VIRIEUX Gaspard
B2 INFORMATIQUE

Année 2024-2025

Documentation d'Exploitation - Projet VPN WireGuard

1. Utilisation des outils et services mis en place

Gestion du serveur VPN WireGuard

Démarrage des services

```
# Démarrer le serveur VPN
docker start wireguard

# Démarrer le serveur web de test
docker start webserver

# Vérifier que les services sont actifs
docker ps
```

Arrêt des services

```
# Arrêter le serveur VPN
docker stop wireguard

# Arrêter le serveur web
docker stop webserver

# Vérifier l'arrêt
docker ps -a
```

Consultation des logs

```
# Voir les logs du serveur VPN
docker logs wireguard

# Voir les logs en temps réel
docker logs -f wireguard

# Voir les dernières 15 lignes (configuration actuelle)
docker logs --tail 15 wireguard

# Rechercher des connexions client
docker logs wireguard | grep handshake
```

Vérification de l'état du serveur VPN

```
# Vérifier l'interface WireGuard
docker exec -it wireguard ip addr show wg0
```

```
# Voir les clients connectés
docker exec -it wireguard wg show

# Vérifier les routes VPN
docker exec -it wireguard ip route | grep "10.13.13"
```

2. Gestion des clients VPN

Obtenir les configurations client existantes

Lister les clients configurés

```
# Voir la structure des configurations
ls -la ~/wireguard-project/config/

# Lister tous les fichiers de configuration client
find ~/wireguard-project/config -name "peer*.conf" -type f

# Résultat attendu : peer1.conf, peer2.conf, peer3.conf
```

Afficher les QR codes

```
docker exec -it wireguard /app/show-peer 1
```

Récupérer les fichiers de configuration

```
# Copier la configuration du client 1 sur le bureau
cp ~/wireguard-project/config/peer1/peer1.conf ~/Desktop/client1-vpn.conf

# Afficher le contenu d'une configuration
cat ~/wireguard-project/config/peer1/peer1.conf

# Configuration type client :
# [Interface]
# Address = 10.13.13.2
# DNS = 10.13.13.1
# [Peer]
# Endpoint = 37.66.226.231:51820
# AllowedIPs = 0.0.0.0/0, ::/0
```

Ajouter un nouveau client VPN

```
# Sauvegarder la configuration actuelle
cp -r ~/wireguard-project/config ~/wireguard-project/config-backup-$(date +%Y%m%d)

# Arrêter le serveur actuel
docker stop wireguard
docker rm wireguard

# Relancer avec plus de clients
docker run -d \
  --name=wireguard \
  -e PUID=1000 \
  -e PGID=1000 \
  -e TZ=Europe/Paris \
```

```
  -e SERVERURL=auto \
  -e PEERS=5 \
  -p 51820:51820/udp \
  -v ~/wireguard-project/config:/config \
  --cap-add=NET_ADMIN \
  --sysctl="net.ipv4.conf.all.src_valid_mark=1" \
  --restart unless-stopped \
  linuxserver/wireguard:latest

# Attendre la génération des nouvelles configurations
sleep 15
docker logs --tail 10 wireguard
```

Révoquer un client VPN

Supprimer un client spécifique

```
# Sauvegarder avant modification
cp ~/wireguard-project/config/wg_confs/wg0.conf ~/wireguard-project/config/wg0.conf.backup

# Supprimer les fichiers du client (exemple peer2)
rm -rf ~/wireguard-project/config/peer2/

# Éditer manuellement la configuration serveur pour retirer le peer
nano ~/wireguard-project/config/wg_confs/wg0.conf
# (Supprimer la section [Peer] correspondante)

# Redémarrer le serveur pour appliquer les changements
docker restart wireguard
```

Réinitialisation complète des clients

```
# Sauvegarder l'ancienne configuration
mv ~/wireguard-project/config ~/wireguard-project/config-old-$(date +%Y%m%d)

# Créer un nouveau dossier de configuration
mkdir -p ~/wireguard-project/config

# Redémarrer le conteneur pour générer de nouvelles configurations
docker restart wireguard

# Attendre la régénération
sleep 15
docker logs --tail 10 wireguard
```

3. Configuration des clients

Configuration sur smartphone (iOS/Android)

Installation de l'application

Dans mon exemple c’est iOS donc WireGuard à télécharger sur AppStore

Ajout d'une configuration par QR code

1. Ouvrir l'application WireGuard
2. Appuyer sur le bouton "+" (ajouter)
3. Sélectionner "Scanner un code QR"
4. Scanner le QR code généré par : docker exec -it wireguard /app/show-peer 1
5. Nommer le tunnel (ex: "VPN-Projet" ou "VPN-10.13.13.2")
6. Sauvegarder la configuration

Configuration sur ordinateur (macOS)

Installation WireGuard sur macOS

1. Ouvrir l'App Store
2. Rechercher "WireGuard"
3. Installer l'application officielle

Import de configuration

1. Ouvrir l'application WireGuard
2. Cliquer sur "Import tunnel(s) from file..."
3. Naviguer vers le fichier de configuration (utiliser Cmd+Shift+G pour aller à ~/wireguard-project/config/peer1/)
4. Sélectionner peer1.conf
5. Activer le tunnel

Configuration manuelle sur macOS

1. Cliquer sur "Add Empty Tunnel..."
2. Copier-coller le contenu du fichier peer1.conf
3. Sauvegarder avec un nom descriptif

4. Tests et vérifications

Vérifier que le VPN fonctionne

Test de base - Changement d'adresse IP

Avant connexion VPN :

1. Aller sur https://whatismyip.com
2. Noter l'adresse IP affichée

Après connexion VPN :

1. Activer le VPN sur l'appareil client

2. Aller à nouveau sur https://whatismyip.com
3. Vérifier que l'adresse IP a changé.

Test d'accès aux ressources internes

```
# Depuis un client connecté au VPN
# Tester l'accès au serveur web via l'IP locale
curl http://192.168.1.75:8081

# Ou dans un navigateur web
http://192.168.1.75:8081

# Résultat attendu : Page d'accueil Nginx (code 200)
```

Test d'accès au serveur DNS privé

```
# Depuis un client VPN, tester le DNS interne
nslookup google.com 10.13.13.1

# Ou sur mobile, vérifier dans les paramètres réseau que le DNS est 10.13.13.1
```

Vérification côté serveur

```
# Voir les clients actuellement connectés
docker exec -it wireguard wg show

# Exemple de sortie :
# interface: wg0
#   public key: [clé publique du serveur]
#   listening port: 51820
#
# peer: [clé publique du client]
#   endpoint: [IP client]:port
#   allowed ips: 10.13.13.2/32
#   latest handshake: X seconds ago
#   transfer: X.XX KiB received, X.XX KiB sent
```

Test de connectivité réseau interne

```
# Ping vers un client VPN depuis le serveur
docker exec -it wireguard ping -c 3 10.13.13.2

# Test de route
docker exec -it wireguard ip route get 10.13.13.2
```

Diagnostiquer les problèmes de connexion

Problème : Le client ne peut pas se connecter

Vérifications serveur :

```
# 1. Vérifier que le serveur fonctionne
docker ps | grep wireguard

# 2. Vérifier les logs d'erreur
```

```
docker logs wireguard | grep -i error

# 3. Vérifier que le port est ouvert
sudo lsof -i :51820

# 4. Vérifier l'interface WireGuard
docker exec -it wireguard ip addr show wg0
```

Solutions :

- Si le conteneur est arrêté : docker start wireguard
- Si le port n'est pas ouvert : Redémarrer le conteneur
- Si l'interface n'existe pas : docker restart wireguard

Problème : Le client se connecte mais n'a pas Internet

Diagnostic :

```
# Vérifier la configuration NAT
docker exec -it wireguard iptables -t nat -L

# Doit montrer une règle MASQUERADE
```

Solutions :

1. Vérifier AllowedIPs sur le client : doit être 0.0.0.0/0, ::/0
2. Vérifier DNS sur le client : doit être 10.13.13.1 ou 1.1.1.1, 8.8.8.8
3. Redémarrer le serveur : docker restart wireguard

Problème : Conflit d'adresses IP

Diagnostic :

```
# Vérifier les réseaux locaux
ifconfig | grep "inet " | grep -v 127.0.0.1

# Si l'IP locale change (actuellement 192.168.1.75)
# Les clients doivent être reconfigurés avec le nouveau Endpoint
```

Solution :

1. Noter la nouvelle IP locale
2. Régénérer les configurations client
3. Reconfigurer tous les clients avec les nouveaux QR codes

Problème : Performance dégradée

Monitoring :

```
# Vérifier l'utilisation des ressources
docker stats

# Vérifier les erreurs réseau
docker exec -it wireguard dmesg | tail

# Vérifier les transferts de données
docker exec -it wireguard wg show
```

5. Maintenance et sauvegarde

Sauvegarde des configurations

Sauvegarde complète

```
# Créer un dossier de sauvegarde avec horodatage
mkdir -p ~/wireguard-backup/$(date +%Y%m%d-%H%M%S)

# Copier toutes les configurations
cp -r ~/wireguard-project/config/* ~/wireguard-backup/$(date +%Y%m%d-%H%M%S)/

# Créer une archive compressée
tar -czf ~/wireguard-backup/config-$(date +%Y%m%d-%H%M%S).tar.gz ~/wireguard-project/config/

# Sauvegarder aussi les configurations Docker
docker inspect wireguard > ~/wireguard-backup/wireguard-container-$(date +%Y%m%d-%H%M%S).json
```

Restaurer des configurations

```
# Arrêter le serveur
docker stop wireguard

# Restaurer depuis la sauvegarde (remplacer YYYYMMDD-HHMMSS par la date souhaitée)
rm -rf ~/wireguard-project/config/*
cp -r ~/wireguard-backup/YYYYMMDD-HHMMSS/* ~/wireguard-project/config/

# Ou restaurer depuis une archive
tar -xzf ~/wireguard-backup/config-YYYYMMDD-HHMMSS.tar.gz -C ~/

# Redémarrer le serveur
docker start wireguard

# Vérifier que la restauration fonctionne
docker logs --tail 10 wireguard
```

Mise à jour du serveur

Mettre à jour l'image Docker

```
# Télécharger la dernière version
docker pull linuxserver/wireguard:latest

# Sauvegarder la configuration actuelle
cp -r ~/wireguard-project/config ~/wireguard-project/config-backup-avant-maj
```

```
# Arrêter et supprimer l'ancien container
docker stop wireguard
docker rm wireguard

# Relancer avec la nouvelle image (mêmes paramètres qu'actuellement)
docker run -d \
  --name=wireguard \
  -e PUID=1000 \
  -e PGID=1000 \
  -e TZ=Europe/Paris \
  -e SERVERURL=auto \
  -e PEERS=3 \
  -p 51820:51820/udp \
  -v ~/wireguard-project/config:/config \
  --cap-add=NET_ADMIN \
  --sysctl="net.ipv4.conf.all.src_valid_mark=1" \
  --restart unless-stopped \
  linuxserver/wireguard:latest

# Vérifier que tout fonctionne
docker ps
docker logs --tail 15 wireguard
```

Scripts de monitoring

Script de vérification de santé

```
#!/bin/bash
# Fichier : ~/check-vpn-health.sh

echo "=== Vérification VPN - $(date) ==="

echo "État des containers :"
docker ps | grep -E "(wireguard|webserver)" || echo "ERREUR: Containers non trouvés"

echo -e "\nUtilisation des ressources :"
docker stats --no-stream | grep -E "(wireguard|webserver)"

echo -e "\nClients connectés :"
docker exec -it wireguard wg show 2>/dev/null | grep -A5 "peer:" || echo "Aucun client connecté"

echo -e "\nInterface WireGuard :"
docker exec -it wireguard ip addr show wg0 2>/dev/null | grep "inet" || echo "Interface WireGuard non
active"

echo "=== Fin vérification ==="
```

Utilisation du script de monitoring

```
# Rendre le script exécutable
chmod +x ~/check-vpn-health.sh

# Exécuter la vérification
~/check-vpn-health.sh
```

```
# Programmer des vérifications automatiques (optionnel)
# Ajouter à crontab pour vérification toutes les heures :
# 0 * * * * ~/check-vpn-health.sh >> ~/vpn-health.log 2>&1
```

Script de redémarrage automatique

```
#!/bin/bash
# Fichier : ~/restart-vpn-if-needed.sh

# Vérifier si WireGuard répond
if ! docker exec -it wireguard wg show >/dev/null 2>&1; then
   echo "$(date): WireGuard ne répond pas, redémarrage..." >> ~/vpn-restart.log
   docker restart wireguard
   sleep 10

   # Vérifier si le redémarrage a fonctionné
   if docker exec -it wireguard wg show >/dev/null 2>&1; then
      echo "$(date): Redémarrage réussi" >> ~/vpn-restart.log
   else
      echo "$(date): ERREUR - Redémarrage échoué" >> ~/vpn-restart.log
   fi
fi
```

6. Sécurité opérationnelle

Bonnes pratiques de sécurité

Audit des connexions

```
# Voir l'historique des connexions dans les logs
docker logs wireguard | grep -i "handshake\|peer"

# Exporter les logs pour analyse (avec horodatage)
docker logs wireguard > ~/vpn-audit-$(date +%Y%m%d-%H%M%S).log

# Analyser les tentatives de connexion
docker logs wireguard | grep -E "handshake|endpoint" | tail -20
```

Rotation des clés de sécurité

```
# Sauvegarder l'ancienne configuration
cp -r ~/wireguard-project/config ~/wireguard-project/config-old-$(date +%Y%m%d)

# Supprimer les anciennes clés
rm -rf ~/wireguard-project/config/*

# Redémarrer pour générer de nouvelles clés
docker restart wireguard

# Attendre la génération
sleep 15

# Vérifier que les nouvelles configurations sont créées
ls -la ~/wireguard-project/config/peer*/
```

```
# Distribuer les nouvelles configurations aux clients
echo "Nouvelles configurations générées, redistribution requise :"
for i in {1..3}; do
   echo "Client $i : docker exec -it wireguard /app/show-peer $i"
done
```

Surveillance des tentatives d'intrusion

```
# Surveiller les tentatives de connexion échouées
docker logs wireguard | grep -i "failed\|error\|denied" | tail -10

# Surveiller l'utilisation anormale de bande passante
docker exec -it wireguard wg show | grep "transfer:"

# Analyser le trafic réseau si nécessaire (mode debug)
# sudo tcpdump -i any port 51820 -n | head -20
```

Gestion des accès d'urgence

```
# Créer un client d'urgence avec accès limité
# (À faire uniquement en cas de besoin)

# 1. Créer une configuration temporaire
docker exec -it wireguard /app/add-peer emergency

# 2. Configurer avec accès restreint (éditer manuellement)
# AllowedIPs = 10.13.13.0/24  # Seulement réseau VPN, pas Internet
```

Monitoring avancé

Métriques de performance

```
# Script de collecte de métriques
#!/bin/bash
# Fichier : ~/collect-metrics.sh

TIMESTAMP=$(date +"%Y-%m-%d %H:%M:%S")
LOG_FILE=~/vpn-metrics.log

echo "$TIMESTAMP - Début collecte métriques" >> $LOG_FILE

# Utilisation CPU/RAM
docker stats --no-stream --format "table {{.Name}}\t{{.CPUPerc}}\t{{.MemUsage}}" | grep -E
"(wireguard|webserver)" >> $LOG_FILE

# Nombre de clients connectés
CLIENTS=$(docker exec -it wireguard wg show | grep -c "peer:")
echo "$TIMESTAMP - Clients connectés: $CLIENTS" >> $LOG_FILE

# Trafic réseau total
docker exec -it wireguard wg show | grep "transfer:" >> $LOG_FILE

# Test de latence
PING_RESULT=$(ping -c 1 192.168.1.75 | grep "time=" | cut -d'=' -f4)
```

```
echo "$TIMESTAMP - Latence locale: $PING_RESULT" >> $LOG_FILE

echo "$TIMESTAMP - Fin collecte métriques" >> $LOG_FILE
echo "---" >> $LOG_FILE
```

Alertes automatiques

```
# Script d'alerte simple
#!/bin/bash
# Fichier : ~/vpn-alerts.sh

# Vérifier l'utilisation mémoire (alerte si > 100MB)
MEM_USAGE=$(docker stats --no-stream --format "{{.MemUsage}}" wireguard | cut -d'/' -f1 | sed
's/MiB//')
if (( $(echo "$MEM_USAGE > 100" | bc -l) )); then
   echo "ALERTE: Utilisation mémoire élevée: ${MEM_USAGE}MiB" | tee -a ~/vpn-alerts.log
fi

# Vérifier que le serveur répond
if ! curl -s http://localhost:8081 >/dev/null; then
   echo "ALERTE: Serveur web ne répond pas" | tee -a ~/vpn-alerts.log
fi

# Vérifier l'uptime des containers
UPTIME=$(docker ps --format "{{.Status}}" | grep wireguard)
if [[ $UPTIME == *"Restarting"* ]]; then
   echo "ALERTE: Container WireGuard en redémarrage" | tee -a ~/vpn-alerts.log
fi
```

7. Procédures d'urgence

Arrêt d'urgence du VPN

```
# Arrêt immédiat de tous les services VPN
echo "URGENCE: Arrêt des services VPN - $(date)" >> ~/vpn-emergency.log

# Arrêter les containers
docker stop wireguard webserver

# Bloquer le port VPN au niveau système (si nécessaire)
# sudo pfctl -f /dev/stdin <<< "block in on any port 51820"

# Vérifier l'arrêt
docker ps | grep -E "(wireguard|webserver)" || echo "Services arrêtés avec succès"

# Sauvegarder les logs pour investigation
docker logs wireguard > ~/emergency-wireguard-logs-$(date +%Y%m%d-%H%M%S).log
```

Réinitialisation complète d'urgence

```
# Procédure de réinitialisation complète
echo "URGENCE: Réinitialisation complète - $(date)" >> ~/vpn-emergency.log
```

```
# 1. Sauvegarder les données importantes
mkdir -p ~/emergency-backup-$(date +%Y%m%d-%H%M%S)
cp -r ~/wireguard-project/config ~/emergency-backup-$(date +%Y%m%d-%H%M%S)/
docker logs wireguard > ~/emergency-backup-$(date +%Y%m%d-%H%M%S)/wireguard.log

# 2. Arrêter et supprimer tous les containers
docker stop wireguard webserver
docker rm wireguard webserver

# 3. Nettoyer les configurations
rm -rf ~/wireguard-project/config/*

# 4. Redéployer depuis zéro
docker run -d \
  --name=wireguard \
  -e PUID=1000 \
  -e PGID=1000 \
  -e TZ=Europe/Paris \
  -e SERVERURL=auto \
  -e PEERS=3 \
  -p 51820:51820/udp \
  -v ~/wireguard-project/config:/config \
  --cap-add=NET_ADMIN \
  --sysctl="net.ipv4.conf.all.src_valid_mark=1" \
  --restart unless-stopped \
  linuxserver/wireguard:latest

docker run -d --name webserver -p 8081:80 nginx

# 5. Attendre la configuration et vérifier
sleep 20
docker ps
docker logs --tail 10 wireguard

echo "Réinitialisation terminée. Nouvelles configurations à distribuer." >> ~/vpn-emergency.log
```

Diagnostic rapide en cas de problème

```
# Script de diagnostic complet
#!/bin/bash
# Fichier : ~/diagnose-vpn.sh

echo "=== DIAGNOSTIC VPN COMPLET - $(date) ==="

echo "1. État des containers :"
docker ps -a | grep -E "(wireguard|webserver)"

echo -e "\n2. Utilisation des ressources :"
docker stats --no-stream

echo -e "\n3. Configuration serveur WireGuard :"
docker exec -it wireguard cat /config/wg_confs/wg0.conf 2>/dev/null || echo "ERREUR: Impossible de lire
la configuration"
```

```
echo -e "\n4. Interface réseau :"
docker exec -it wireguard ip addr show wg0 2>/dev/null || echo "ERREUR: Interface wg0 non trouvée"

echo -e "\n5. Routes réseau :"
docker exec -it wireguard ip route 2>/dev/null || echo "ERREUR: Impossible de lire les routes"

echo -e "\n6. Règles NAT :"
docker exec -it wireguard iptables -t nat -L 2>/dev/null || echo "ERREUR: Impossible de lire les règles NAT"

echo -e "\n7. Clients connectés :"
docker exec -it wireguard wg show 2>/dev/null || echo "ERREUR: Impossible de lire l'état WireGuard"

echo -e "\n8. Test serveur web :"
curl -s -o /dev/null -w "Code HTTP: %{http_code}\n" http://localhost:8081

echo -e "\n9. Logs récents (10 dernières lignes) :"
docker logs --tail 10 wireguard 2>/dev/null || echo "ERREUR: Impossible de lire les logs"

echo -e "\n10. IP locale actuelle :"
ifconfig | grep "inet " | grep -v 127.0.0.1

echo -e "\n=== FIN DIAGNOSTIC ==="
```

8. Checklist de maintenance périodique

Vérifications mensuelles

```
# Actions mensuelles recommandées
echo "=== MAINTENANCE MENSUELLE VPN - $(date) ===" >> ~/vpn-maintenance.log

# 1. Mettre à jour l'image Docker
docker pull linuxserver/wireguard:latest

# 2. Vérifier les mises à jour système
# brew update && brew upgrade (si Homebrew installé)

# 3. Analyser les logs de sécurité
docker logs wireguard | grep -i "error\|failed\|denied" > ~/monthly-security-check-$(date +%Y%m%d).log

# 4. Nettoyer les anciens logs Docker
docker system prune -f

# 5. Vérifier l'intégrité des sauvegardes
ls -la ~/wireguard-backup/ >> ~/vpn-maintenance.log

echo "=== FIN MAINTENANCE MENSUELLE ===" >> ~/vpn-maintenance.log
```