

Immich Installation, administration, migration et sauvegarde

Guide complet pour installer, exploiter, sécuriser, sauvegarder, migrer, restaurer et dépanner Immich avec Docker Compose ou TrueNAS.

- [1 - Présentation et architecture](#)
 - [1.1 - Rôle, fonctions et limites d'Immich](#)
 - [1.2 - Architecture et services](#)
 - [1.3 - Prérequis et dimensionnement](#)
 - [1.4 - Données critiques et arborescence](#)
- [2 - Installation avec Docker Compose](#)
 - [2.1 - Préparer Linux et Docker](#)
 - [2.2 - Télécharger Compose et configurer .env](#)
 - [2.3 - Démarrer et contrôler les conteneurs](#)
 - [2.4 - Première connexion et configuration initiale](#)
- [3 - Installation sur TrueNAS](#)
 - [3.1 - Concevoir les datasets Immich](#)
 - [3.2 - Installer et configurer l'application](#)

- [3.3 - Permissions, UID, GID et ACL](#)
- [3.4 - Stockages multiples et bibliothèques externes](#)
- [4 - Configuration et utilisation quotidienne](#)
 - [4.1 - Utilisateurs, quotas et modèle de stockage](#)
 - [4.2 - Application mobile et sauvegarde automatique](#)
 - [4.3 - Créer et gérer une bibliothèque externe](#)
 - [4.4 - Tâches, files d'attente et intégrité](#)
- [5 - Accès distant et sécurité](#)
 - [5.1 - Reverse proxy Caddy](#)
 - [5.2 - Cloudflare, DNS et accès distant](#)
 - [5.3 - Authentification OIDC avec Authentik](#)
 - [5.4 - Sécuriser les comptes, sessions et API](#)
- [6 - Performances et accélération matérielle](#)
 - [6.1 - Transcodage NVIDIA avec une Quadro P2000](#)
 - [6.2 - Machine learning avec CUDA](#)
 - [6.3 - Concurrence, tâches et choix du stockage](#)
 - [6.4 - Surveillance, journaux et contrôles](#)
- [7 - Mises à jour et maintenance](#)
 - [7.1 - Préparer une mise à jour](#)
 - [7.2 - Mettre à jour Docker Compose](#)
 - [7.3 - Mettre à jour TrueNAS et migrer vers VectorChord](#)
 - [7.4 - Commandes d'administration et récupération](#)
- [8 - Stratégie de sauvegarde](#)
 - [8.1 - Périmètre et stratégie 3-2-1](#)
 - [8.2 - Sauvegarder PostgreSQL](#)
 - [8.3 - Snapshots ZFS et réplication TrueNAS](#)
 - [8.4 - Copie distante vers Proxmox et validation](#)
- [9 - Migration complète d'Immich](#)
 - [9.1 - Inventaire et plan de migration](#)

- [9.2 - Préparer et exporter le serveur source](#)
- [9.3 - Transférer et préparer la destination](#)
- [9.4 - Cas pratique Proxmox vers TrueNAS](#)
- [10 - Restauration et dépannage](#)
 - [10.1 - Restauration par l'interface et l'onboarding](#)
 - [10.2 - Restauration PostgreSQL en ligne de commande](#)
 - [10.3 - Plan de reprise après sinistre](#)
 - [10.4 - Erreurs fréquentes et diagnostic](#)

1 - Présentation et architecture

Comprendre Immich, ses composants, ses besoins et les données à protéger.

1.1 - Rôle, fonctions et limites d'Immich

Immich est une plateforme auto-hébergée destinée à sauvegarder, parcourir, rechercher et partager des photos et vidéos. Elle fournit une interface Web, des applications mobiles, une chronologie, des albums, la reconnaissance faciale, la recherche intelligente, une carte et des fonctions de partage.

Ce qu'Immich stocke

- les fichiers originaux envoyés depuis le Web, le mobile ou la CLI ;
- les miniatures et aperçus générés ;
- les vidéos transcodées ;
- les images de profil ;
- les sauvegardes automatiques de PostgreSQL ;
- les métadonnées dans PostgreSQL : utilisateurs, chemins, albums, visages, favoris, partages et paramètres.

Deux types de bibliothèques

Type	Fonctionnement
Bibliothèque interne	Immich reçoit et organise les fichiers envoyés par les utilisateurs
Bibliothèque externe	Immich indexe des fichiers déjà présents dans un dossier monté

Une bibliothèque externe ne transforme pas automatiquement un simple dossier en sauvegarde Immich. Les métadonnées ajoutées dans Immich peuvent rester uniquement dans PostgreSQL. Un déplacement de fichier externe peut être interprété comme une suppression puis un nouvel ajout.

Limites à connaître

- Immich ne remplace pas une stratégie de sauvegarde 3-2-1.
- Une sauvegarde PostgreSQL seule ne contient aucune photo ni vidéo.
- Copier uniquement les médias ne préserve pas les albums, utilisateurs, visages et partages.
- Immich ne rescane pas sa bibliothèque interne pour reconstruire automatiquement toute la base.
- Un retour vers une version plus ancienne du serveur n'est pas pris en charge.
- Le serveur et les applications mobiles doivent rester sur des versions majeures compatibles.
- Les fichiers de la bibliothèque interne ne doivent pas être modifiés directement sur le disque.

Bon usage

1. Envoyer les nouvelles photos avec l'application mobile, le Web ou la CLI.
2. Utiliser une bibliothèque externe pour consulter une photothèque existante.
3. Sauvegarder ensemble PostgreSQL, les fichiers Immich et la configuration.
4. Tester régulièrement une restauration sur une instance isolée.
5. Lire les notes de version avant chaque mise à jour majeure.

La dernière version stable vérifiée lors de la rédaction est Immich v3.1.0. Une installation plus ancienne, comme v2.7.5, doit être sauvegardée et migrée en respectant les changements de version.

Références :

- <https://docs.immich.app/>
- <https://docs.immich.app/administration/backup-and-restore/>
- <https://github.com/immich-app/immich/releases/latest>

1.2 - Architecture et services

Une installation Immich standard repose sur plusieurs services. Le nom exact des conteneurs varie entre Docker Compose, TrueNAS et les versions du catalogue.

Composants principaux

Composant	Rôle
<code>immich-server</code>	interface Web, API, uploads et traitements en arrière-plan
PostgreSQL	utilisateurs, métadonnées, albums, chemins et index vectoriels
VectorChord	extension PostgreSQL utilisée pour la recherche vectorielle
Redis	file d'attente et coordination des tâches
<code>immich-machine-learning</code>	recherche intelligente et reconnaissance faciale
Stockage <code>/data</code>	originaux, miniatures, profils, vidéos encodées et sauvegardes

Les anciennes installations peuvent utiliser `pgvector.rs` et des services nommés `pgvector` ou `pgvector_upgrade`. Les versions récentes utilisent l'image PostgreSQL Immich avec VectorChord.

Flux simplifié

1. Le mobile ou le navigateur envoie un fichier au serveur.
2. Le serveur écrit l'original dans le stockage.
3. Le serveur enregistre le chemin et les métadonnées dans PostgreSQL.
4. Redis distribue les tâches en arrière-plan.
5. Les workers extraient les métadonnées, créent les miniatures et transcodent les vidéos.
6. Le service de machine learning calcule les informations de recherche et de visages.

Workers

Le conteneur serveur comprend généralement :

- `api` pour les requêtes Web et mobiles ;
- `microservices` pour les miniatures, vidéos et autres tâches.

Sur une petite installation, les deux restent dans le même service. Les séparer est réservé aux besoins avancés de répartition ou de limitation des ressources.

Vérifier les services Docker

```
docker compose ps
docker ps --format 'table {{.Names}}\t{{.Image}}\t{{.Status}}'
docker compose logs --tail=100
```

Sur TrueNAS avec accès au shell :

```
sudo docker ps --format 'table {{.Names}}\t{{.Image}}\t{{.Status}}' | grep -i immich
```

Ne pas supposer un nom de conteneur. L'identifier avant d'utiliser `docker logs`, `docker exec` ou `pg_dump`.

Dépendances critiques

Immich ne peut pas fonctionner correctement si :

- PostgreSQL est indisponible ou incompatible ;
- les dossiers de `/data` sont absents ou non accessibles ;
- Redis ne peut pas traiter les files d'attente ;
- le reverse proxy bloque les gros envois ;
- une migration de base a été interrompue.

Références :

- <https://docs.immich.app/developer/architecture/>
- <https://docs.immich.app/administration/jobs-workers/>

1.3 - Prérequis et dimensionnement

Configuration minimale et recommandée

La documentation Immich indique actuellement :

Ressource	Minimum	Recommandation de départ
CPU	2 coeurs	4 coeurs ou davantage
Mémoire	6 Go	8 Go ou davantage
Système	Linux 64 bits	Debian, Ubuntu, TrueNAS ou autre Linux pris en charge
Base PostgreSQL	stockage local	SSD recommandé

Le machine learning, les imports massifs et la génération de miniatures peuvent nécessiter davantage de mémoire et de CPU. Sur TrueNAS, prévoir plus de 8 Go de mémoire pour l'application si le machine learning est activé.

Stockage

Prévoir de l'espace pour :

- les fichiers originaux ;
- les miniatures et aperçus ;
- les vidéos encodées ;
- PostgreSQL ;
- les modèles de machine learning ;
- les sauvegardes locales ;
- la marge nécessaire pendant une migration.

La taille générée dépend fortement des vidéos et des paramètres. Conserver une marge libre suffisante et surveiller séparément l'espace et les inodes.

```
df -hT
df -i
du -sh /chemin/immich/*
```

Réseau

- réseau local stable entre le mobile, le serveur et le stockage ;
- accès HTTPS pour l'extérieur ;
- délais suffisants dans le reverse proxy ;
- upload maximal adapté aux longues vidéos ;
- DNS local et public cohérent si un domaine est utilisé.

GPU

Le GPU est optionnel. Il peut accélérer :

- le transcodage vidéo avec NVENC, Quick Sync ou VAAPI ;
- le machine learning avec CUDA, OpenVINO, ROCm ou un autre moteur pris en charge.

La Quadro P2000 prend en charge NVENC et possède une capacité de calcul CUDA suffisante pour Immich. Le serveur doit néanmoins disposer du pilote NVIDIA officiel et, pour Docker Compose, du NVIDIA Container Toolkit.

Virtualisation

Immich fonctionne dans une machine virtuelle complète. Docker dans un conteneur LXC n'est pas recommandé par le projet pour un déploiement standard. Pour une migration Proxmox, préférer une VM Linux ou déplacer directement Immich vers l'application TrueNAS.

Liste de préparation

- ☐ processeur et mémoire suffisants ;
- ☐ stockage de la base sur SSD si possible ;

- ☐ capacité pour les originaux et les sauvegardes ;
- ☐ heure et DNS corrects ;
- ☐ sauvegarde distante disponible ;
- ☐ accès administrateur au système et à Immich ;
- ☐ versions source et destination identifiées.

Référence : <https://docs.immich.app/install/requirements/>

1.4 - Données critiques et arborescence

Immich utilise six dossiers principaux dans son emplacement de médias, généralement monté dans le conteneur sous `/data`.

Arborescence

Dossier	Contenu	Criticité
<code>upload</code>	originaux envoyés lorsque le modèle de stockage est désactivé	critique
<code>library</code>	originaux organisés lorsque le modèle de stockage est activé	critique si utilisé
<code>profile</code>	images de profil	critique
<code>thumbs</code>	miniatures et aperçus générés	régénérable
<code>encoded-video</code>	vidéos transcodées	régénérable
<code>backups</code>	sauvegardes automatiques de PostgreSQL	critique avec les médias

Exemple Docker Compose :

```
UPLOAD_LOCATION=/srv/immich/data
DB_DATA_LOCATION=/srv/immich/postgres
```

Exemple TrueNAS adapté à ton environnement :

```
/mnt/HDD_DATA_TRUENAS/IMMICH/data
/mnt/HDD_DATA_TRUENAS/IMMICH/pgData
```

Ces sous-dossiers sont une organisation recommandée. Vérifier l'installation existante avant de créer ou déplacer quoi que ce soit.

La base ne contient pas les photos

PostgreSQL contient les chemins, métadonnées et relations. Une sauvegarde `.sql.gz` ne contient pas les originaux. Une restauration complète exige donc :

1. une sauvegarde PostgreSQL compatible ;
2. une copie cohérente des dossiers de médias ;
3. la configuration et les montages ;
4. les bibliothèques externes accessibles aux mêmes chemins vus par le conteneur.

Bibliothèque externe

Une bibliothèque externe doit être montée sous un chemin distinct, par exemple :

```
Hôte TrueNAS : /mnt/HDD_DATA_TRUENAS/PHOTOS
Conteneur    : /mnt/external-libraries/photos
```

Le chemin saisi dans l'interface Immich est celui du conteneur :

```
/mnt/external-libraries/photos
```

Identifier les chemins réels

Docker Compose :

```
grep -E '^(UPLOAD_LOCATION|DB_DATA_LOCATION|IMMICH_VERSION)= ' .env
docker inspect immich_server --format '{{json .Mounts}}'
```

TrueNAS :

```
sudo docker inspect NOM_CONTENEUR_IMMICH --format '{{json .Mounts}}'
sudo findmnt | grep -i immich
```

Ne pas modifier directement les fichiers générés sous `/mnt/.ix-apps/app_configs/`. TrueNAS peut les recréer lors d'une mise à jour de l'application.

Références :

- <https://docs.immich.app/administration/backup-and-restore/>
- <https://docs.immich.app/install/truenas/>

2 - Installation avec Docker Compose

Déployer Immich sur un serveur Linux avec la méthode recommandée Docker Compose.

2.1 - Préparer Linux et Docker

Docker Compose est la méthode recommandée par Immich pour un déploiement Linux standard.

Préparer le serveur

```
sudo apt update
sudo apt upgrade
sudo apt install ca-certificates curl wget openssl rsync
```

Installer Docker Engine depuis le dépôt officiel Docker correspondant à la distribution. Vérifier ensuite :

```
docker --version
docker compose version
docker info
```

La commande attendue est `docker compose` avec un espace. L'ancien exécutable `docker-compose` peut être incompatible avec le fichier officiel actuel.

Créer l'arborescence

```
sudo mkdir -p /srv/immich/{app,data,postgres}
sudo chown -R "$USER":"$USER" /srv/immich/app
```

Définir séparément les permissions des données et de PostgreSQL selon les UID réellement utilisés par les conteneurs. Ne pas appliquer `chmod 777`.

Vérifier le stockage

```
df -hT /srv/immich
df -i /srv/immich
findmnt /srv/immich
```

La base PostgreSQL doit rester sur un stockage local fiable. Les partages réseau ne sont pas pris en charge pour `DB_DATA_LOCATION` dans le déploiement Compose officiel.

Heure et réseau

```
timedatectl
getent hosts ghcr.io
curl -I https://github.com
```

Le serveur doit pouvoir télécharger les images depuis GitHub Container Registry et les modèles de machine learning.

Autoriser Docker sans sudo

Option facultative :

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

Fermer puis rouvrir la session. L'appartenance au groupe Docker donne des privilèges proches de root. Elle doit rester limitée aux administrateurs.

Contrôles avant installation

```
docker run --rm hello-world
docker compose version
free -h
lscpu
```

Si un GPU NVIDIA sera utilisé :

```
nvidia-smi
docker run --rm --gpus all nvidia/cuda:12.3.2-base-ubuntu22.04 nvidia-smi
```

Le second test nécessite le NVIDIA Container Toolkit et une image disponible. Adapter la version CUDA au pilote installé si nécessaire.

Références :

- <https://docs.immich.app/install/docker-compose/>
- <https://docs.immich.app/install/requirements/>

2.2 - Télécharger Compose et configurer .env

Télécharger les fichiers officiels

```
cd /srv/immich/app

wget -O docker-compose.yml \
  https://github.com/immich-app/immich/releases/latest/download/docker-compose.yml

wget -O .env \
  https://github.com/immich-app/immich/releases/latest/download/example.env
```

Conserver une copie avant modification :

```
cp -a docker-compose.yml docker-compose.yml.original
cp -a .env .env.original
```

Configurer .env

```
UPLOAD_LOCATION=/srv/immich/data
DB_DATA_LOCATION=/srv/immich/postgres
TZ=Europe/Paris
IMMICH_VERSION=v3
DB_PASSWORD=REPLACER_PAR_UN_SECRET_ALPHANUMERIQUE
DB_USERNAME=postgres
DB_DATABASE_NAME=immich
```

La documentation recommande un mot de passe PostgreSQL composé de lettres et chiffres afin d'éviter les problèmes d'interprétation par Docker.

Générer un secret :

```
openssl rand -hex 24
```

Ne pas afficher ou partager le fichier `.env` complet après y avoir ajouté le secret.

Version fixe ou version majeure

Pour une nouvelle installation stable :

```
IMMICH_VERSION=v3
```

Pour restaurer une ancienne instance v2.7.5, il peut être préférable de démarrer avec la version exacte de la source :

```
IMMICH_VERSION=v2.7.5
```

Après validation de la restauration, lire les notes de version et effectuer la montée vers v3. Ne jamais tenter un downgrade d'une base déjà migrée vers une version plus récente.

Valider le fichier Compose

```
docker compose config --quiet  
docker compose config > /tmp/immich-compose-rendu.yml
```

La seconde commande développe les variables. Le fichier rendu peut donc contenir des secrets et doit rester local puis être supprimé après contrôle.

Protéger la configuration

```
chmod 600 .env  
chmod 644 docker-compose.yml
```

Sauvegarder `.env` et `docker-compose.yml` dans un emplacement chiffré. Le mot de passe de base et les chemins sont nécessaires pour une reconstruction fidèle.

Référence : <https://docs.immich.app/install/docker-compose/>

2.3 - Démarrer et contrôler les conteneurs

Démarrer Immich

Depuis le dossier contenant `.env` et `docker-compose.yml` :

```
cd /srv/immich/app
docker compose pull
docker compose up -d
```

Contrôler l'état

```
docker compose ps
docker compose images
docker compose logs --tail=100
```

Suivre le serveur :

```
docker compose logs -f immich-server
```

Quitter le suivi avec `Ctrl+C`. Cela n'arrête pas le conteneur.

Tester le port

Le déploiement Compose officiel utilise normalement le port 2283 :

```
ss -ltnp | grep 2283
curl -I http://127.0.0.1:2283
```

Ouvrir ensuite :

```
http://ADRESSE_DU_SERVEUR:2283
```

Examiner un service en erreur

```
docker compose ps -a
docker compose logs --tail=200 database
docker compose logs --tail=200 immich-server
docker compose logs --tail=200 immich-machine-learning
```

Le service PostgreSQL peut porter un autre nom dans une ancienne configuration. Utiliser :

```
docker ps --format 'table {{.Names}}\t{{.Image}}\t{{.Status}}'
```

Vérifier les montages

```
docker inspect immich_server --format '{{range .Mounts}}{{println .Source "->"
.Destination}}{{end}}'
docker exec -it immich_server sh -lc 'ls -la /data'
```

Les dossiers suivants doivent être accessibles :

```
/data/upload
/data/library
/data/thumbs
/data/profile
/data/encoded-video
/data/backups
```

Commandes de cycle de vie

```
docker compose stop
docker compose start
docker compose restart immich-server
```

```
docker compose down
```

`docker compose down` supprime les conteneurs et réseaux du projet, mais les bind mounts externes restent normalement présents. Ne pas ajouter `-v` sans avoir vérifié les volumes, car cette option supprime les volumes gérés par Docker.

Après modification de .env

Un simple redémarrage ne remplace pas toujours l'environnement du conteneur :

```
docker compose up -d
```

Si nécessaire :

```
docker compose up -d --force-recreate
```

Référence : <https://docs.immich.app/install/environment-variables/>

2.4 - Première connexion et configuration initiale

Créer le compte administrateur

Le premier compte enregistré devient administrateur. Accéder à l'interface, cliquer sur le bouton de démarrage et créer ce compte avec une adresse maîtrisée et un mot de passe unique.

Créer ensuite les autres utilisateurs depuis :

Administration > Users

Régler les paramètres essentiels

Dans Administration > Settings :

1. vérifier le fuseau horaire et les paramètres généraux ;
2. configurer la sauvegarde automatique PostgreSQL ;
3. choisir le comportement de transcodage vidéo ;
4. vérifier les paramètres de machine learning ;
5. définir la périodicité des bibliothèques externes ;
6. examiner les tâches nocturnes et l'intégrité.

Modèle de stockage

Le modèle de stockage peut organiser les fichiers selon une structure lisible. Exemple :

```
{{y}}/{{y}}-{{MM}}-{{dd}}/{{filename}}
```

Tester le résultat affiché dans Immich avant d'appliquer une migration de stockage à une grande bibliothèque. Le changement déplace des fichiers et exige des permissions correctes.

Application mobile

1. Installer l'application officielle Android ou iOS.
2. Saisir l'URL complète du serveur, idéalement en HTTPS.
3. Se connecter.
4. Choisir les albums à sauvegarder.
5. Vérifier les règles Wi-Fi, données mobiles et arrière-plan.
6. Laisser l'application ouverte lors du premier envoi massif.

Premier test de bout en bout

- envoyer une petite photo depuis le mobile ;
- vérifier sa présence dans la chronologie ;
- ouvrir la photo et contrôler les métadonnées ;
- regarder les tâches dans `Administration > Jobs` ;
- vérifier la création de miniatures ;
- tester la recherche après le traitement machine learning ;
- déclencher une sauvegarde de base manuelle.

Contrôle serveur

```
docker compose ps
docker compose logs --tail=100 immich-server
find /srv/immich/data -maxdepth 2 -type f | head
```

Sauvegarde initiale

Avant d'importer toute la photothèque :

- sauvegarder `.env` et `docker-compose.yml` ;
- déclencher un dump PostgreSQL ;
- vérifier le chemin réel des médias ;
- préparer la copie distante ;

- documenter la version Immich.

Références :

- <https://docs.immich.app/install/post-install/>
- <https://docs.immich.app/features/mobile-backup/>

3 - Installation sur TrueNAS

Préparer les datasets, permissions, ressources et stockages de l'application Immich sur TrueNAS.

3.1 - Concevoir les datasets Immich

Organisation recommandée

Pour TrueNAS Community Edition, Immich utilise au minimum deux zones de stockage :

Dataset	Rôle	Stockage conseillé
data	médias et sauvegardes automatiques	pool de grande capacité
pgData	base PostgreSQL	SSD si possible

Exemple adapté à ton pool :

```
/mnt/HDD_DATA_TRUENAS/IMMICH/data  
/mnt/HDD_DATA_TRUENAS/IMMICH/pgData
```

Il est possible de créer un parent `IMMICH`, puis les deux datasets enfants. Ne pas transformer un dossier existant contenant des données en dataset sans plan de copie et sauvegarde préalable.

Création dans l'interface

1. Ouvrir `Datasets`.
2. Sélectionner le pool.
3. Créer le parent `IMMICH` avec le preset Generic.
4. Créer `data` avec le preset Apps.
5. Créer `pgData` avec le preset Generic.
6. Vérifier les chemins complets.

Le preset Apps prépare normalement les permissions pour l'utilisateur `apps`.

Propriétaires attendus

- `data` doit être modifiable par l'utilisateur qui exécute Immich, par défaut `apps` UID 568 et GID 568.
- `pgData` doit appartenir à l'utilisateur utilisé par PostgreSQL. La documentation TrueNAS actuelle indique `netdata` UID 999 pour le dataset PostgreSQL de l'application.

Ces valeurs peuvent évoluer avec le catalogue. Vérifier l'écran d'installation et les conteneurs avant de changer un propriétaire.

Contrôles depuis le shell

```
sudo ls -ldn /mnt/HDD_DATA_TRUENAS/IMMICH
sudo ls -ldn /mnt/HDD_DATA_TRUENAS/IMMICH/data
sudo ls -ldn /mnt/HDD_DATA_TRUENAS/IMMICH/pgData
sudo zfs list | grep -i immich
```

Datasets supplémentaires

Immich utilise :

```
library
upload
thumbs
profile
encoded-video
backups
```

Il est possible de créer un dataset pour certains dossiers, par exemple les miniatures sur SSD. Chaque dataset supplémentaire augmente toutefois la complexité des montages, permissions, snapshots et sauvegardes.

Propriétés ZFS

Recommandations générales à adapter :

- compression ZFS activée ;
- snapshots réguliers ;
- pas de déduplication ZFS sans besoin démontré ;
- quotas uniquement avec une marge suffisante ;
- PostgreSQL sur un stockage à faible latence.

Ne pas modifier au hasard `recordsize`, `sync` ou d'autres propriétés de fiabilité. Mesurer et comprendre l'impact avant tout réglage avancé.

Référence : <https://docs.immich.app/install/truenas/>

3.2 - Installer et configurer l'application

Installer l'application

- 1. Ouvrir `Apps`.
- 2. Cliquer sur `Discover Apps`.
- 3. Rechercher `Immich`.
- 4. Ouvrir l'application puis cliquer sur `Install`.

La méthode TrueNAS est une contribution communautaire intégrée à la documentation Immich. Elle doit être maintenue via le catalogue TrueNAS, pas en modifiant les fichiers générés du projet Compose.

Paramètres principaux

Paramètre	Exemple
Nom	<code>immich</code>
Fuseau horaire	<code>Europe/Paris</code>
Port Web	<code>30041</code> par défaut sur l'application TrueNAS
Stockage de données	<code>/mnt/HDD_DATA_TRUENAS/IMMICH/data</code>
Stockage PostgreSQL	<code>/mnt/HDD_DATA_TRUENAS/IMMICH/pgData</code>
Machine learning	activé si les ressources le permettent

Dans le conteneur, les données doivent être montées à l'emplacement attendu par l'application, généralement `/data`.

Ressources

La documentation TrueNAS indique au moins 6 Go de mémoire pour Immich et recommande davantage avec le machine learning. Pour une bibliothèque importante :

- 4 à 8 threads CPU selon les ressources du serveur ;
- 8 Go de mémoire ou davantage ;
- base sur SSD si disponible ;
- GPU configuré seulement après un fonctionnement correct sur CPU.

Les limites CPU et mémoire ne sont pas des réservations. Éviter de priver TrueNAS et ZFS de mémoire.

Premier démarrage

Une fois l'état `Running`, ouvrir :

```
http://ADRESSE_TRUENAS:30041
```

Vérifier depuis le shell :

```
sudo docker ps --format 'table {{.Names}}\t{{.Image}}\t{{.Status}}' | grep -i immich
sudo ss -ltnp | grep 30041
```

Journaux

Découvrir les noms :

```
sudo docker ps -a --format '{{.Names}} {{.Image}} {{.Status}}' | grep -i immich
```

Puis :

```
sudo docker logs --tail 200 NOM_CONTENEUR
```

Ton installation observée exposait le port interne 2283 sur le port hôte 30041 et utilisait un projet Compose nommé `ix-immich`.

Modifier les paramètres

Utiliser `Apps > Installed Applications > Immich > Edit`. TrueNAS recrée ensuite les conteneurs. Ne pas modifier durablement :

```
/mnt/.ix-apps/app_configs/immich/versions/.../templates/rendered/
```

Ces fichiers peuvent être écrasés lors d'une mise à jour ou d'un redéploiement.

Référence : <https://docs.immich.app/install/truenas/>

3.3 - Permissions, UID, GID et ACL

Les erreurs de permissions empêchent souvent Immich ou PostgreSQL de démarrer. Identifier les UID et GID réellement utilisés avant toute correction.

Valeurs courantes

Élément	Compte courant
Application Immich	<code>apps</code> , UID 568, GID 568
PostgreSQL de l'application	UID 999 selon le guide TrueNAS actuel

Ces valeurs sont des références et non une raison d'appliquer un changement récursif sans vérification.

Identifier les utilisateurs des conteneurs

```
sudo docker inspect NOM_CONTENEUR_IMMICH --format '{{.Config.User}}'
sudo docker exec NOM_CONTENEUR_IMMICH id
sudo docker exec NOM_CONTENEUR_POSTGRES id
```

Examiner les datasets

```
sudo stat -c '%u:%g %a %n' /mnt/HDD_DATA_TRUENAS/IMMICH/data
sudo stat -c '%u:%g %a %n' /mnt/HDD_DATA_TRUENAS/IMMICH/pgData
```

```
sudo getfacl /mnt/HDD_DATA_TRUENAS/IMMICH/data
```

Corriger avec l'interface

Préférer l'éditeur ACL TrueNAS :

1. sélectionner le dataset ;
2. ouvrir les permissions ;
3. donner `Modify` à l'utilisateur ou groupe de l'application ;
4. préserver les accès nécessaires aux sauvegardes ;
5. appliquer récursivement uniquement si le contenu et l'effet sont compris.

Pour un dataset SMB/NFSv4 utilisé avec le modèle de stockage, la documentation TrueNAS demande un mode ACL `Passthrough` afin qu'Immich puisse effectuer les changements nécessaires.

Bibliothèque externe en lecture seule

L'utilisateur Immich a besoin au minimum de lecture et traversée :

```
sudo -u '#568' find /mnt/HDD_DATA_TRUENAS/PHOTOS -maxdepth 1 -type f -print | head
```

Le test peut échouer si l'utilisateur 568 n'existe pas dans la base locale. Dans ce cas, contrôler depuis le conteneur après montage :

```
sudo docker exec NOM_CONTENEUR_IMMICH sh -lc 'id; ls -la /mnt/external-libraries/photos | head'
```

Erreurs à éviter

- `chmod -R 777` sur les datasets ;
- changer récursivement le propriétaire de `pgData` pendant que PostgreSQL fonctionne ;
- appliquer une ACL SMB incompatible avec les opérations POSIX requises ;
- donner l'écriture à Immich sur une archive devant rester immuable ;
- utiliser le même chemin de montage pour deux bibliothèques différentes.

Après correction, redéployer l'application depuis l'interface et examiner les journaux.

Référence : <https://docs.immich.app/install/truenas/>

3.4 - Stockages multiples et bibliothèques externes

Stockages Immich séparés

Pour chaque dataset supplémentaire, ajouter une entrée `Additional Storage` dans l'application TrueNAS.

Dossier Immich	Chemin dans le conteneur	Exemple hôte
originaux organisés	<code>/data/library</code>	<code>/mnt/HDD_DATA_TRUENAS/IMMICH/library</code>
uploads	<code>/data/upload</code>	<code>/mnt/HDD_DATA_TRUENAS/IMMICH/upload</code>
miniatures	<code>/data/thumbs</code>	<code>/mnt/SSD/IMMICH/thumbs</code>
profils	<code>/data/profile</code>	<code>/mnt/HDD_DATA_TRUENAS/IMMICH/profile</code>
vidéos encodées	<code>/data/encoded-video</code>	<code>/mnt/HDD_DATA_TRUENAS/IMMICH/encoded-video</code>
dumps automatiques	<code>/data/backups</code>	<code>/mnt/HDD_DATA_TRUENAS/IMMICH/backups</code>

Le chemin du conteneur doit commencer par `/data/` et correspondre exactement au dossier attendu.

Bibliothèque externe

Exemple :

Host Path

: /mnt/HDD_DATA_TRUENAS/PHOTOS

Mount Path

: /mnt/external-libraries/photos

Read Only

: activé

Dans Immich, ajouter :

/mnt/external-libraries/photos

Le chemin `/mnt/HDD_DATA_TRUENAS/PHOTOS` n'existe pas forcément dans le conteneur. Il ne doit pas être saisi dans Immich si le montage interne utilise un autre nom.

Lecture seule ou lecture-écriture

Lecture seule est recommandée lorsqu'Immich sert uniquement à indexer une photothèque existante. En lecture-écriture, Immich peut supprimer un original lorsque la corbeille est vidée et peut créer des fichiers XMP selon les fonctions utilisées.

Contrôler depuis le conteneur

```
sudo docker exec NOM_CONTENEUR_IMMICH sh -lc \  
'find /mnt/external-libraries/photos -maxdepth 2 -type f | head'
```

Sauvegardes

Chaque dataset séparé doit être inclus dans :

- les snapshots ;
- la réplication ;
- la copie distante ;
- le plan de restauration ;
- les contrôles d'intégrité.

Si `profile` ou `backups` est déplacé vers un autre dataset, une sauvegarde limitée au dataset `data` ne suffit plus.

Changer un montage existant

1. déclencher un dump PostgreSQL ;
2. arrêter l'application ;
3. copier les données avec préservation des attributs ;
4. vérifier les tailles et les sommes ;
5. modifier le montage dans TrueNAS ;
6. redémarrer ;
7. contrôler l'intégrité et les journaux ;

8. conserver l'ancienne copie jusqu'à validation.

Références :

- <https://docs.immich.app/install/truenas/>
- <https://docs.immich.app/guides/custom-locations/>

4 - Configuration et utilisation quotidienne

Configurer les utilisateurs, les modèles de stockage, la sauvegarde mobile, les bibliothèques et les tâches.

4.1 - Utilisateurs, quotas et modèle de stockage

Gérer les utilisateurs

Le premier compte créé est administrateur. Les comptes supplémentaires se gèrent dans :

Administration > Users

L'administrateur peut créer un compte, réinitialiser son mot de passe, fixer un quota, définir une étiquette de stockage et planifier sa suppression.

Quotas

Un quota limite les uploads internes d'un utilisateur. Les bibliothèques externes ne sont pas comptées dans ce quota.

Vérifier la consommation dans :

Administration > Server Stats

Étiquette de stockage

Sans étiquette, les fichiers utilisent souvent un identifiant utilisateur. Une étiquette rend le chemin plus lisible :

upload/alex/...

Après modification d'une étiquette sur un compte ayant déjà des fichiers, lancer la tâche `Storage Migration`.

Modèle de stockage

Le modèle peut organiser les originaux dans `library`. Exemple :

```
{{y}}/{{y}}-{{MM}}-{{dd}}/{{filename}}
```

Avant application :

- vérifier les permissions du dataset ;
- contrôler l'espace disponible ;
- effectuer une sauvegarde ;
- tester sur une petite bibliothèque ;
- surveiller `Administration > Jobs`.

Suppression d'un utilisateur

La suppression désactive le compte puis planifie l'effacement de ses données selon le délai configuré. Une suppression immédiate des actifs est irréversible depuis l'interface.

Avant toute suppression :

1. vérifier le propriétaire des bibliothèques ;
2. exporter ou transférer les données utiles ;
3. déclencher une sauvegarde de la base ;
4. confirmer le délai de suppression ;
5. documenter la demande.

Comptes administrateurs

Limiter le nombre d'administrateurs. Conserver un compte local de secours avec un mot de passe fort même si Authentik est utilisé, sauf politique contraire et procédure de récupération testée.

Références :

- <https://docs.immich.app/administration/user-management/>
- <https://docs.immich.app/administration/storage-template/>

4.2 - Application mobile et sauvegarde automatique

L'application mobile peut envoyer automatiquement les photos et vidéos sélectionnées vers le serveur Immich.

Configuration initiale

1. Installer l'application officielle.
2. Saisir l'URL HTTPS du serveur.
3. Se connecter avec le compte Immich ou OIDC.
4. Ouvrir les paramètres de sauvegarde.
5. Sélectionner les albums inclus.
6. Exclure les albums inutiles.
7. Choisir Wi-Fi uniquement ou autoriser les données mobiles.

Premier import

Pour une photothèque importante :

- garder le téléphone branché ;
- utiliser un Wi-Fi stable ;
- laisser l'application ouverte au début ;
- désactiver temporairement l'économie d'énergie pour Immich ;
- surveiller l'espace serveur et les files d'attente ;
- éviter une mise à jour serveur pendant l'import.

Déduplication

L'application calcule une somme du contenu. Si l'actif existe déjà dans la bibliothèque de l'utilisateur, l'envoi peut être ignoré. La déduplication n'est pas globale entre tous les utilisateurs et

bibliothèques.

Synchronisation des albums

L'option de synchronisation crée des albums serveur correspondant aux albums mobiles. Il s'agit d'une synchronisation du téléphone vers le serveur, pas d'une réplication bidirectionnelle complète.

Vérifier les résultats

Dans l'application :

- nombre d'éléments sauvegardés ;
- erreurs ou éléments restants ;
- permissions d'accès aux photos ;
- état du traitement en arrière-plan.

Sur le serveur :

```
docker compose logs --tail=100 immich-server  
df -hT /srv/immich/data
```

Dans Immich :

```
Administration > Jobs  
Administration > Server Stats
```

Changement d'URL

Après une migration, mettre à jour l'URL du serveur dans l'application mobile et tester la connexion avant de supprimer l'ancienne instance. Si Authentik est utilisé, les URI de redirection et le domaine doivent également être cohérents.

Important

La présence des photos sur le téléphone et sur Immich crée deux copies, mais ne garantit pas une sauvegarde durable. Une panne, une suppression synchronisée ou une erreur humaine peut toucher les deux. Maintenir une troisième copie hors du serveur.

Référence : <https://docs.immich.app/features/mobile-backup/>

4.3 - Créer et gérer une bibliothèque externe

Une bibliothèque externe indexe des photos et vidéos déjà présentes sur le système de fichiers.

Monter le dossier

Docker Compose :

```
services:
  immich-server:
    volumes:
      - ${UPLOAD_LOCATION}:/data
      - /srv/photos:/mnt/external-libraries/photos:ro
```

Redéploier :

```
docker compose up -d
```

TrueNAS : ajouter un stockage supplémentaire avec un chemin interne distinct, par exemple `/mnt/external-libraries/photos`.

Créer la bibliothèque

1. ouvrir `Administration > External Libraries` ;
2. cliquer sur `Create Library` ;
3. choisir le propriétaire ;
4. ajouter `/mnt/external-libraries/photos` ;
5. ajouter les exclusions si nécessaire ;
6. lancer `Scan New Library Files` ;
7. suivre les tâches.

Le propriétaire ne peut pas être changé directement après la création selon le fonctionnement documenté.

Exclusions

Exemple pour ignorer les fichiers RAW dans un sous-dossier :

```
**/Raw/**
```

Tester les motifs sur une petite partie avant un scan complet.

Lecture seule

Le suffixe `:ro` protège les fichiers contre les modifications du conteneur. C'est le choix le plus prudent pour une archive existante. Certaines modifications de métadonnées nécessitent cependant la création de fichiers XMP et ne fonctionneront pas en lecture seule.

Déplacements et suppressions

- un fichier retiré du chemin est placé dans la corbeille Immich après scan ;
- un fichier déplacé peut être considéré comme un nouvel actif ;
- les albums et descriptions stockés uniquement dans PostgreSQL peuvent être perdus lors d'un déplacement ;
- vider la corbeille d'une bibliothèque en lecture-écriture peut supprimer l'original.

Diagnostic

```
docker exec -it immich_server sh -lc \  
'id; find /mnt/external-libraries/photos -maxdepth 2 -type f | head'
```

Vérifier les montages, permissions, chemins avec `/` et absence de liens symboliques traversant plusieurs montages.

Références :

- <https://docs.immich.app/features/libraries/>

- <https://docs.immich.app/guides/external-library/>

4.4 - Tâches, files d'attente et intégrité

Immich traite les médias avec des tâches en arrière-plan : extraction de métadonnées, miniatures, transcodage vidéo, reconnaissance faciale, recherche intelligente et migration de stockage.

Consulter les tâches

Administration > Jobs

Lors d'un import important, des files actives sont normales. Examiner les journaux si le nombre d'échecs augmente ou si aucune tâche ne progresse.

Ordre de traitement

Après un upload, Immich lance généralement :

1. extraction des métadonnées ;
2. génération de miniatures ;
3. traitement machine learning ;
4. transcodage selon les paramètres ;
5. autres tâches planifiées.

Concurrence

Augmenter la concurrence au-delà du nombre de coeurs peut réduire la réactivité sans accélérer le traitement. Procéder par petites étapes et surveiller CPU, mémoire et latence du stockage.

Intégrité système

Immich vérifie :

- les fichiers présents mais non suivis ;
- les fichiers référencés mais absents ;
- les différences de somme de contrôle.

Les dossiers contrôlés comprennent :

```
upload
library
thumbs
encoded-video
profile
backups
```

Les fichiers `.immich` servent de marqueurs de montage. Ne pas les créer ou supprimer manuellement sans comprendre la cause de l'erreur.

Contrôles serveur

```
docker compose ps
docker compose logs --tail=200 immich-server
docker stats --no-stream
df -hT
df -i
```

Sur TrueNAS :

```
sudo docker logs --tail 200 NOM_CONTENEUR_IMMICH
sudo zpool status
sudo zfs list
```

Réparer sans aggraver

- une miniature manquante peut être régénérée ;
- un original manquant doit être recherché dans les sauvegardes ;
- un checksum différent peut signaler une modification ou une corruption ;
- ne pas supprimer un original directement de la bibliothèque interne ;
- déclencher les tâches manquantes tout en suivant les journaux.

La variable `IMMICH_IGNORE_MOUNT_CHECK_ERRORS=true` désactive une protection importante. Ne l'utiliser que pour un diagnostic maîtrisé, jamais comme correction permanente.

Références :

- <https://docs.immich.app/administration/jobs-workers/>
- <https://docs.immich.app/administration/system-integrity/>

5 - Accès distant et sécurité

Publier Immich en HTTPS, intégrer Authentik et protéger les comptes et clés d'API.

5.1 - Reverse proxy Caddy

Immich doit être publié à la racine d'un domaine ou sous-domaine. Une publication sous `/immich` n'est pas prise en charge.

Exemple Caddy

Pour ton environnement :

```
immich.dreamsaphir.net {  
    reverse_proxy http://ADRESSE_TRUENAS:30041  
}
```

Après modification :

```
caddy validate --config /etc/caddy/Caddyfile  
sudo systemctl reload caddy  
sudo systemctl status caddy --no-pager
```

Si Caddy fonctionne en conteneur, l'adresse `127.0.0.1` désigne le conteneur Caddy lui-même. Utiliser une adresse ou un réseau Docker permettant d'atteindre TrueNAS.

En-têtes attendus

Le proxy doit transmettre :

```
Host  
X-Real-IP  
X-Forwarded-For  
X-Forwarded-Proto
```

Caddy gère normalement les en-têtes de proxy requis. Une configuration personnalisée ne doit pas les supprimer.

Gros fichiers et délais

Les vidéos volumineuses exigent des délais suffisants. Tester un vrai upload depuis l'extérieur, pas seulement l'ouverture de la page de connexion.

Contrôles

```
getent hosts immich.dreamsaphir.net
curl -I https://immich.dreamsaphir.net
curl -I http://ADRESSE_TRUENAS:30041
journalctl -u caddy -n 100 --no-pager
```

Tester également :

```
https://immich.dreamsaphir.net/.well-known/immich
```

Ce point de terminaison doit atteindre Immich, notamment pour éviter des problèmes avec l'application mobile.

Certificat interne

Si Caddy et Immich communiquent sur le réseau local en HTTP, le chiffrement public est terminé par Caddy. Restreindre le port 30041 au réseau interne et ne pas l'exposer directement sur Internet.

Après une migration

1. tester la nouvelle instance par IP locale ;
2. conserver l'ancienne instance arrêtée ou en lecture seule ;
3. modifier la cible Caddy ;
4. recharger Caddy ;
5. vérifier Web, mobile, OAuth et uploads ;
6. conserver un moyen de retour rapide vers l'ancienne cible.

Référence : <https://docs.immich.app/administration/reverse-proxy/>

5.2 - Cloudflare, DNS et accès distant

Cloudflare peut fournir le DNS, un proxy public et des certificats entre le client et son réseau. La configuration doit rester cohérente avec Caddy et Immich.

DNS

Créer un enregistrement pour :

immich.dreamsaphir.net

Deux modes sont possibles :

Mode	Effet
DNS only	le client rejoint directement l'adresse publiée
Proxied	le trafic passe par le proxy Cloudflare

Le mode proxied ajoute une couche de protection mais aussi des limites de taille, de délai et de comportement selon l'offre. Tester les vidéos les plus volumineuses. Si les envois échouent uniquement à travers Cloudflare, comparer avec un accès local et examiner les limites du compte.

TLS

Avec un certificat valide sur Caddy, utiliser un mode de chiffrement strict entre Cloudflare et l'origine. Éviter les modes qui acceptent un certificat invalide.

Ports et pare-feu

- exposer uniquement 80 et 443 vers Caddy si nécessaire ;

- ne pas publier directement 30041 sur Internet ;
- filtrer l'administration TrueNAS ;
- conserver WireGuard pour les accès d'administration ;
- limiter les règles NAT à la cible exacte.

En-têtes et adresse client

Le proxy Caddy doit transmettre l'adresse client et le protocole d'origine. Si plusieurs proxies se suivent, configurer les plages de confiance avec prudence afin d'éviter qu'un client forge `X-Forwarded-For`.

Diagnostic

```
dig +short immich.dreamsaphir.net
curl -vI https://immich.dreamsaphir.net
curl -I http://ADRESSE_TRUENAS:30041
```

Comparer :

1. accès local direct ;
2. accès local par le domaine ;
3. accès extérieur par le domaine ;
4. upload d'une petite photo ;
5. upload d'une grande vidéo.

Erreurs courantes

Symptôme	Piste
525	certificat ou TLS entre Cloudflare et Caddy
403	règle Cloudflare, authentification ou pare-feu
upload interrompu	taille ou délai du proxy
boucle OAuth	domaine ou URI de redirection incorrecte
mauvaise adresse client	chaîne d'en-têtes proxy incorrecte

La protection d'accès Cloudflare placée devant tout Immich peut interférer avec l'application mobile et OAuth. Tester avant de l'imposer.

Référence Immich : <https://docs.immich.app/administration/reverse-proxy/>

5.3 - Authentification OIDC avec Authentik

Immich prend en charge OpenID Connect. Ton fournisseur utilise le domaine `auth.dreamsaphir.net`.

Créer l'application Authentik

Dans Authentik :

1. ouvrir `Applications > Applications` ;
2. créer une application Immich ;
3. choisir un fournisseur `OAuth2/OpenID Connect` ;
4. utiliser un client confidentiel et le flux Authorization Code ;
5. noter le Client ID et le Client Secret ;
6. choisir une clé de signature.

URI de redirection

Ajouter exactement :

```
app.immich:///oauth-callback  
https://immich.dreamsaphir.net/auth/login  
https://immich.dreamsaphir.net/user-settings
```

La première URI est indispensable pour Android et iOS.

Pour Authentik 2026.5 et versions plus récentes, définir les URI comme `Strict` et `Authorization` selon le guide d'intégration. Les versions antérieures gèrent différemment le type d'URI.

Configurer Immich

Ouvrir :

Administration > Settings > OAuth Authentication

Valeurs principales :

```
Enabled      : true
Issuer URL   : https://auth.dreamsaphir.net/application/o/immich/
Client ID    : valeur Authentik
Client Secret : valeur Authentik
Scope        : openid email profile
Button Text  : Connexion avec Authentik
Auto Register : selon la politique choisie
```

Adapter le slug `immich` si l'application Authentik en utilise un autre.

Vérifier la découverte OIDC

```
curl -fsS \
  https://auth.dreamsaphir.net/application/o/immich/.well-known/openid-configuration \
  | python3 -m json.tool
```

Tester sans se verrouiller

1. garder une session administrateur locale ouverte ;
2. utiliser une fenêtre privée pour le test OIDC ;
3. tester le Web ;
4. tester l'application mobile ;
5. vérifier la création ou liaison du compte ;
6. vérifier le retour après déconnexion.

Ne pas activer immédiatement `Auto Launch` ni désactiver le mot de passe local avant un test complet.

Récupération

Dans le conteneur serveur :

```
immich-admin enable-password-login  
immich-admin disable-oauth-login  
immich-admin reset-admin-password
```

Docker Compose :

```
docker exec -it immich_server immich-admin enable-password-login
```

Références :

- <https://docs.immich.app/administration/oauth/>
- <https://integrations.goauthentik.io/media/immich/>

5.4 - Sécuriser les comptes, sessions et API

Comptes

- utiliser un mot de passe unique pour le compte local de secours ;
- limiter le rôle administrateur ;
- désactiver ou supprimer les comptes inutiles après sauvegarde ;
- vérifier les sessions après une réinitialisation ;
- protéger Authentik par MFA ;
- conserver une procédure de récupération hors d'Immich.

Sessions

Après suspicion de compromission :

1. réinitialiser le mot de passe ;
2. invalider les sessions si l'option est proposée ;
3. révoquer les clés API ;
4. examiner les journaux du reverse proxy et d'Immich ;
5. vérifier les comptes administrateurs ;
6. changer les secrets OAuth si nécessaire.

Clés API

Les clés se créent dans les paramètres utilisateur. Limiter leurs permissions aux opérations nécessaires.

Ne jamais placer une clé directement dans :

- un script partagé ;
- BookStack ;

- l'historique du shell ;
- une URL ;
- un dépôt Git ;
- une capture d'écran.

Utiliser un fichier d'environnement protégé :

```
chmod 600 ~/.config/immich/cli.env
```

CLI Immich en conteneur

```
docker run --rm -it \  
  --env-file ~/.config/immich/cli.env \  
  -v "$PWD":/import:ro \  
  ghcr.io/immich-app/immich-cli:latest server-info
```

Exemple de fichier :

```
IMMICH_INSTANCE_URL=https://immich.dreamsaphir.net/api  
IMMICH_API_KEY=REPLACER
```

Exposition réseau

- publier uniquement le reverse proxy ;
- garder PostgreSQL et Redis non exposés ;
- restreindre le port TrueNAS 30041 au LAN ;
- utiliser WireGuard pour l'administration ;
- mettre à jour Immich, TrueNAS, Caddy et Authentik ;
- sauvegarder avant une mise à jour majeure.

Données sensibles

Les photos contiennent potentiellement visages, lieux et dates. Le partage partenaire peut exposer des métadonnées GPS. Vérifier les droits des albums, liens publics et partenaires.

Contrôle périodique

- liste des utilisateurs ;
- administrateurs ;
- clés API ;
- partages publics ;
- sauvegardes et tests de restauration ;
- certificats ;
- journaux d'authentification ;
- versions et correctifs.

Références :

- <https://docs.immich.app/administration/user-management/>
- <https://docs.immich.app/features/command-line-interface/>

6 - Performances et accélération matérielle

Exploiter le GPU, régler les tâches et surveiller les performances d'Immich.

6.1 - Transcodage NVIDIA avec une Quadro P2000

La NVIDIA Quadro P2000 peut accélérer le transcodage vidéo avec NVENC et réduire la charge CPU. Immich indique que le transcodage matériel peut produire des fichiers plus volumineux et parfois de qualité inférieure à réglage comparable.

Vérifier le GPU sur TrueNAS

```
sudo lspci -nnk | grep -A3 -i nvidia  
nvidia-smi
```

Ton environnement a déjà utilisé un pilote NVIDIA 570.172.08 avec la P2000 reconnue. Après chaque mise à jour TrueNAS, refaire le contrôle.

Vérifier dans le conteneur

Découvrir le conteneur :

```
sudo docker ps --format '{{.Names}} {{.Image}}' | grep -i immich
```

Puis :

```
sudo docker exec NOM_CONTENEUR_IMMICH nvidia-smi
```

L'image serveur observée utilisait `NVIDIA_DRIVER_CAPABILITIES=all`. Vérifier cette variable et l'allocation GPU dans l'interface TrueNAS plutôt que de modifier les fichiers Compose générés.

Activer dans TrueNAS

1. éditer l'application Immich ;
2. activer la configuration GPU ;
3. sélectionner la P2000 pour le serveur Immich ;
4. redéployer ;
5. ouvrir les paramètres de transcodage vidéo ;
6. choisir NVENC ;
7. commencer avec le décodage matériel désactivé ;
8. tester un fichier compatible puis activer le décodage si utile.

Contrôler l'utilisation

```
watch -n 1 nvidia-smi
```

Dans une autre session, lancer une tâche de transcodage et suivre les journaux :

```
sudo docker logs -f NOM_CONTENEUR_IMMICH
```

Limites de la P2000

- génération Pascal ancienne mais encore adaptée à H.264 et HEVC ;
- pas d'encodage AV1 ;
- qualité et formats limités par le matériel ;
- 5 Go de VRAM à partager si le machine learning utilise aussi le GPU ;
- nombre de sessions et performances variables selon les fichiers.

En cas d'artefacts

1. tester le même fichier en transcodage logiciel ;
2. désactiver le décodage matériel ;
3. comparer H.264 et HEVC ;
4. vérifier les journaux FFmpeg ;
5. contrôler pilote et température ;
6. ne pas conclure à une panne GPU à partir d'un seul fichier corrompu.

Référence : <https://docs.immich.app/features/hardware-transcoding/>

6.2 - Machine learning avec CUDA

Le machine learning Immich fournit la recherche intelligente et la reconnaissance faciale. CUDA est pris en charge sur les GPU NVIDIA avec une capacité de calcul 5.2 ou supérieure.

Conditions actuelles

- GPU NVIDIA compatible ;
- pilote officiel NVIDIA 545 ou plus récent ;
- accès au GPU depuis le conteneur ;
- image machine learning CUDA ;
- mémoire GPU suffisante pour le modèle.

La Quadro P2000 répond à la condition de capacité de calcul et ton pilote 570.172.08 dépasse le minimum documenté.

Docker Compose

Le service doit utiliser une image se terminant par `-cuda` et réserver un GPU. Exemple conceptuel :

```
immich-machine-learning:
  image: ghcr.io/immich-app/immich-machine-learning:${IMMICH_VERSION:-release}-cuda
  deploy:
    resources:
      reservations:
        devices:
          - driver: nvidia
            count: 1
            capabilities:
              - gpu
```

Utiliser de préférence les fichiers officiels `hwaccel.ml.yml` correspondant à la version Immich.

TrueNAS

Dans les paramètres de l'application :

1. activer le GPU pour le machine learning ;
2. sélectionner la P2000 ;
3. redéployer ;
4. vérifier le conteneur ;
5. lancer une tâche de visages ou recherche intelligente.

Vérifier

```
sudo docker exec NOM_CONTENEUR_ML nvidia-smi  
sudo docker logs --tail 200 NOM_CONTENEUR_ML
```

Chercher :

```
CUDAExecutionProvider  
Available ORT providers
```

Surveiller :

```
watch -n 1 nvidia-smi
```

Gestion de la VRAM

La P2000 possède 5 Go de VRAM. Éviter de lancer simultanément un import massif, plusieurs transcodages et de nombreux traitements ML sans surveillance.

Réduire si nécessaire :

- le nombre de workers ;
- la concurrence des tâches ;
- la taille ou complexité du modèle ;
- les transcodages parallèles.

Après activation

Il n'est pas nécessaire de relancer tous les anciens jobs uniquement pour profiter de l'accélération. Les prochaines tâches utiliseront le backend disponible.

Référence : <https://docs.immich.app/features/ml-hardware-acceleration/>

6.3 - Concurrence, tâches et choix du stockage

Priorités de stockage

1. PostgreSQL sur SSD si possible.
2. Miniatures sur SSD si la navigation est lente.
3. Originaux sur le pool de capacité.
4. Sauvegardes sur un autre support et un autre système.

Séparer les datasets peut améliorer l'organisation, mais complexifie la restauration. Mesurer avant de multiplier les montages.

Concurrence des tâches

Dans `Administration > Settings`, ajuster progressivement :

- extraction de métadonnées ;
- miniatures ;
- transcodage ;
- recherche intelligente ;
- détection faciale.

Ne pas dépasser sans raison le nombre de coeurs disponibles, surtout pour les miniatures. Une concurrence trop forte peut ralentir l'API et saturer le stockage.

Import massif

Avant :

```
df -hT
free -h
```

```
nvidia-smi
```

Pendant :

```
docker stats  
iostat -xz 2  
watch -n 2 nvidia-smi
```

`iostat` est fourni par `sysstat` sur de nombreuses distributions.

Machine learning distant

Immich peut envoyer les aperçus à un service machine learning plus puissant. Cette option convient à un serveur limité, mais ajoute une dépendance réseau et un second service à sauvegarder ou reconstruire.

Base PostgreSQL

Éviter les modifications manuelles de paramètres PostgreSQL copiées depuis un autre environnement. L'image officielle Immich contient des réglages et extensions adaptés.

Contrôles utiles :

```
docker exec -it immich_postgres psql -U postgres -d immich -c 'select version();'  
docker exec -it immich_postgres pg_isready -U postgres -d immich
```

Adapter utilisateur, base et conteneur.

Nettoyage

Après une mise à jour Docker validée :

```
docker image prune
```

Cette commande supprime les images inutilisées, pas les images en cours d'utilisation. Ne pas supprimer les volumes ou dossiers de données pour gagner de la place.

Mesurer

Comparer avant et après :

- temps d'ouverture des miniatures ;
- durée d'une tâche de 1000 actifs ;
- charge CPU ;
- mémoire disponible ;
- latence disque ;
- utilisation GPU ;
- réactivité de l'API.

Références :

- <https://docs.immich.app/administration/jobs-workers/>
- <https://docs.immich.app/install/truenas/>

6.4 - Surveillance, journaux et contrôles

État des conteneurs

Docker Compose :

```
docker compose ps
docker compose logs --tail=100
docker stats --no-stream
```

TrueNAS :

```
sudo docker ps --format 'table {{.Names}}\t{{.Status}}'
sudo docker stats --no-stream
```

Version et schéma

```
docker exec -it immich_server immich-admin version
docker exec -it immich_server immich-admin schema-check
```

Adapter le nom du conteneur.

Santé de PostgreSQL

```
docker exec immich_postgres pg_isready -U postgres -d immich
docker logs --tail 100 immich_postgres
```

Ressources hôte

```
uptime  
free -h  
df -hT  
df -i  
lsblk -f
```

TrueNAS :

```
sudo zpool status  
sudo zpool list  
sudo zfs list
```

GPU :

```
nvidia-smi
```

Interface Immich

- Administration > Server Stats pour les actifs et quotas ;
- Administration > Jobs pour les files ;
- Administration > Maintenance pour l'intégrité ;
- Administration > Settings > Backup pour les dumps ;
- Administration > External Libraries pour les scans.

Collecte de diagnostic

Le pack contient `scripts/diagnostic_immich.sh`. Exemple :

```
IMMICH_COMPOSE_DIR=/srv/immich/app \  
./scripts/diagnostic_immich.sh
```

Le script n'effectue aucune modification. Examiner le rapport avant de le partager, car il peut contenir noms de conteneurs, chemins, versions et adresses.

Alertes utiles

- espace libre faible ;
- pool ZFS dégradé ;
- conteneur en redémarrage ;
- sauvegarde PostgreSQL absente ou ancienne ;
- files d'attente bloquées ;
- erreurs d'intégrité ;
- certificat proche de l'expiration ;
- GPU absent après mise à jour.

Ne pas attendre un incident pour tester les commandes de diagnostic et la restauration.

7 - Mises à jour et maintenance

Mettre Immich à jour, gérer les changements de version et utiliser les commandes d'administration.

7.1 - Préparer une mise à jour

Une mise à jour Immich peut inclure des changements de schéma PostgreSQL, d'images de conteneurs, de variables ou de montages. Un downgrade n'est pas pris en charge.

Avant toute mise à jour

1. lire les notes de version depuis la version actuelle jusqu'à la cible ;
2. relever la version du serveur et des clients mobiles ;
3. vérifier l'état des conteneurs et de PostgreSQL ;
4. déclencher un dump de base ;
5. copier ou snapshotter les médias et la configuration ;
6. vérifier l'espace libre ;
7. attendre la fin des imports et tâches critiques ;
8. prévoir une fenêtre de maintenance.

Inventaire

```
docker exec immich_server immich-admin version
docker compose images
docker compose ps
docker compose config --services
grep -E '^(IMMICH_VERSION|UPLOAD_LOCATION|DB_DATA_LOCATION)=' .env
```

TrueNAS :

```
sudo docker ps --format '{{.Names}} {{.Image}} {{.Status}}' | grep -i immich
sudo zfs list | grep -i immich
sudo zpool status
```

Sauvegarde de base

```
mkdir -p /srv/backups/immich
docker exec -t immich_postgres \
    pg_dump --clean --if-exists --dbname=immich --username=postgres \
    | gzip > /srv/backups/immich/pre-update.sql.gz
```

Adapter le conteneur, la base et l'utilisateur. Vérifier le fichier :

```
gzip -t /srv/backups/immich/pre-update.sql.gz
ls -lh /srv/backups/immich/pre-update.sql.gz
```

Snapshot

Sur TrueNAS, créer des snapshots manuels des datasets Immich après le dump. Le dump présent dans `backups` doit être inclus dans le snapshot ou copié avec les médias.

Compatibilité mobile

La documentation Immich indique que l'application mobile est généralement compatible avec la version majeure actuelle et précédente, tandis que le serveur attend la même version majeure. Mettre à jour les mobiles avant le serveur lors d'un passage majeur.

Critères de retour

Avant de commencer, décider ce qui impose un arrêt :

- PostgreSQL ne démarre pas ;
- migration de schéma en erreur ;
- médias absents ;
- authentification impossible ;
- erreurs d'intégrité nouvelles ;
- reverse proxy ou mobile inutilisable.

Le retour consiste à restaurer une sauvegarde et des fichiers cohérents, pas à simplement remettre une ancienne image sur une base déjà migrée.

Références :

- <https://docs.immich.app/install/upgrading/>
- <https://docs.immich.app/administration/backup-and-restore/>

7.2 - Mettre à jour Docker Compose

Mise à jour standard

Depuis le dossier Compose :

```
cd /srv/immich/app  
docker compose pull  
docker compose up -d
```

Suivre :

```
docker compose ps  
docker compose logs -f immich-server
```

Les migrations de base peuvent prendre du temps. Ne pas interrompre un traitement simplement parce que le journal reste plusieurs minutes sur une réindexation.

Mettre à jour les fichiers Compose

Lors d'une version majeure, comparer le fichier local au fichier officiel :

```
wget -O /tmp/immich-compose-nouveau.yml \  
https://github.com/immich-app/immich/releases/latest/download/docker-compose.yml  
  
diff -u docker-compose.yml /tmp/immich-compose-nouveau.yml
```

Ne pas remplacer aveuglément un fichier contenant des montages GPU, bibliothèques externes ou réseaux personnalisés. Reporter les adaptations sur la nouvelle base officielle.

Télécharger aussi l'exemple d'environnement :

```
wget -O /tmp/immich-env-nouveau \  
https://github.com/immich-app/immich/releases/latest/download/example.env  
  
diff -u .env.example /tmp/immich-env-nouveau
```

Ne pas comparer ou partager un `.env` réel contenant des secrets.

Passage de v2.7.5 vers v3

Procédure recommandée :

1. sauvegarder complètement v2.7.5 ;
2. restaurer ou valider la copie sur v2.7.5 si une migration d'hôte est en cours ;
3. mettre les applications mobiles à jour ;
4. lire les notes de rupture v3 ;
5. adapter Compose et les variables ;
6. passer `IMMICH_VERSION` à `v3` ;
7. lancer `docker compose pull && docker compose up -d` ;
8. vérifier schéma, comptes, albums, OAuth et médias.

Vérifications après mise à jour

```
docker exec immich_server immich-admin version  
docker exec immich_server immich-admin schema-check  
docker compose ps  
docker compose logs --tail=200
```

Puis tester interface, mobile, upload, lecture vidéo, recherche, visages, bibliothèque externe et dump PostgreSQL.

Nettoyage différé

Après plusieurs jours de validation :

```
docker image prune
```

Conserver les sauvegardes pré-mise à jour selon la politique de rétention.

Référence : <https://docs.immich.app/install/upgrading/>

7.3 - Mettre à jour TrueNAS et migrer vers VectorChord

L'application TrueNAS gère ses images et fichiers Compose. Les changements doivent passer par le catalogue et l'écran de mise à jour.

Avant la mise à jour TrueNAS ou Immich

- dump PostgreSQL validé ;
- snapshots des datasets `data` et `pgData` ;
- réplication terminée ;
- version actuelle notée ;
- état du pool sain ;
- P2000 visible par `nvidia-smi` ;
- fenêtre de maintenance prévue.

Mettre l'application à jour

1. ouvrir `Apps > Installed Applications` ;
2. sélectionner Immich ;
3. lire les notes de mise à jour ;
4. lancer la mise à jour ;
5. attendre l'état `Running` ;
6. examiner tous les conteneurs ;
7. contrôler la base et l'interface.

```
sudo docker ps -a --format '{{.Names}} {{.Image}} {{.Status}}' | grep -i immich  
sudo docker logs --tail 200 NOM_CONTENEUR_SERVEUR  
sudo docker logs --tail 200 NOM_CONTENEUR_POSTGRES
```

pgvecto.rs et VectorChord

Les anciennes installations peuvent comporter :

```
pgvecto
pgvecto_upgrade
```

Ton installation a déjà rencontré un `pgvecto_upgrade` quittant avec le code 1. Ne pas relancer au hasard ni supprimer `pgData`.

Procédure de diagnostic :

```
sudo docker ps -a --format '{{.Names}} {{.Image}} {{.Status}}' | grep -Ei
'immich|pgvecto|postgres'
sudo docker logs --tail 300 NOM_CONTENEUR_PGVECTO_UPGRADE
sudo docker logs --tail 300 NOM_CONTENEUR_POSTGRES
```

Relever :

- version PostgreSQL ;
- image pgvecto.rs source ;
- version Immich ;
- première erreur réelle dans le journal ;
- espace libre et permissions du dataset.

La documentation Immich actuelle utilise VectorChord. L'image PostgreSQL officielle récente inclut les extensions nécessaires pour restaurer d'anciens dumps pgvecto.rs. Pour une application TrueNAS, suivre le chemin de migration fourni par le mainteneur du catalogue au lieu de modifier le Compose rendu.

Après mise à jour TrueNAS

```
nvidia-smi
sudo docker exec NOM_CONTENEUR_IMMICH nvidia-smi
sudo zpool status
```

Vérifier aussi le port 30041, les montages externes et les ACL.

Références :

- <https://docs.immich.app/install/upgrading/>

- <https://docs.immich.app/install/truenas/>

7.4 - Commandes d'administration et récupération

L'image `immich-server` contient la commande `immich-admin`.

Lancer la commande

Docker Compose :

```
docker exec -it immich_server immich-admin help
```

TrueNAS :

```
sudo docker exec -it NOM_CONTENEUR_IMMICH immich-admin help
```

Commandes utiles

```
immich-admin version
immich-admin schema-check
immich-admin list-users
immich-admin reset-admin-password
immich-admin enable-password-login
immich-admin disable-password-login
immich-admin enable-oauth-login
immich-admin disable-oauth-login
immich-admin enable-maintenance-mode
immich-admin disable-maintenance-mode
```

```
immich-admin grant-admin  
immich-admin revoke-admin
```

Récupération d'un accès administrateur

```
docker exec -it immich_server immich-admin reset-admin-password
```

Choisir d'invalider les sessions existantes si le mot de passe a pu être compromis.

Si OIDC bloque la connexion :

```
docker exec -it immich_server immich-admin enable-password-login  
docker exec -it immich_server immich-admin disable-oauth-login
```

Mode maintenance

```
docker exec -it immich_server immich-admin enable-maintenance-mode
```

La commande affiche une URL temporaire d'accès maintenance. La protéger comme un secret.

Changement de chemin média

Les versions récentes proposent :

```
immich-admin change-media-location
```

Cette commande modifie les chemins stockés en base pour un nouvel emplacement de médias. Elle ne copie aucun fichier. Avant de l'utiliser :

1. sauvegarder la base ;
2. copier et vérifier les médias ;
3. comprendre ancien et nouveau chemin vus par le conteneur ;
4. exécuter en maintenance ;
5. contrôler l'intégrité.

Pour les bibliothèques externes, conserver de préférence le même chemin interne lors d'une migration.

Schéma

```
docker exec immich_server immich-admin schema-check
```

Une dérive de schéma doit être analysée avec la version et les journaux. Ne pas modifier manuellement les tables pour faire disparaître le message.

Référence : <https://docs.immich.app/administration/server-commands/>

8 - Stratégie de sauvegarde

Protéger la base, les médias, la configuration et les copies distantes avec des tests de restauration.

8.1 - Périmètre et stratégie

3-2-1

Immich recommande une stratégie 3-2-1 :

- 3 copies des données ;
- 2 types de supports ou systèmes ;
- 1 copie hors du serveur principal.

Éléments à protéger

Élément	Pourquoi
PostgreSQL	utilisateurs, albums, chemins, visages, partages et paramètres
<code>upload</code>	originaux internes
<code>library</code>	originaux organisés si le modèle est actif
<code>profile</code>	profils utilisateurs
<code>backups</code>	dumps automatiques de la base
<code>thumbs</code>	régénérable mais coûteux en temps
<code>encoded-video</code>	régénérable mais coûteux en ressources
bibliothèques externes	originaux hors du stockage interne
configuration	Compose, <code>.env</code> , paramètres de proxy et montages

Exemple adapté

1. copie de production sur TrueNAS ;
2. snapshots ZFS locaux ;
3. réplication ou rsync vers DreamProxmox ;
4. copie supplémentaire déconnectée ou hors site.

Un snapshot sur le même pool protège contre certaines suppressions mais pas contre la panne du pool, le vol ou la destruction du serveur.

Cohérence

Le dump PostgreSQL et les médias doivent correspondre à une période proche. Pour un point de restauration parfaitement cohérent :

1. mettre Immich en maintenance ou arrêter les écritures ;
2. créer le dump ;
3. copier ou snapshotter les médias ;
4. reprendre le service.

Les dumps automatiques quotidiens offrent une bonne protection, mais il faut aussi sauvegarder le dossier qui les contient.

Objectifs de reprise

Définir :

- RPO - quantité maximale de données récentes pouvant être perdue ;
- RTO - durée maximale de remise en service ;
- rétention quotidienne, hebdomadaire et mensuelle ;
- responsable du contrôle ;
- emplacement des secrets de restauration.

Vérification

Chaque sauvegarde doit être contrôlée :

```
gzip -t dump.sql.gz  
sha256sum dump.sql.gz  
find /chemin/sauvegarde -type f | wc -l  
du -sh /chemin/sauvegarde
```

Le test réel consiste à restaurer dans un environnement isolé, ouvrir des photos, vérifier albums, comptes, recherche et bibliothèques externes.

Référence : <https://docs.immich.app/administration/backup-and-restore/>

8.2 - Sauvegarder PostgreSQL

Immich crée automatiquement des dumps PostgreSQL dans :

```
UPLOAD_LOCATION/backups
```

La rétention et l'horaire se règlent dans :

```
Administration > Settings > Backup
```

Le réglage par défaut documenté est un dump quotidien à 02:00 avec conservation des 14 derniers dumps.

Déclencher un dump depuis Immich

```
Administration > Job Queues > Create job > Create Database Dump
```

Dump manuel Docker

Identifier le conteneur :

```
docker ps --format '{{.Names}} {{.Image}}' | grep -Ei 'immich.*postgres|postgres.*immich'
```

Créer un dump :

```
mkdir -p /srv/backups/immich

docker exec -t immich_postgres \
```

```
pg_dump --clean --if-exists --dbname=immich --username=postgres \  
| gzip > /srv/backups/immich/immich-$(date +%F-%H%M).sql.gz
```

Vérifier :

```
gzip -t /srv/backups/immich/immich-AAAA-MM-JJ-HHMM.sql.gz  
ls -lh /srv/backups/immich
```

TrueNAS

Découvrir le conteneur exact :

```
sudo docker ps --format '{{.Names}} {{.Image}}' | grep -Ei 'immich|postgres|vector'
```

Puis adapter :

```
sudo docker exec -t NOM_CONTENEUR_POSTGRES \  
pg_dump --clean --if-exists --dbname=immich --username=postgres \  
| gzip > /mnt/HDD_DATA_TRUENAS/IMMICH/data/backups/manuel-$(date +%F-%H%M).sql.gz
```

Ce que le dump ne contient pas

- photos ;
- vidéos ;
- miniatures ;
- fichiers externes ;
- configuration `.env` ;
- paramètres du reverse proxy.

Règles

- ne pas utiliser une copie brute du dossier PostgreSQL comme seule sauvegarde ;
- protéger le dump avec les mêmes exigences de confidentialité que les photos ;
- copier le dump hors du serveur ;
- tester `gzip -t` ;
- tester une restauration ;
- utiliser une version compatible.

Le processus de restauration a changé avec Immich v2.5.0. Pour un ancien dump, consulter la documentation correspondant à la version de création.

Référence : <https://docs.immich.app/administration/backup-and-restore/>

8.3 - Snapshots ZFS et réplication TrueNAS

Les snapshots ZFS protègent rapidement l'état d'un dataset. Ils ne remplacent pas un dump PostgreSQL logique ni une copie hors du pool.

Snapshot manuel

Après un dump de base :

```
sudo zfs snapshot HDD_DATA_TRUENAS/IMMICH/data@manuel-$(date +%F-%H%M)
sudo zfs snapshot HDD_DATA_TRUENAS/IMMICH/pgData@manuel-$(date +%F-%H%M)
```

Adapter les noms ZFS réels obtenus avec :

```
sudo zfs list
```

Tâche périodique TrueNAS

Dans `Data Protection > Periodic Snapshot Tasks` :

1. choisir le dataset parent ou les datasets enfants ;
2. activer la récursivité si nécessaire ;
3. définir l'horaire ;
4. définir la durée de conservation ;
5. éviter une rétention qui remplit le pool ;
6. vérifier les snapshots créés.

Cohérence PostgreSQL

Un snapshot de `pgData` pendant que PostgreSQL écrit est généralement comparable à un arrêt brutal. PostgreSQL sait souvent récupérer, mais ce snapshot ne doit pas remplacer un dump logique.

Pour un snapshot applicatif cohérent :

1. activer la maintenance ;
2. créer un dump ;
3. arrêter Immich si un point strict est requis ;
4. prendre les snapshots ;
5. redémarrer ;
6. vérifier les services.

Réplication

Dans `Data Protection > Replication Tasks` :

- source : datasets Immich ;
- destination : autre pool, autre TrueNAS ou serveur compatible ;
- transport : local ou SSH ;
- snapshots : tâche périodique correspondante ;
- rétention : cohérente avec la source et la capacité.

Vérifier

```
sudo zfs list -t snapshot | grep -i immich
sudo zpool status
sudo zpool list
```

Restauration d'un fichier

Préférer cloner ou parcourir un snapshot puis copier le fichier voulu. Éviter un rollback complet sans analyse, car il supprime les modifications plus récentes du dataset.

Après réplication

Vérifier que la destination contient :

- les médias ;
- les dumps ;
- les profils ;
- les datasets séparés ;
- les bibliothèques externes si elles font partie du plan.

8.4 - Copie distante vers Proxmox et validation

Ton environnement utilise déjà une liaison SSH de TrueNAS vers Proxmox. Elle peut transporter une copie supplémentaire des données Immich.

Préparer la destination

Sur Proxmox :

```
sudo mkdir -p /srv/backups/immich  
sudo chown UTILISATEUR_BACKUP:UTILISATEUR_BACKUP /srv/backups/immich
```

Restreindre la clé SSH au compte et à l'usage de sauvegarde si possible.

Simulation rsync

Depuis TrueNAS :

```
rsync -aHn --numeric-ids --info=progress2 \  
/mnt/HDD_DATA_TRUENAS/IMMICH/data/ \  
UTILISATEUR_BACKUP@ADRESSE_PROXMOX:/srv/backups/immich/data/
```

Après vérification :

```
rsync -aH --numeric-ids --info=progress2 \  
/mnt/HDD_DATA_TRUENAS/IMMICH/data/ \  
UTILISATEUR_BACKUP@ADRESSE_PROXMOX:/srv/backups/immich/data/
```

Ne pas ajouter `--delete` tant qu'une politique de miroir et de rétention séparée n'est pas en place.

Base PostgreSQL

Créer le dump avant rsync afin que le dossier `backups` contienne une version récente et cohérente.

Borg

Immich fournit un modèle Borg qui sauvegarde la base et les médias avec déduplication et rétention. Borg peut réduire l'espace par rapport à des copies complètes répétées. Il demande cependant une configuration, une clé et des tests de restauration.

Contrôles de destination

```
ssh UTILISATEUR_BACKUP@ADRESSE_PROXMOX \  
'du -sh /srv/backups/immich/data; find /srv/backups/immich/data -type f | wc -l'
```

Comparer quelques sommes :

```
sha256sum /mnt/HDD_DATA_TRUENAS/IMMICH/data/backups/FICHER.sql.gz  
ssh UTILISATEUR_BACKUP@ADRESSE_PROXMOX \  
'sha256sum /srv/backups/immich/data/backups/FICHER.sql.gz'
```

Script fourni

Le pack contient :

```
./scripts/sauvegarde_immich_docker.sh
```

Il crée un dump logique puis copie les médias sans suppression vers un dossier de sauvegarde. Lire et configurer ses variables avant exécution.

Test trimestriel conseillé

1. choisir une sauvegarde ;
2. la restaurer sur un réseau isolé ;

3. vérifier plusieurs originaux ;
4. ouvrir des albums ;
5. tester un compte ;
6. vérifier les bibliothèques externes ;
7. documenter la durée et les problèmes.

Référence Borg : <https://docs.immich.app/guides/template-backup-script/>

9 - Migration complète d'Immich

Inventorier la source, transférer les données et migrer proprement vers une nouvelle installation.

9.1 - Inventaire et plan de migration

Une migration réussie commence par un inventaire. Ne pas arrêter ou supprimer la source tant que la destination et la restauration ne sont pas validées.

Informations à relever

Élément	Exemple
Version Immich source	v2.7.5
Type de déploiement	VM Proxmox avec Docker Compose
Version PostgreSQL	valeur du conteneur
Extension vectorielle	pgvecto.rs ou VectorChord
Emplacement des médias	UPLOAD_LOCATION
Emplacement PostgreSQL	DB_DATA_LOCATION
Bibliothèques externes	chemins hôte et conteneur
Domaine	immich.dreamsaphir.net
OIDC	auth.dreamsaphir.net
Destination	TrueNAS et dataset Immich

Commandes source

```
cd /chemin/compose/immich
docker compose ps
docker compose images
docker exec immich_server immich-admin version
docker exec immich_server immich-admin schema-check
grep -E '^(UPLOAD_LOCATION|DB_DATA_LOCATION|IMMICH_VERSION|DB_USERNAME|DB_DATABASE_NAME)='
```

```
.env
docker inspect immich_server --format '{{range .Mounts}}{{println .Source "->"
.Destination}}{{end}}'
```

Ne pas afficher `DB_PASSWORD` dans le rapport.

Taille et fichiers

```
df -hT
du -sh CHEMIN_MEDIA
find CHEMIN_MEDIA -type f | wc -l
```

Pour les bibliothèques externes :

```
du -sh CHEMIN_BIBLIOTHEQUE_EXTERNE
find CHEMIN_BIBLIOTHEQUE_EXTERNE -type f | wc -l
```

Version de PostgreSQL et extensions

```
docker exec immich_postgres psql -U postgres -d immich -c 'select version();'
docker exec immich_postgres psql -U postgres -d immich -c '\dx'
```

Adapter conteneur, utilisateur et base.

Choisir la stratégie de version

Stratégie conservatrice :

1. restaurer avec la même version Immich que la source ;
2. vérifier le contenu ;
3. sauvegarder à nouveau ;
4. effectuer la montée de version.

Stratégie directe : restaurer le dump dans la version cible et laisser Immich appliquer les migrations. Elle est plus rapide mais ajoute des changements simultanés.

Pour une source v2.7.5 et une cible v3.1.0, la stratégie conservatrice facilite le diagnostic.

Critères de succès

- nombre d'actifs cohérent ;
- albums et partages présents ;
- utilisateurs accessibles ;
- originaux ouvrables ;
- bibliothèques externes scannables ;
- OAuth Web et mobile fonctionnel ;
- jobs sans erreur critique ;
- dump de la nouvelle instance validé.

9.2 - Préparer et exporter le serveur source

Première copie sans interruption

Créer une destination de préparation et copier les médias pendant que la source fonctionne :

```
rsync -aHn --numeric-ids --info=progress2 \  
  CHEMIN_MEDIA_SOURCE/ \  
  UTILISATEUR@DESTINATION:/chemin/immich-data/
```

Après vérification :

```
rsync -aH --numeric-ids --info=progress2 \  
  CHEMIN_MEDIA_SOURCE/ \  
  UTILISATEUR@DESTINATION:/chemin/immich-data/
```

Ne pas utiliser `--delete` pour cette première migration.

Créer un dump de test

```
mkdir -p /srv/export-immich  
  
docker exec -t immich_postgres \  
  pg_dump --clean --if-exists --dbname=immich --username=postgres \  
  | gzip > /srv/export-immich/immich-test.sql.gz  
  
gzip -t /srv/export-immich/immich-test.sql.gz
```

Fenêtre de bascule

1. annoncer l'interruption ;
2. terminer les uploads mobiles ;
3. attendre les tâches indispensables ;
4. activer la maintenance ou arrêter le serveur ;
5. créer le dump final ;
6. faire un rsync final ;
7. relever les sommes et tailles ;
8. garder la source intacte.

Dump final

```
docker exec -it immich_server immich-admin enable-maintenance-mode

docker exec -t immich_postgres \
  pg_dump --clean --if-exists --dbname=immich --username=postgres \
  | gzip > /srv/export-immich/immich-final.sql.gz

gzip -t /srv/export-immich/immich-final.sql.gz
sha256sum /srv/export-immich/immich-final.sql.gz \
  > /srv/export-immich/immich-final.sql.gz.sha256
```

Si la maintenance n'est pas disponible sur la version source, arrêter le service serveur tout en gardant PostgreSQL disponible, ou planifier une courte interruption complète après le dump.

Copie finale des médias

```
rsync -aH --numeric-ids --info=progress2 \
  CHEMIN_MEDIA_SOURCE/ \
  UTILISATEUR@DESTINATION:/chemin/immich-data/
```

Copier aussi le dump, les fichiers Compose et une version expurgée de l'inventaire. Conserver `.env` dans un canal sécurisé.

Bibliothèques externes

Deux choix :

- conserver les fichiers au même emplacement réseau et monter le même chemin interne ;
- copier la bibliothèque vers TrueNAS et conserver le même chemin vu par le conteneur.

Le second choix nécessite une vérification des nombres de fichiers et de quelques sommes.

9.3 - Transférer et préparer la destination

Préparer TrueNAS

Exemple :

```
/mnt/HDD_DATA_TRUENAS/IMMICH/data  
/mnt/HDD_DATA_TRUENAS/IMMICH/pgData
```

Créer les datasets et permissions avant de copier. Ne pas démarrer Immich avec des montages incomplets.

Copier vers le dataset

Depuis la source :

```
rsync -aHn --numeric-ids --info=progress2 \  
CHEMIN_MEDIA_SOURCE/ \  
truenas_admin@ADRESSE_TRUENAS:/mnt/HDD_DATA_TRUENAS/IMMICH/data/
```

Après vérification :

```
rsync -aH --numeric-ids --info=progress2 \  
CHEMIN_MEDIA_SOURCE/ \  
truenas_admin@ADRESSE_TRUENAS:/mnt/HDD_DATA_TRUENAS/IMMICH/data/
```

La connexion SSH peut ne pas autoriser l'écriture directe. Dans ce cas, copier vers un dossier de transit contrôlé puis déplacer depuis TrueNAS avec les droits appropriés.

Placer le dump

```
sudo mkdir -p /mnt/HDD_DATA_TRUENAS/IMMICH/data/backups
sudo cp immich-final.sql.gz \
/mnt/HDD_DATA_TRUENAS/IMMICH/data/backups/
```

Corriger les permissions via l'interface TrueNAS ou avec des commandes ciblées après identification des UID.

Configurer les montages

- données internes vers `/data` ;
- chaque dataset séparé vers `/data/NOM` ;
- bibliothèque externe vers `/mnt/external-libraries/NOM` ;
- mêmes chemins internes que la source si possible.

Version cible

Si l'application TrueNAS ne propose que la version majeure récente :

- utiliser la restauration onboarding avec le dump compatible ;
- ou valider d'abord le dump dans une instance Docker Compose temporaire à la version source ;
- conserver la source et les sauvegardes jusqu'à réussite.

Avant démarrage

```
sudo find /mnt/HDD_DATA_TRUENAS/IMMICH/data -maxdepth 1 -mindepth 1 -printf '%f\n'
sudo du -sh /mnt/HDD_DATA_TRUENAS/IMMICH/data
sudo ls -lh /mnt/HDD_DATA_TRUENAS/IMMICH/data/backups
```

Attendre les dossiers :

```
backups
encoded-video
library
```

profile

thumbs

upload

Selon la configuration source, `library` peut être vide ou non utilisé.

Bascule DNS

Tester d'abord en local. Modifier ensuite Caddy vers `ADRESSE_TRUENAS:30041`, recharger et tester OAuth et mobile. Ne pas changer simultanément domaine, chemins internes et identité OIDC si cela peut être évité.

9.4 - Cas pratique Proxmox vers TrueNAS

Ce cas reprend la migration de ton ancienne instance Proxmox vers TrueNAS.

Situation

- source : VM ou conteneur sur DreamProxmox ;
- destination : TrueNAS Community Edition sur le HPE ML30 Gen9 ;
- dataset : `/mnt/HDD_DATA_TRUENAS/IMMICH` ;
- GPU : Quadro P2000 5 Go ;
- domaine Immich : `immich.dreamsaphir.net` ;
- fournisseur OIDC : `auth.dreamsaphir.net` ;
- port TrueNAS observé : `30041` ;
- ancienne base : PostgreSQL avec pgvecto.rs ;
- méthode déjà utilisée : dump SQL et copie des actifs.

Procédure recommandée

1. relever la version exacte sur Proxmox ;
2. déclencher un dump PostgreSQL ;
3. copier les six dossiers Immich ;
4. copier les bibliothèques externes ou conserver leur montage ;
5. préparer `data` et `pgData` sur TrueNAS ;
6. configurer l'application et les chemins internes ;
7. restaurer le dump par l'onboarding ou une procédure contrôlée ;
8. vérifier les migrations pgvecto.rs vers VectorChord ;
9. tester localement sur le port 30041 ;
10. basculer Caddy ;
11. tester Authentik et les applications mobiles ;
12. activer la P2000 après validation sur CPU ;
13. déclencher un nouveau dump sur TrueNAS ;
14. mettre en place snapshots et copie vers Proxmox.

Contrôle des données

Comparer source et destination :

```
find CHEMIN_SOURCE -type f | wc -l  
du -sh CHEMIN_SOURCE  
  
sudo find /mnt/HDD_DATA_TRUENAS/IMMICH/data -type f | wc -l  
sudo du -sh /mnt/HDD_DATA_TRUENAS/IMMICH/data
```

Les nombres peuvent différer si des fichiers temporaires ou générés sont recréés. Comparer surtout les originaux dans `upload`, `library` et la bibliothèque externe.

Validation fonctionnelle

- compte administrateur ;
- utilisateurs ;
- chronologie ;
- dix photos anciennes et récentes ;
- plusieurs vidéos ;
- albums ;
- favoris ;
- visages et recherche ;
- bibliothèques externes ;
- application mobile ;
- OAuth ;
- sauvegarde de base ;
- accélération GPU.

Conserver la source

Garder la VM Proxmox arrêtée, non publiée et non modifiée pendant la période de validation. Ne pas lancer les deux instances en écriture derrière le même domaine.

Après validation, archiver la configuration source et conserver au moins un dump et une copie cohérente des médias selon la rétention choisie.

10 - Restauration et dépannage

Restaurer Immich par l'interface ou la ligne de commande et résoudre les incidents courants.

10.1 - Restauration par l'interface et l'onboarding

Les versions récentes d'Immich permettent de restaurer un dump depuis l'interface. Cette méthode est recommandée pour la plupart des utilisateurs.

Instance existante

1. ouvrir `Administration > Maintenance` ;
2. ouvrir la section de restauration de base ;
3. choisir un dump disponible ;
4. vérifier sa version ;
5. confirmer la restauration ;
6. attendre le contrôle de santé ;
7. examiner les journaux et l'intégrité.

Immich crée un point de restauration de la base actuelle avant l'opération et tente un retour automatique si la restauration échoue.

Nouvelle instance avec onboarding

Avant le premier démarrage, replacer les dossiers de l'ancienne instance dans le nouvel

`UPLOAD_LOCATION` :

```
backups
encoded-video
library
profile
thumbs
upload
```

Configurer également les bibliothèques externes avec les mêmes chemins internes.

Démarrer Immich, puis :

1. cliquer sur `Restore from backup` ;
2. examiner les contrôles de lecture et d'écriture ;
3. sélectionner un dump présent dans `backups` ;
4. ou envoyer un fichier `.sql.gz` ;
5. confirmer ;
6. attendre les migrations et le contrôle de santé.

Compatibilité

Immich affiche un indicateur de compatibilité de version. Restaurer de préférence un dump produit par une version identique ou compatible.

Pour une source v2.7.5 et une cible v3, conserver une copie complète avant la migration. Si le diagnostic devient difficile, restaurer d'abord dans une instance v2.7.5 isolée, puis effectuer la montée de version.

Après restauration

- contrôler les comptes ;
- ouvrir plusieurs originaux ;
- vérifier albums, favoris et partages ;
- examiner `Administration > Maintenance` ;
- lancer les tâches manquantes ;
- vérifier les bibliothèques externes ;
- tester OAuth et mobile ;
- créer un nouveau dump.

Ne pas confondre

La restauration de la base ne restaure pas les photos. Les fichiers doivent déjà être présents et cohérents avec les chemins du dump.

Référence : <https://docs.immich.app/administration/backup-and-restore/>

10.2 - Restauration PostgreSQL en ligne de commande

La restauration en ligne de commande est réservée aux cas avancés. Utiliser un nouvel emplacement PostgreSQL vide au lieu de supprimer immédiatement l'ancien.

Préparer un nouvel emplacement

Arrêter l'instance :

```
cd /srv/immich/app  
docker compose down
```

Dans `.env`, pointer temporairement vers un nouveau dossier vide et explicite :

```
DB_DATA_LOCATION=/srv/immich/postgres-restore-2026-08-07
```

Créer les conteneurs sans lancer le serveur :

```
docker compose pull  
docker compose create  
docker start immich_postgres
```

Adapter le nom du conteneur PostgreSQL.

Vérifier PostgreSQL

```
docker exec immich_postgres pg_isready -U postgres -d immich
```

Restaurer le dump

```
gunzip --stdout /srv/backups/immich/dump.sql.gz \  
| sed "s/SELECT pg_catalog.set_config('search_path', '', false);/SELECT  
pg_catalog.set_config('search_path', 'public, pg_catalog', true);/g" \  
| docker exec -i immich_postgres \  
    psql --dbname=immich --username=postgres \  
    --single-transaction --set ON_ERROR_STOP=on
```

Cette commande suit le principe documenté par Immich. Adapter les valeurs sans ajouter le mot de passe dans la ligne de commande.

Démarrer Immich

```
docker compose up -d  
docker compose logs -f immich-server
```

Si le serveur démarre trop tôt

Dans certains déploiements, utiliser temporairement :

```
DB_SKIP_MIGRATIONS=true
```

Restaurer la base, retirer la variable puis recréer les conteneurs. Ne pas laisser cette variable active.

Revenir sans destruction

Si la restauration échoue :

1. arrêter les conteneurs ;
2. conserver le nouvel emplacement pour analyse ;
3. remettre dans `.env` l'ancien `DB_DATA_LOCATION` ;
4. redémarrer l'ancienne instance ;
5. ne pas mélanger les deux bases.

TrueNAS

L'application TrueNAS rend difficile le démarrage isolé de PostgreSQL. Préférer l'interface de restauration. Pour une intervention manuelle, effectuer d'abord snapshots et dump, puis suivre les instructions correspondant exactement à la version du catalogue.

Référence : <https://docs.immich.app/administration/backup-and-restore/>

10.3 - Plan de reprise après sinistre

Scénario

Le serveur Immich ou son pool principal est perdu. Les éléments disponibles sont :

- dump PostgreSQL ;
- copie des médias ;
- configuration ;
- secrets ;
- bibliothèques externes ;
- documentation des versions et montages.

Ordre de reprise

1. isoler l'incident et protéger les sauvegardes ;
2. choisir un nouvel hôte ;
3. installer une version compatible d'Immich ;
4. recréer les datasets ou dossiers ;
5. restaurer les médias ;
6. restaurer les bibliothèques externes ;
7. configurer les mêmes chemins internes ;
8. restaurer PostgreSQL ;
9. démarrer sans exposition publique ;
10. vérifier l'intégrité ;
11. tester les fonctions ;
12. basculer le reverse proxy et le DNS ;
13. créer une nouvelle sauvegarde.

Priorité des dossiers

Si seuls les originaux sont disponibles :

```
upload  
library  
profile
```

Les dossiers suivants peuvent être régénérés :

```
thumbs  
encoded-video
```

La régénération peut prendre longtemps et solliciter fortement CPU, stockage et GPU.

Validation technique

```
docker compose ps  
docker exec immich_server immich-admin schema-check  
docker exec immich_postgres pg_isready -U postgres -d immich  
df -hT
```

TrueNAS :

```
sudo zpool status  
sudo zfs list  
nvidia-smi
```

Validation fonctionnelle

- connexion locale ;
- utilisateurs ;
- photos de plusieurs dates ;
- vidéos ;
- albums et favoris ;
- visages et recherche ;
- bibliothèque externe ;
- Authentik ;
- application mobile ;
- upload ;

- sauvegarde automatique.

Preuves et compte rendu

Documenter :

- origine de chaque sauvegarde ;
- sommes de contrôle ;
- version restaurée ;
- durée de chaque étape ;
- fichiers manquants ;
- décisions prises ;
- tests réalisés ;
- nouvelles mesures de prévention.

Exercice

Réaliser un exercice isolé au moins périodiquement. Un plan non testé ne garantit ni la lisibilité du dump, ni la présence des médias, ni la connaissance des secrets nécessaires.

10.4 - Erreurs fréquentes et diagnostic

Conteneur PostgreSQL ou pgvecto_upgrade en erreur

```
sudo docker ps -a --format '{{.Names}} {{.Image}} {{.Status}}' | grep -Ei  
'immich|postgres|pgvecto|vector'  
sudo docker logs --tail 300 NOM_CONTENEUR  
sudo zfs list  
df -hT
```

Chercher la première erreur, pas seulement `exit 1`. Vérifier version PostgreSQL, extension, permissions, espace et migration interrompue. Ne pas supprimer `pgData`.

Erreur Permission denied

```
sudo docker exec NOM_CONTENEUR_IMMICH id  
sudo docker inspect NOM_CONTENEUR_IMMICH --format '{{range .Mounts}}{{println .Source "->"  
.Destination}}{{end}}'  
sudo stat -c '%u:%g %a %n' CHEMIN_DATASET  
sudo getfacl CHEMIN_DATASET
```

Corriger l'utilisateur ou l'ACL exacte, pas avec `777`.

Fichier `.immich` absent

Vérifier que le bon dataset est monté au bon chemin et que l'application peut lire et écrire. Ne pas recréer manuellement les marqueurs pour masquer un montage incorrect.

Port 30041 inaccessible

```
sudo ss -ltnp | grep 30041  
sudo docker ps --format '{{.Names}} {{.Ports}}' | grep -i immich  
curl -I http://127.0.0.1:30041
```

Reverse proxy 525 ou 403

- 525 : certificat ou négociation TLS avec l'origine ;
- 403 : règle Cloudflare, Authentik, pare-feu ou application ;
- comparer accès direct local et domaine public ;
- examiner Caddy et Cloudflare.

```
curl -vI https://immich.dreamsaphir.net  
journalctl -u caddy -n 100 --no-pager
```

Upload interrompu

- taille maximale du proxy ;
- délai de 600 secondes ;
- espace libre ;
- réseau mobile ;
- limite Cloudflare ;
- journaux serveur.

GPU absent

```
nvidia-smi  
sudo docker exec NOM_CONTENEUR_IMMICH nvidia-smi  
sudo docker exec NOM_CONTENEUR_ML nvidia-smi
```

Si le premier fonctionne et pas les suivants, vérifier l'allocation GPU des applications. Après une mise à jour TrueNAS, contrôler le pilote et redéployer l'application.

OAuth en boucle

- Issuer URL ;
- Client ID et secret ;
- trois URI de redirection ;
- domaine public ;
- heure système ;
- certificat ;
- accès mobile `app.immich:///oauth-callback`.

Diagnostic fourni

```
IMMICH_COMPOSE_DIR=/srv/immich/app ./scripts/diagnostic_immich.sh
```

Examiner et masquer les informations sensibles avant partage.