

Wiki.js sur Debian 13

FICHE DE PROCÉDURE | Docker Compose • PostgreSQL 17 • Sauvegardes • Synology

Objectif Reproduire un déploiement de wiki privé et tester la reprise des données. Commandes tirées du déploiement PROCYB, à adapter à votre environnement.

Paramètre	Valeur de référence
Hôte Debian / Wiki.js	192.168.1.239 / port 3015
NAS Synology SFTP	192.168.1.241 / partage docker-backups
Projet Docker Compose	/opt/stacks/wikijs
Images	ghcr.io/requarks/wiki:2 et postgres:17-alpine
Sauvegardes	7 jours locaux / 30 jours sur NAS

01 Préparer l'hôte et les répertoires

```
docker --version
docker compose version
sudo mkdir -p /opt/stacks/wikijs/backups
sudo chown -R "$USER:${id -gn}" /opt/stacks/wikijs
chmod 750 /opt/stacks/wikijs
cd /opt/stacks/wikijs
```

02 Générer le fichier .env

```
umask 077
printf 'POSTGRES_DB=wiki\nPOSTGRES_USER=wikijs\nPOSTGRES_PASSWORD=%s\nWIKI_PORT=3015\n' \
"$(openssl rand -hex 32)" > .env
chmod 600 .env
printf '.env\nbackups/\n' > .gitignore
```

Sécurité Ne jamais publier .env, une clé SSH privée, ni la sortie d'une commande qui révèle des secrets.

03 Créer compose.yaml

```
services:
  db:
    image: postgres:17-alpine
    restart: unless-stopped
    environment:
      POSTGRES_DB: ${POSTGRES_DB}
      POSTGRES_USER: ${POSTGRES_USER}
      POSTGRES_PASSWORD: ${POSTGRES_PASSWORD}
    volumes:
      - postgres_data:/var/lib/postgresql/data
    networks: [wiki_backend]
    healthcheck:
      test: ["CMD-SHELL", "pg_isready -U \"$$POSTGRES_USER\" -d \"$$POSTGRES_DB\""]
      interval: 10s
      timeout: 5s
      retries: 5
      start_period: 20s
  wiki:
    image: ghcr.io/requarks/wiki:2
    restart: unless-stopped
    init: true
    depends_on:
      db:
        condition: service_healthy
    environment:
      DB_TYPE: postgres
      DB_HOST: db
      DB_PORT: 5432
      DB_USER: ${POSTGRES_USER}
      DB_PASS: ${POSTGRES_PASSWORD}
      DB_NAME: ${POSTGRES_DB}
    ports: [ "${WIKI_PORT}:3000" ]
    networks: [wiki_backend]
volumes:
  postgres_data:
networks:
  wiki_backend:
    driver: bridge
```

04 Déployer et contrôler

```
docker compose config --quiet
docker compose pull
docker compose up -d
docker compose ps
docker compose logs --tail=30 wiki
```

Contrôle attendu : PostgreSQL healthy, Wiki.js Up, connexion base réussie dans les logs. PostgreSQL n'a pas de port publié sur le LAN.

05 Première configuration et sécurité applicative

Ouvrir `http://192.168.1.239:3015` ; créer le compte administrateur et configurer la langue française dans Administration → Locale.

Dans Administration → Groupes → Guests → Page Rules, supprimer les règles de lecture générale non souhaitées (par exemple « Path Starts With / »). Publier une page de test et vérifier, en navigation privée, l'affichage de « Non autorisé ».

Périmètre Accès privé recommandé : filtrage réseau / VPN WireGuard. Le mapping du port 3015 de la configuration de référence écoute sur les interfaces de la VM : ne pas ouvrir ce port sur Internet.

06 Sauvegarde manuelle et inspection

```
cd /opt/stacks/wikijs
umask 077
docker compose exec -T db pg_dump -U wikijs -d wiki -Fc \
  > backups/wikijs-test.dump
docker compose exec -T db pg_restore -l \
  < backups/wikijs-test.dump
ls -lh backups/
```

Le catalogue lisible ne prouve pas à lui seul qu'une restauration fonctionnera. La procédure doit inclure un essai d'import.

07 Script local backup.sh

```
#!/usr/bin/env bash
set -Eeuo pipefail
umask 077
cd /opt/stacks/wikijs
BACKUP_DIR="$PWD/backups"
mkdir -p "$BACKUP_DIR"
chmod 700 "$BACKUP_DIR"
exec 9>"$PWD/.backup.lock"
flock -n 9 || exit 1
DATE=$(date +%Y%m%d_%H%M%S)
BACKUP="$BACKUP_DIR/wikijs_${DATE}.dump"
TEMP="${BACKUP}.tmp"
trap 'rm -f -- "$TEMP"' EXIT
docker compose exec -T db pg_dump -U wikijs -d wiki -Fc > "$TEMP"
test -s "$TEMP"
docker compose exec -T db pg_restore -l < "$TEMP" > /dev/null
mv -- "$TEMP" "$BACKUP"
find "$BACKUP_DIR" -maxdepth 1 -type f -name 'wikijs_*.dump' \
  -mtime +6 -delete
echo "Sauvegarde créée : $BACKUP"
```

```
chmod 700 backup.sh
./backup.sh
ls -lh backups/
```

Important Le script traite la base PostgreSQL. Vérifier séparément l'existence d'éventuels contenus externes à la base avant de considérer la sauvegarde comme exhaustive.

08 Restaurer dans une base isolée

```
docker compose exec -T db createdb -U wikijs wiki_restore_test
docker compose exec -T db pg_restore -U wikijs -d wiki_restore_test \
  --no-owner --no-acl < backups/wikijs-test.dump
docker compose exec -T db psql -U wikijs -d wiki_restore_test -c '\dt'
docker compose exec -T db psql -U wikijs -d wiki_restore_test \
  -c 'SELECT count(*) FROM pages;'
# UNIQUEMENT après vérification :
docker compose exec -T db dropdb -U wikijs wiki_restore_test
```

09 Préparer l'externalisation SFTP sur Synology

Dans DSM, créer le partage docker-backups et un compte non administrateur svc_wikijs_backup autorisé en écriture uniquement sur ce partage. Activer SFTP (sans exposer le port à Internet).

Sur Debian, générer une clé dédiée. Installer sa partie publique dans authorized_keys du compte Synology. Protéger .ssh (700) et authorized_keys (600). Ne jamais copier la clé privée sur le NAS.

```
mkdir -p ~/.ssh && chmod 700 ~/.ssh
ssh-keygen -t ed25519 -f ~/.ssh/id_ed25519_wikijs_backup \
-C 'wikijs-backup'
chmod 600 ~/.ssh/id_ed25519_wikijs_backup
sftp -o BatchMode=yes -o IdentitiesOnly=yes \
-i ~/.ssh/id_ed25519_wikijs_backup \
svc_wikijs_backup@192.168.1.241
```

10 Script de transfert sync-nas.sh

```
#!/usr/bin/env bash
set -Eeuo pipefail
umask 077
BACKUP_DIR='/opt/stacks/wikijs/backups'
BACKUP=$(find "$BACKUP_DIR" -maxdepth 1 -type f -name 'wikijs_*.dump' \
-printf '%T@ %p\n' | sort -nr | head -n 1 | cut -d' ' -f2-)
[[ -n "$BACKUP" ]] || { echo 'Aucun dump'; exit 1; }
NAME=$(basename "$BACKUP")
sftp -o BatchMode=yes -o IdentitiesOnly=yes -o ConnectTimeout=15 \
-i /home/laurent/.ssh/id_ed25519_wikijs_backup \
-b - svc_wikijs_backup@192.168.1.241 <<EOF
cd /docker-backups
put $BACKUP $NAME
EOF
echo "Transfert terminé : $NAME"
```

```
chmod 700 sync-nas.sh
./sync-nas.sh
```

Vérification ponctuelle : récupérer une archive distante via sftp get, puis comparer son sha256sum à l'archive originale. Une empreinte identique prouve que ces deux fichiers sont identiques, pas que la restauration applicative est garantie.

11 Programmer la chaîne complète

```
crontab -e
# Une seule ligne dans la crontab utilisateur :
0 2 * * * { /opt/stacks/wikijs/backup.sh && /opt/stacks/wikijs/sync-nas.sh; } >> /opt/stacks/wikijs/backups/backup.log 2>&1
```

Test manuel : exécuter la chaîne, examiner le code retour et tail -n 20 backups/backup.log. Vérifier ensuite la première exécution automatique réelle.

12 Rotation distante sur NAS (30 jours)

Sur le Synology du laboratoire : volume ext4 avec partage chiffré eCryptfs. Script ci-dessous à installer comme /usr/local/etc/scripts/rotate-wikijs-backups.sh (700) ; planification DSM quotidienne à 04:00, par root. Tester d'abord en remplaçant la suppression par -print.

```
#!/bin/sh
set -eu
BACKUP_DIR='/volume1/docker-backups'
LOG='/var/log/rotate-wikijs-backups.log'
mount | grep -F " on $BACKUP_DIR type ecryptfs " >/dev/null || exit 1
[ -d "$BACKUP_DIR" ] || exit 1
echo "$(date) Début rotation" >> "$LOG"
find "$BACKUP_DIR" -maxdepth 1 -type f -name 'wikijs_*.dump' \
    -mtime +29 -exec sh -c '
    for file do echo "$(date) Suppression : $file"; rm -- "$file"; done
    ' sh {} + >> "$LOG" 2>&1
echo "$(date) Rotation terminée" >> "$LOG"
```

13 Rotation du journal sur Debian

Fichier /etc/logrotate.d/wikijs-backup :

```
/opt/stacks/wikijs/backups/backup.log {
    weekly
    rotate 4
    compress
    delaycompress
    missingok
    notifempty
    su laurent laurent
    create 0600 laurent laurent
}
```

```
chmod 600 /opt/stacks/wikijs/backups/backup.log
sudo /usr/sbin/logrotate -d /etc/logrotate.d/wikijs-backup
systemctl status logrotate.timer --no-pager
```

14 Contrôles d'exploitation

Fréquence	Vérification
Après déploiement	Wiki.js répond ; DB healthy ; groupe Guests testé
Quotidienne	Nouveau dump local, copie NAS, absence d'erreurs
Mensuelle	Restaurer un dump en base isolée ; tester accès et rétention
Avant mise à jour	Sauvegarde vérifiée ; version des images et retour arrière documentés
Incident	Conserver les journaux ; ne pas écraser la base active avant diagnostic

Limites / sécurité Sur le NAS, le compte SFTP conserve des droits de suppression sur les fichiers accessibles. Le chiffrement eCryptfs ne protège pas d'un effacement autorisé : prévoir une copie indépendante ou non modifiable. Les permissions affichées par SSH peuvent être influencées par les ACL DSM.

15 Fiche de validation du déploiement

- [] Docker Compose démarré et base saine
- [] Assistant Wiki.js achevé et compte admin créé
- [] Accès anonyme refusé sur la page test
- [] Archive locale créée et catalogue lisible
- [] Restauration testée en base temporaire
- [] SFTP par clé opérationnel, transfert vérifié
- [] Planifications cron / DSM / logrotate contrôlées
- [] Emplacement et procédure de reprise documentés

Référence : article « Installer, sécuriser et sauvegarder Wiki.js avec Docker Compose sur Debian 13 » — PROCYB. Cette fiche résume le laboratoire et ne remplace pas les adaptations à un environnement de production.