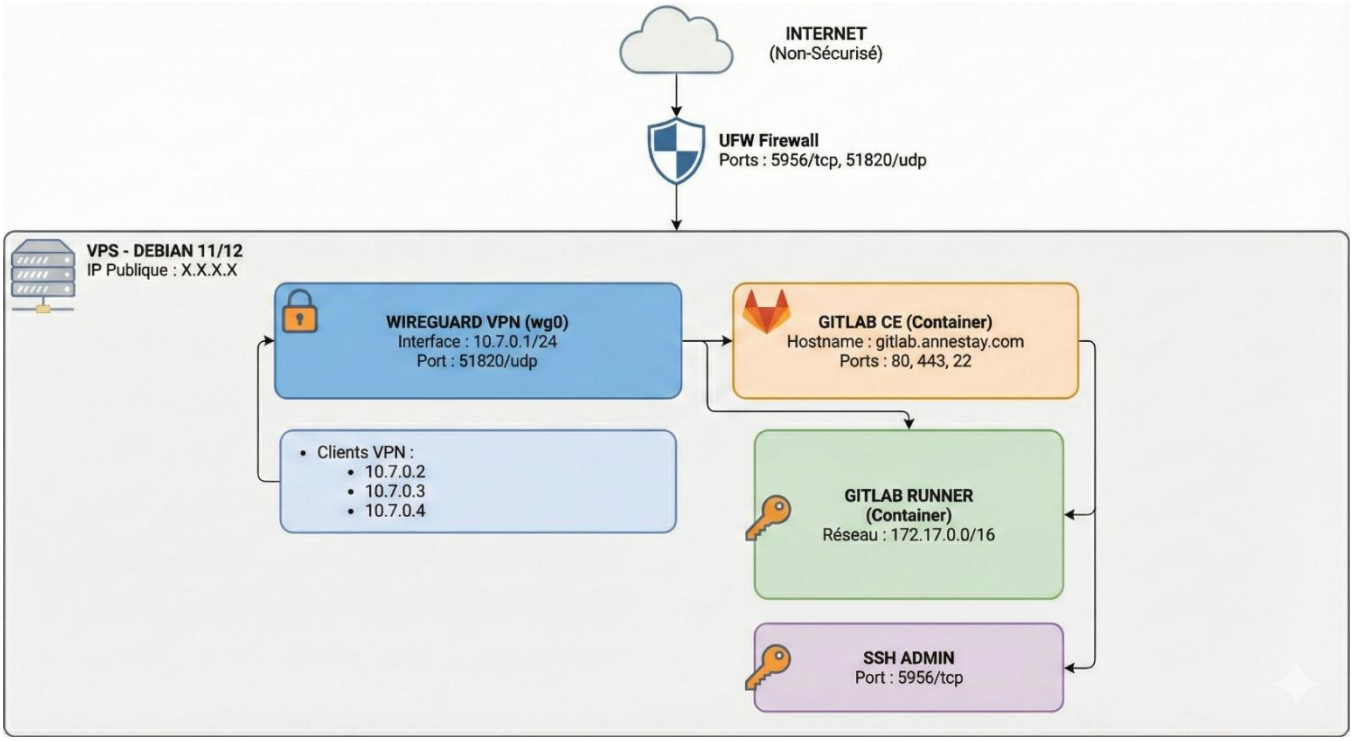
Les missions réalisées, telles que le durcissement d'un serveur Debian, la sécurisation des accès (SSH, pare-feu, VPN), le déploiement de services via Docker et Kubernetes, ainsi que la rédaction de documentations techniques, m'ont permis de mieux comprendre les enjeux de disponibilité, de sécurité et de maintenance des systèmes informatiques.

Ce stage m'a également appris à travailler de manière plus rigoureuse et autonome, à analyser des problèmes techniques concrets et à proposer des solutions adaptées aux contraintes de production. Il m'a confirmé mon intérêt pour l'administration système, la cybersécurité et les environnements DevOps, domaines dans lesquels je souhaite continuer à évoluer professionnellement.

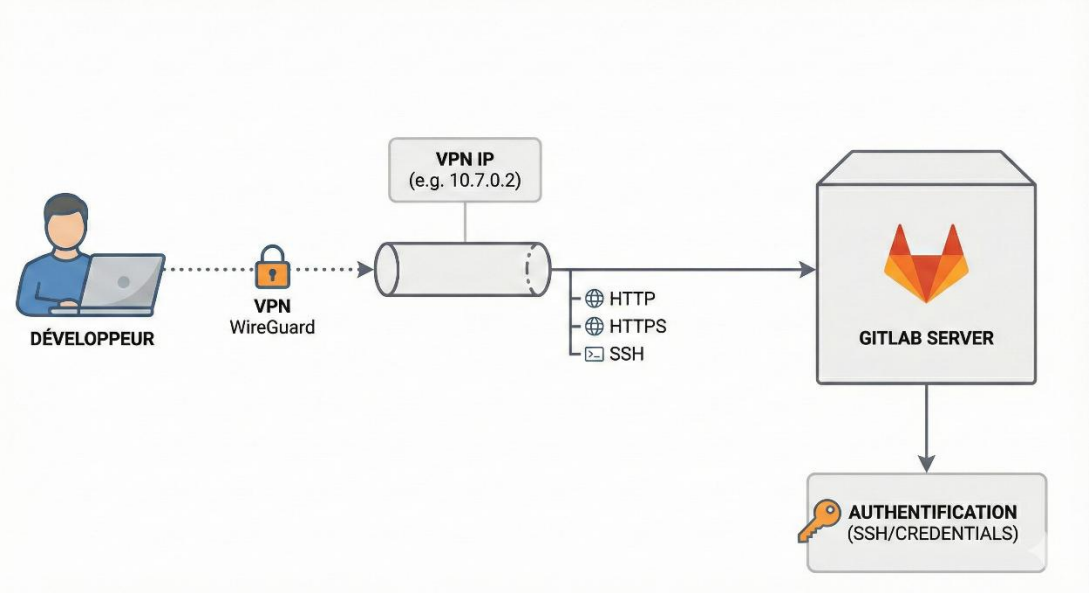
Enfin, cette expérience constitue une étape importante dans mon parcours de formation, en faisant le lien entre les enseignements théoriques du BTS SIO option SISR et leur application concrète en milieu professionne

Annexes

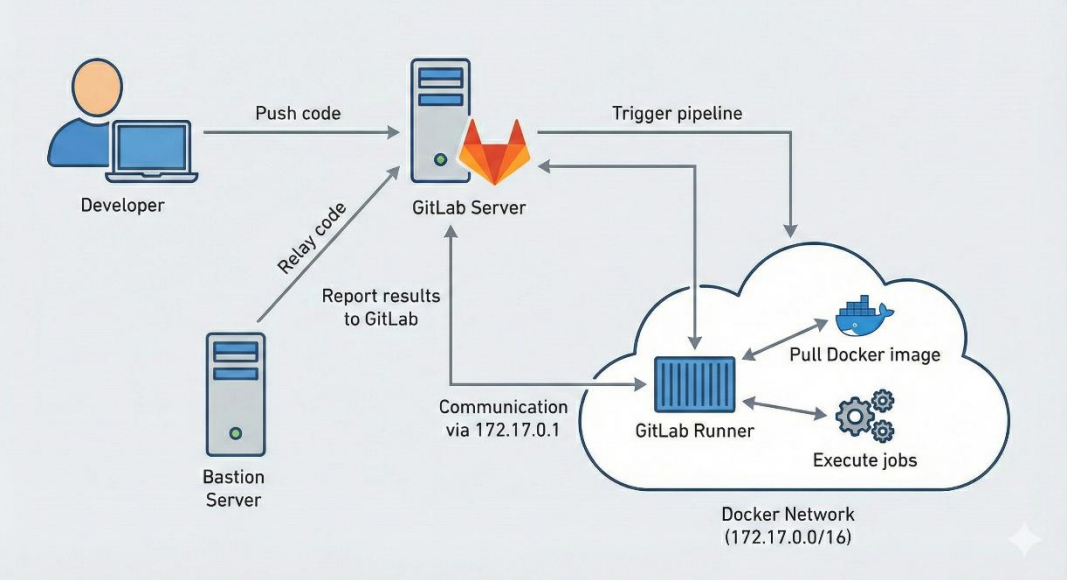
Architecture



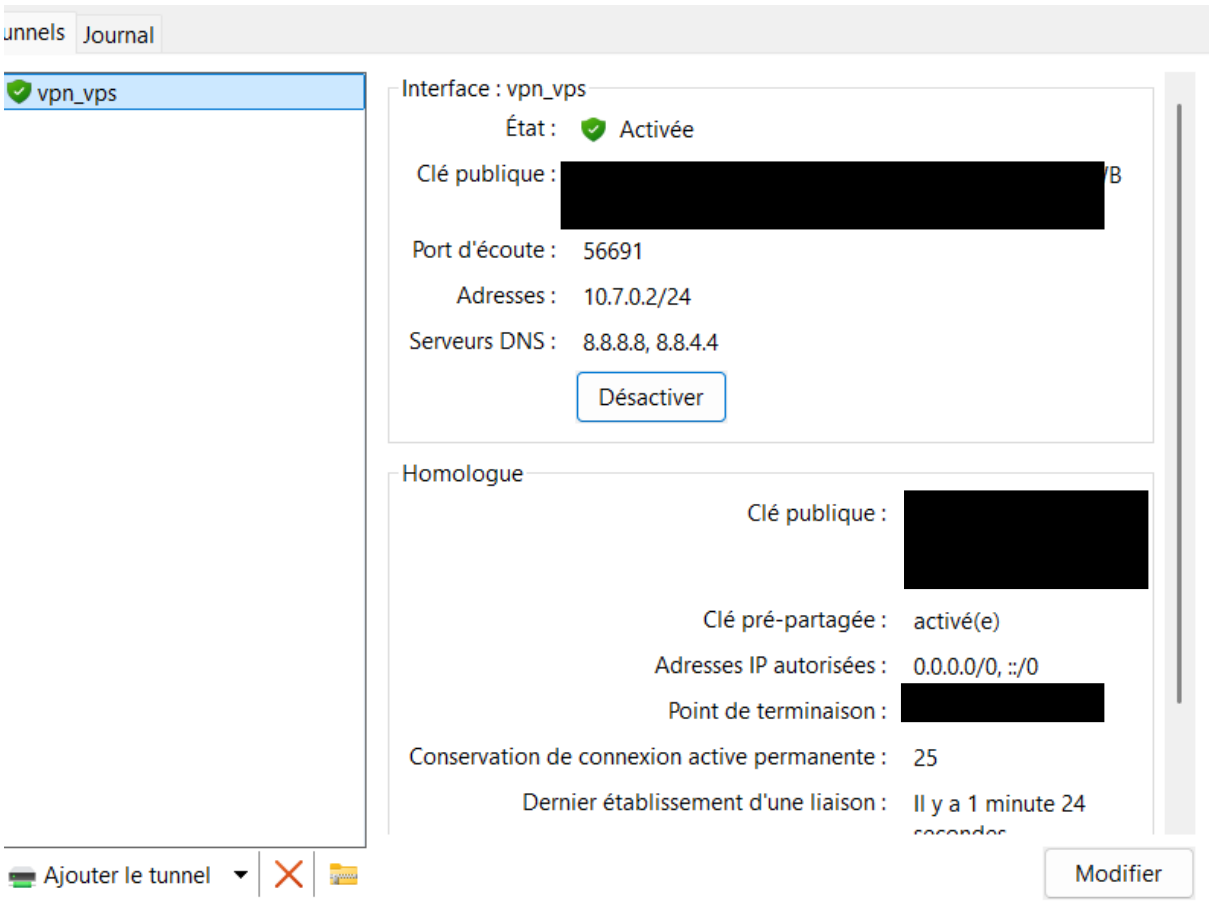
Accès développeur au Gilab



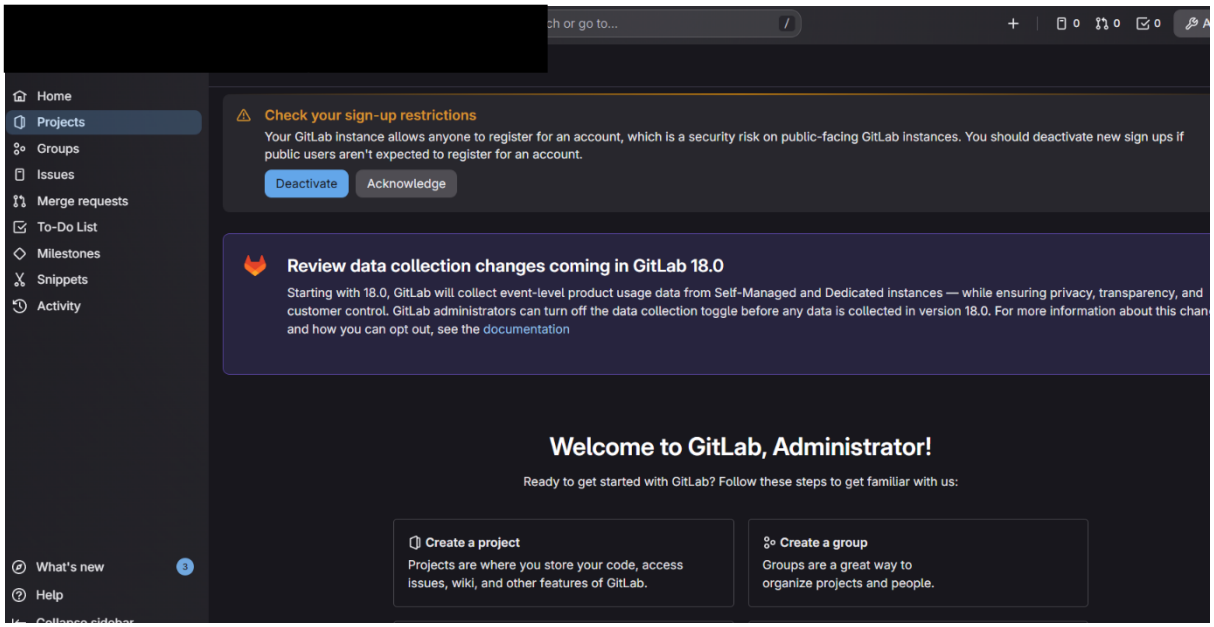
Exécution pipeline CI/CD



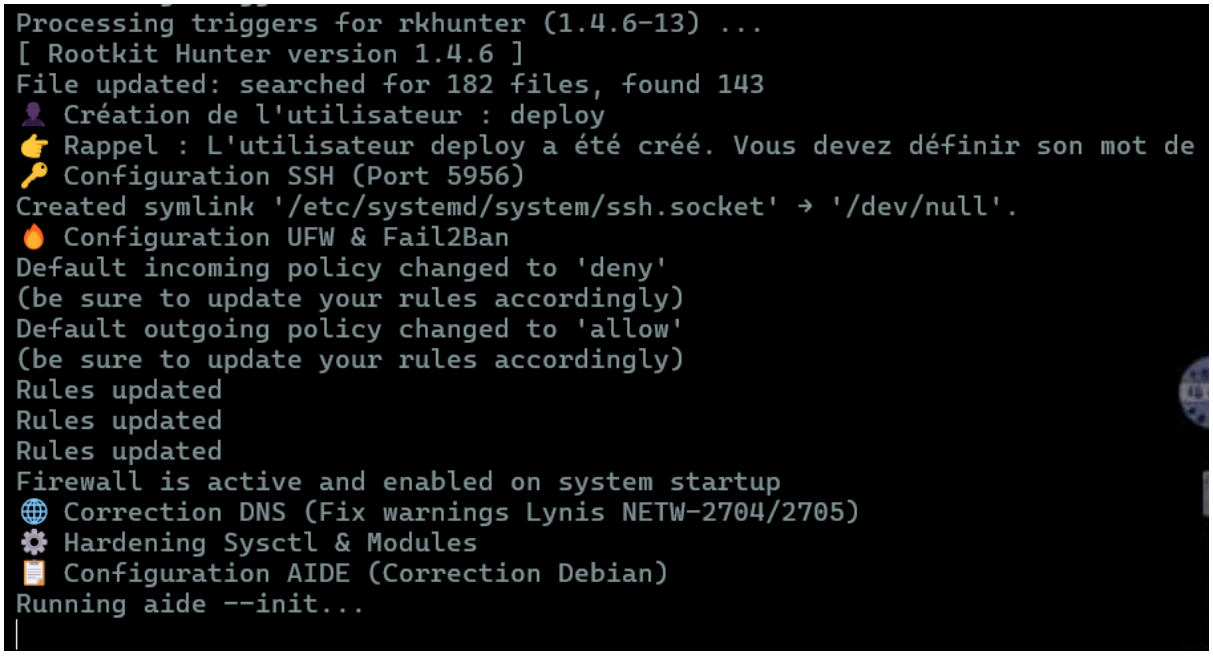
VPN

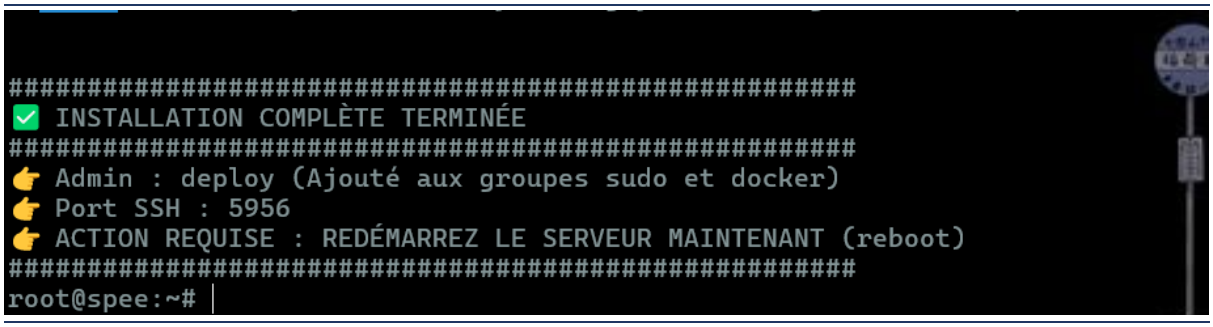


Interface GitLab

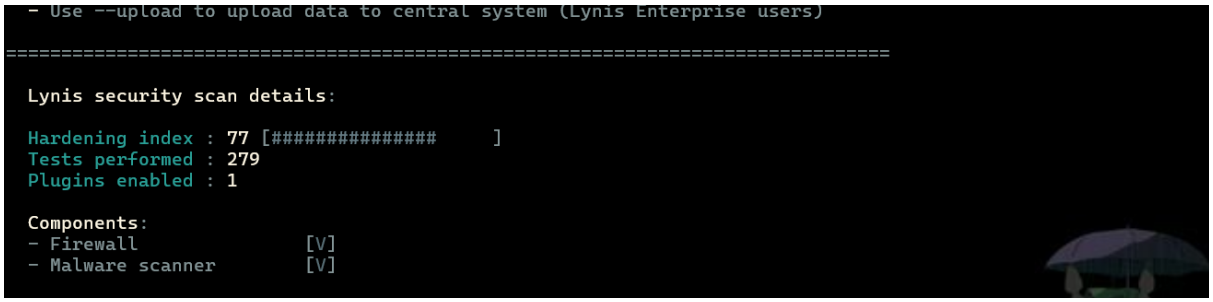


Extrait Script





Audit Lynes


- Use --upload to upload data to central system (Lynis Enterprise users)
Lynis security scan details:
Hardening index : 77 [###############   ]
Tests performed : 279
Plugins enabled : 1
Components:
- Firewall [V]
- Malware scanner [V]