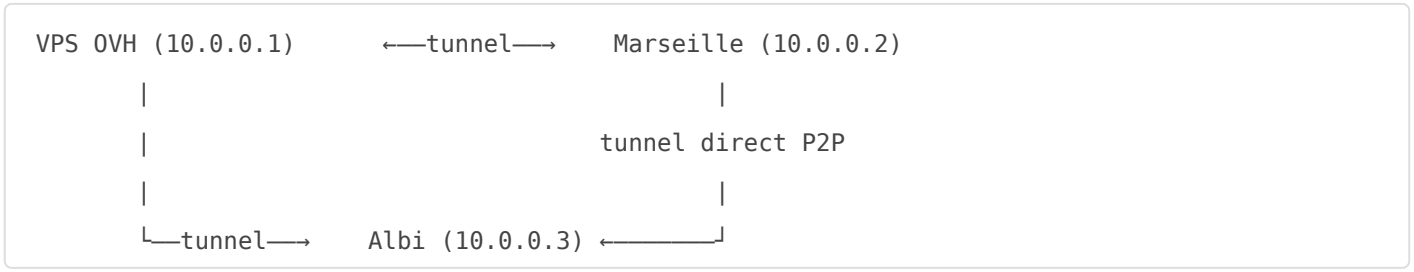
Architecture WireGuard Mesh

Architecture



Plan d'adressage complet

Serveurs (peers fixes)

Serveur	IP WireGuard	IP publique	LAN local	Commentaire
VPS OVH	10.0.0.1	<IP_PUBLIQUE_OVH>	—	—
Marseille	10.0.0.2	<IP_PUBLIQUE_MRS>	192.168.1.0/24	—
Albi	10.0.0.3	<IP_PUBLIQUE_ALBI>	192.168.2.0/24	—

Clients (plages réservées)

Plage	Usage	Profil	Commentaire
10.0.0.10–10.0.0.19	Exit node OVH	ovh-exitnode.conf	IP publique = OVH
10.0.0.20–10.0.0.29	Exit node / LAN Marseille	mrs-exitnode / mrs-lan	IP publique = Marseille
10.0.0.30–10.0.0.39	Exit node / LAN Albi	albi-exitnode / albi-lan	IP publique = Albi
10.0.0.40–10.0.0.49	Réseau privé	private-only.conf	Réseau privé seul, traffic internet intact

Ordre de déploiement

Étape 1 — Générer toutes les clés

Sur chaque serveur, générer une paire de clés :

```
# Sur OVH
wg genkey | tee /etc/wireguard/privatekey-ovh | wg pubkey > /etc/wireguard/publickey-ovh

# Sur Marseille
wg genkey | tee /etc/wireguard/privatekey-mrs | wg pubkey > /etc/wireguard/publickey-mrs

# Sur Albi
wg genkey | tee /etc/wireguard/privatekey-albi | wg pubkey > /etc/wireguard/publickey-albi

# Pour chaque client
wg genkey | tee privatekey-client | wg pubkey > publickey-client
```

Étape 2 — Remplacer les placeholders

Dans tous les fichiers .conf, remplacer :

- `<CLE_PRIVEE_OVH>` → contenu de privatekey-ovh
- `<CLE_PUBLIQUE_OVH>` → contenu de publickey-ovh
- `<CLE_PRIVEE_MRS>` → contenu de privatekey-mrs
- `<CLE_PUBLIQUE_MRS>` → contenu de publickey-mrs
- `<CLE_PRIVEE_ALBI>` → contenu de privatekey-albi
- `<CLE_PUBLIQUE_ALBI>` → contenu de publickey-albi
- `<IP_PUBLIQUE_OVH>` → IP publique du VPS OVH
- `<IP_PUBLIQUE_MRS>` → IP publique de la box Marseille
- `<IP_PUBLIQUE_ALBI>` → IP publique de la box Albi
- (idem pour les clés clients)

Étape 3 — Configuration des box internet

Box Marseille :

1. Port forwarding : UDP 51820 → 192.168.1.100 (IP locale du serveur)
2. Route statique : 10.0.0.0/24 via 192.168.1.100

Box Albi :

1. Port forwarding : UDP 51820 → 192.168.2.100 (IP locale du serveur)
2. Route statique : 10.0.0.0/24 via 192.168.2.100

Étape 4 — Démarrer dans l'ordre

```
# 1. Démarrer le hub OVH en premier
ssh ovh "cd /opt/wireguard && docker compose up -d"

# 2. Démarrer Marseille
ssh marseille "cd /opt/wireguard && docker compose up -d"

# 3. Démarrer Albi
ssh albi "cd /opt/wireguard && docker compose up -d"
```

Étape 5 — Configurer Portainer UI

1. Ouvrir http://10.0.0.1:9000 avec le profil private-only.conf actif
2. Aller dans Environments > Add environment > Agent
3. Ajouter Marseille : URL = `10.0.0.2:9001`, nom = "Marseille"
4. Ajouter Albi : URL = `10.0.0.3:9001`, nom = "Albi"

Commandes utiles

Vérifier l'état des tunnels

```
# Sur n'importe quel serveur
docker exec wireguard-ovh wg show       # OVH
docker exec wireguard-mrs wg show       # Marseille
docker exec wireguard-albi wg show      # Albi

# Résultat attendu : chaque peer doit avoir un "latest handshake" récent
# et un compteur de bytes "received/sent" qui monte
```

Tester la connectivité mesh

```
# Depuis OVH, pinger Marseille et Albi
docker exec wireguard-ovh ping 10.0.0.2 -c 3
docker exec wireguard-ovh ping 10.0.0.3 -c 3

# Depuis Marseille, pinger Albi directement (tunnel P2P)
docker exec wireguard-mrs ping 10.0.0.3 -c 3

# Vérifier que le LAN de Marseille est accessible depuis Albi
docker exec wireguard-albi ping 192.168.1.1 -c 3
```

Vérifier que Portainer UI n'est pas exposé

```
# Cette commande doit échouer (connexion refusée) depuis internet
curl -v http://<IP_PUBLIQUE_OVH>:9000

# Ceci doit fonctionner uniquement avec le VPN actif
curl -v http://10.0.0.1:9000
```

Tester une sauvegarde rsync inter-serveurs

```
# Depuis Marseille vers Albi (tunnel P2P direct)
docker run --rm --network host \
  -v /data/backups:/source:ro \
  instrumentisto/rsync-ssh \
  rsync -avz --progress /source/ user@10.0.0.3:/backup/marseille/

# Vérifier que le trafic ne passe PAS par OVH :
# Sur OVH pendant le rsync : docker exec wireguard-ovh wg show
# → les bytes du peer MRS et Albi ne doivent PAS augmenter
```

Générer un QR code pour mobile (iOS/Android)

```
# Installer qrencode
apt install qrencode
```

```
# Générer le QR pour le profil OVH exit node
qrencode -t ansiutf8 < clients/ovh-exitnode.conf

# Scanner avec l'app WireGuard sur le téléphone
```

Redémarrer un tunnel après modification de config

```
docker compose restart wireguard
# OU
docker exec wireguard-ovh wg syncconf wg0 <(wg-quick strip wg0)
```

Structure des fichiers

```
wireguard-configs/
├── servers/
│   ├── ovh-wg0.conf           # Config WireGuard du VPS OVH
│   ├── marseille-wg0.conf     # Config WireGuard de Marseille
│   └── albi-wg0.conf          # Config WireGuard d'Albi
├── clients/
│   ├── ovh-exitnode.conf      # Client : IP publique = OVH
│   ├── mrs-exitnode.conf      # Client : IP publique = Marseille
│   ├── albi-exitnode.conf     # Client : IP publique = Albi
│   ├── mrs-lan.conf           # Client : accès LAN Marseille (split tunnel)
│   ├── albi-lan.conf          # Client : accès LAN Albi (split tunnel)
│   └── private-only.conf      # Client : réseau privé uniquement
├── docker-compose-all.yml     # docker-compose OVH + Marseille/Albi
└── README.md                  # Ce fichier
```

Sécurité — points d'attention

1. **Ne jamais committer les clés privées** dans Git. Ajouter `*privatekey*` au `.gitignore`.

2. **Portainer UI bind sur 10.0.0.1** uniquement — vérifier avec `ss -tlnp | grep 9000`.
3. **Portainer Agent bind sur 10.0.0.x** uniquement — même vérification sur chaque serveur.
4. **Les fichiers .conf clients** contiennent des clés privées — les traiter comme des mots de passe.
5. **Renouveler les clés** si un appareil client est perdu ou compromis : supprimer son [Peer] de tous les serveurs et faire `docker compose restart wireguard`.

Revision #3
Created 29 May 2026 13:24:18 by gpatruno
Updated 29 May 2026 13:46:25 by gpatruno