Backup NAS avec Rclone

- Sauvegarde TrueNAS vers Hetzner Storage Box

Sauvegarde TrueNAS vers Hetzner Storage Box

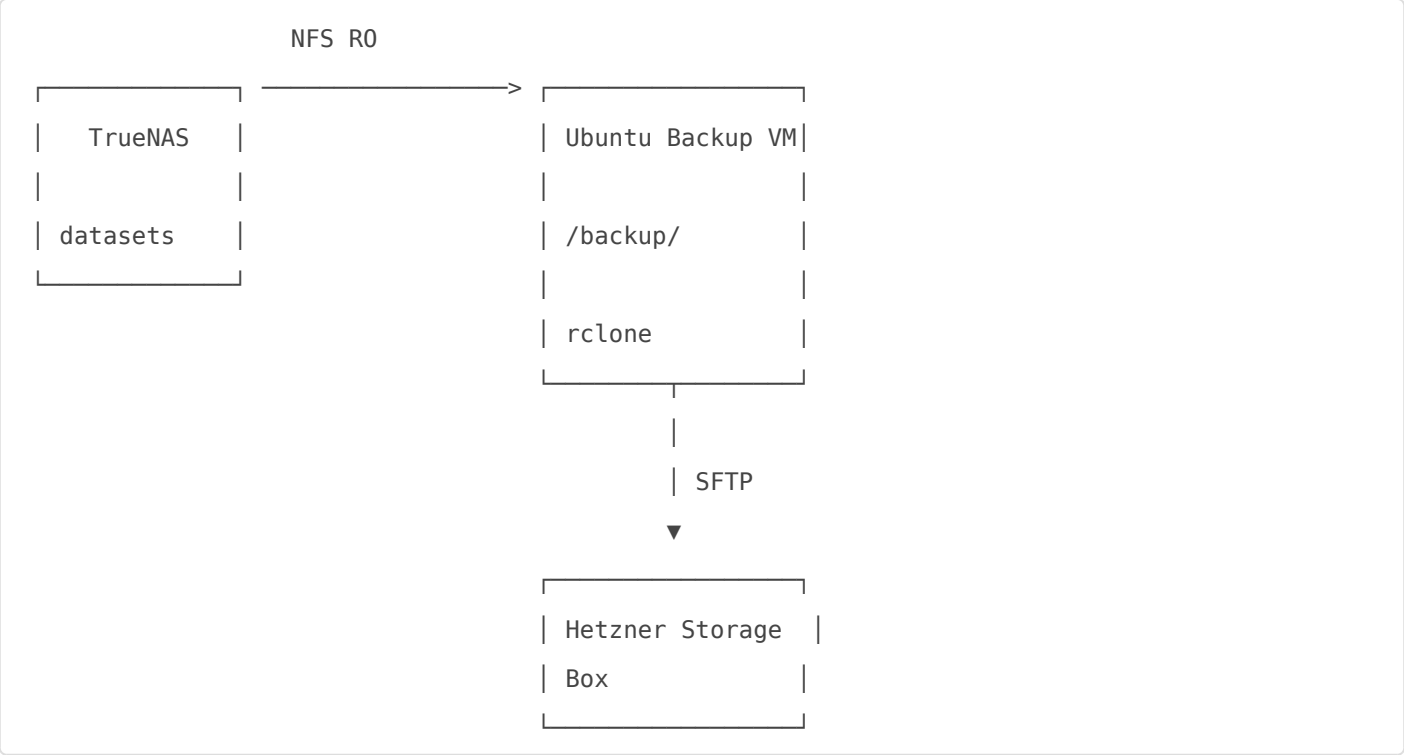
1. Objectif

Cette infrastructure permet de sauvegarder automatiquement certains datasets d'un serveur **TrueNAS** vers un **Hetzner Storage Box**.

La sauvegarde est réalisée par une VM **Ubuntu Server** dédiée, qui :

1. monte les datasets TrueNAS via **NFS** en lecture seule ;
2. utilise **Rclone** pour synchroniser les fichiers vers Hetzner ;
3. exécute la sauvegarde quotidiennement via **cron** ;
4. conserve les logs localement via `/var/log/rclone-backup.log`.

Architecture



Le serveur Ubuntu est volontairement séparé du NAS afin que le mécanisme de sauvegarde ne dépende pas directement de TrueNAS.

---

2. Pré-requis

- Un serveur TrueNAS fonctionnel
- Des datasets TrueNAS contenant les données à sauvegarder
- Une VM Ubuntu Server
- Une connectivité réseau entre Ubuntu et TrueNAS
- Un abonnement Hetzner Storage Box
- NFS activé sur TrueNAS
- Rclone installé sur Ubuntu

---

3. Configuration TrueNAS

3.1. Datasets

Les datasets à sauvegarder sont exposés individuellement via NFS.

Exemple :

```
tank/
├── immich
├── paperless
└── configs
```

Chaque dataset possède son propre export NFS.

3.2. Sécurité NFS

Les exports NFS doivent idéalement être limités à l'adresse IP de la VM de sauvegarde.

Exemple :

```
Dataset : tank/immich
Client autorisé : 192.168.1.20
Mode : lecture seule
```

La VM de sauvegarde ne doit pas avoir besoin d'écrire sur TrueNAS.

Cette configuration limite les conséquences d'une compromission de la VM de sauvegarde.

---

4. Configuration de la VM Ubuntu

4.1. Installation des paquets

```
sudo apt update
sudo apt install nfs-common rclone
```

Vérification :

```
rclone version
```

---

5. Montage des datasets NFS

5.1. Création des points de montage

Exemple :

```
sudo mkdir -p /backup/immich
sudo mkdir -p /backup/paperless
sudo mkdir -p /backup/configs
```

5.2. Test manuel

Exemple :

```
sudo mount -t nfs truenas:/mnt/tank/immich /backup/immich
```

Vérification :

```
ls -lah /backup/immich
```

5.3. Montage permanent

Les montages sont définis dans `/etc/fstab`.

Exemple :

```
truenas:/mnt/tank/immich    /backup/immich    nfs    ro,hard,_netdev,nofail 0 0
truenas:/mnt/tank/paperless /backup/paperless nfs    ro,hard,_netdev,nofail 0 0
truenas:/mnt/tank/configs   /backup/configs   nfs    ro,hard,_netdev,nofail 0 0
```

Tester la configuration :

```
sudo mount -a
```

Puis :

```
mount | grep /backup
```

Les datasets doivent apparaître comme montés en lecture seule.

---

6. Configuration Rclone

6.1. Création du remote

Lancer :

```
rclone config
```

Créer un nouveau remote nommé :

```
hetzner
```

Type :

```
SFTP
```

Utiliser les informations fournies par Hetzner pour le Storage Box.

Tester la connexion :

```
rclone lsd hetzner:
```

6.2. Création du répertoire distant

Créer le répertoire dédié aux sauvegardes :

```
rclone mkdir hetzner:truenas
```

L'arborescence distante sera par exemple :

```
truenas/
├── immich/
├── paperless/
└── configs/
```

---

7. Script de sauvegarde

Le script est situé dans :

```
/usr/local/bin/truenas-backup.sh
```

Contenu actuel :

```
#!/bin/bash

set -euo pipefail

LOG="/var/log/rclone-backup.log"

DATASETS=(
    immich
    paperless
    configs
)

for dataset in "${DATASETS[@]}"; do

    echo "$(date '+%F %T') - Backup $dataset" >> "$LOG"

    rclone sync \
        "/backup/$dataset" \
        "hetzner:truenas/$dataset" \
        --fast-list \
        --transfers 8 \
        --checkers 8 \
        --log-file="$LOG" \
        --log-level INFO

done

echo "$(date '+%F %T') - Backup terminé" >> "$LOG"
```

Le rendre exécutable :

```
sudo chmod +x /usr/local/bin/truenas-backup.sh
```

---

8. Fonctionnement du script

8.1. Liste des datasets

La liste :

```
DATASETS=(
    immich
    paperless
    configs
)
```

détermine les datasets à sauvegarder.

Pour ajouter un dataset :

```
DATASETS=(
    immich
    paperless
    configs
    documents
)
```

Le point de montage `/backup/documents` doit évidemment exister.

8.2. Synchronisation

Pour chaque dataset, Rclone exécute :

```
rclone sync /backup/DATASET hetzner:truenas/DATASET
```

`sync` maintient un miroir entre la source et la destination.

Cela signifie :

- nouveau fichier local → copié vers Hetzner ;
- fichier local modifié → mis à jour sur Hetzner ;
- fichier supprimé localement → supprimé sur Hetzner ;
- fichier identique → aucun transfert.

8.3. `--fast-list`

```
--fast-list
```

Demande à Rclone de privilégier une récupération globale de la liste des fichiers.

Cela peut accélérer les opérations sur des arborescences importantes, au prix d'une consommation mémoire supérieure.

8.4. `--transfers 8`

```
--transfers 8
```

Autorise jusqu'à 8 transferts simultanés.

Cette valeur peut être ajustée en fonction de la connexion réseau et des performances du serveur.

8.5. `--checkers 8`

```
--checkers 8
```

Permet à Rclone d'effectuer plusieurs vérifications de fichiers en parallèle.

8.6. Logs

Les logs sont écrits dans :

```
/var/log/rclone-backup.log
```

Le niveau utilisé est :

```
INFO
```

---

9. Exécution automatique

La sauvegarde est exécutée par `cron`.

Modifier le crontab root :

```
sudo crontab -e
```

Configuration actuelle :

```
15 2 * * * /usr/local/bin/truenas-backup.sh
```

La sauvegarde est donc exécutée tous les jours à **02:15**.

---

10. Rotation des logs

Pour éviter que le fichier de log grossisse indéfiniment, utiliser `logrotate`.

Créer :

```
sudo nano /etc/logrotate.d/rclone-backup
```

Contenu :

```
/var/log/rclone-backup.log {
    weekly
    rotate 8
    compress
    missingok
    notifempty
}
```

Cette configuration conserve environ 8 semaines de logs.

---

11. Tests

11.1. Exécution manuelle

Le script peut être lancé manuellement :

```
sudo /usr/local/bin/truenas-backup.sh
```

11.2. Consulter les logs

```
tail -f /var/log/rclone-backup.log
```

Ou :

```
less /var/log/rclone-backup.log
```

11.3. Vérifier le contenu distant

Exemple :

```
rclone ls hetzner:truenas/immich
```

Pour afficher l'arborescence :

```
rclone tree hetzner:truenas/immich
```

---

12. Vérification d'intégrité

Rclone permet de comparer une source avec la sauvegarde distante.

Exemple :

```
rclone check /backup/immich hetzner:truenas/immich
```

Cette commande permet de détecter des différences entre les deux côtés.

Une vérification régulière, par exemple hebdomadaire ou mensuelle, peut être ajoutée indépendamment de la sauvegarde quotidienne.

---

13. Restauration

La restauration consiste à inverser le sens de la synchronisation.

Exemple :

```
rclone copy \
    hetzner:truenas/immich \
    /restore/immich
```

`copy` est volontairement utilisé ici plutôt que `sync`.

Cela évite qu'une erreur de manipulation sur la source de restauration entraîne des suppressions sur le Storage Box.

Après vérification des données restaurées, celles-ci peuvent être remises sur TrueNAS.

---

14. Attention : la sauvegarde actuelle est un miroir

La configuration actuelle utilise :

```
rclone sync
```

sans mécanisme de versionnage.

Le Storage Box contient donc une copie miroir de TrueNAS.

Exemple :

```
Jour 1
TrueNAS       Hetzner
photo.jpg  →  photo.jpg

Jour 2
photo.jpg supprimé

Jour 2 après backup
TrueNAS       Hetzner
                 photo.jpg supprimé
```

Cela protège contre :

- panne du disque TrueNAS ;
- panne du NAS ;
- corruption physique du NAS ;
- perte de la VM ou du serveur ;
- problème local affectant le stockage.

Mais **cela ne protège pas contre une suppression accidentelle ou une corruption logique qui serait ensuite synchronisée**.

Exemple :

```
Utilisateur
    ↓
supprime 10 000 photos
    ↓
rclone sync
    ↓
10 000 photos supprimées du Storage Box
```

Il est donc important de choisir consciemment entre **miroir simple** et **historique de sauvegarde** .

---

15. Option : corbeille distante avec `--backup-dir`

Rclone permet de conserver les fichiers supprimés ou remplacés grâce à l'option :

```
--backup-dir
```

Exemple :

```
rclone sync \
    /backup/immich \
    hetzner:truenas/immich \
    --backup-dir "hetzner:trash/immich/$(date +%F)"
```

Lorsqu'un fichier doit être supprimé ou remplacé sur la destination, Rclone peut alors le déplacer dans le répertoire de sauvegarde au lieu de le supprimer définitivement.

On obtient par exemple :

```
trash/
└── immich/
    ├── 2026-08-07/
    ├── 2026-08-08/
    └── 2026-08-09/
```

Cette approche constitue une **corbeille distante / historique simple**.

Attention

Il faut prévoir un mécanisme de nettoyage des anciennes sauvegardes.

Sinon le Storage Box finira par contenir :

```
copie actuelle
+
tous les fichiers supprimés depuis la mise en place
```

L'espace consommé peut donc augmenter considérablement.

---

16. Option : véritable versionnage avec `rclone`

Une autre possibilité consiste à utiliser un mécanisme de versionnage côté stockage distant.

L'objectif est de pouvoir retrouver l'état d'un fichier à une date antérieure :

```
photo.jpg
photo.jpg.ancienne-version
photo.jpg.version-2026-08-01
...
```

Cette approche est particulièrement intéressante pour les documents susceptibles d'être modifiés, mais elle est généralement moins pertinente pour une bibliothèque de photos essentiellement immuable.

Pour des fichiers comme les photos Immich, le scénario le plus intéressant est souvent :

```
fichiers actuels
+
corbeille/historique des suppressions
+
politique de rétention
```

plutôt qu'un versionnage de chaque fichier.

---

17. Choix recommandé

Pour cette infrastructure, trois niveaux sont possibles :

Stratégie	Protection contre panne	Suppression accidentelle	Corruption logique
`rclone sync` actuel	Oui	Non	Non

Stratégie	Protection contre panne	Suppression accidentelle	Corruption logique
`sync` + `--backup-dir`	Oui	Oui	Partiellement
Versionnage + rétention	Oui	Oui	Oui

La configuration actuelle reste volontairement simple.

Si les données sont importantes, il est recommandé d'ajouter à terme au minimum une **corbeille distante avec une politique de rétention**.

---

18. Maintenance

Ajouter un dataset

1. Créer l'export NFS sur TrueNAS.
2. Créer le point de montage :

```
sudo mkdir -p /backup/nouveau-dataset
```

3. Ajouter le montage dans `/etc/fstab`.
4. Tester :

```
sudo mount -a
```

5. Ajouter le nom dans `DATASETS` :

```
DATASETS=(
    immich
    paperless
    configs
    nouveau-dataset
)
```

6. Tester manuellement le script.

Modifier l'horaire

Modifier :

```
sudo crontab -e
```

Exemple pour 03:30 :

```
30 3 * * * /usr/local/bin/truenas-backup.sh
```

Forcer une sauvegarde

```
sudo /usr/local/bin/truenas-backup.sh
```

Vérifier que cron est actif

Selon la distribution :

```
systemctl status cron
```

Et consulter les journaux :

```
journalctl -u cron
```

ou :

```
grep CRON /var/log/syslog
```

---

19. Résumé de l'installation



Principe général

TrueNAS est la source.

Ubuntu est le moteur de sauvegarde.

Hetzner est la destination.

La VM Ubuntu ne modifie jamais les données TrueNAS : les exports NFS sont montés en lecture seule.

La destination Hetzner est actuellement un miroir des données locales. Toute évolution vers une véritable stratégie de sauvegarde avec historique, corbeille ou versionnage doit donc être décidée explicitement et accompagnée d'une politique de rétention.