

NOUVEAU

TUTORIEL PRATIQUE



nextcloud

EXPOSER NEXTCLOUD DERRIERE TRAEFIK 3.5 (DOCKER COMPOSE)

De l'installation de base aux premières manipulations

POINTS CLES DU TUTORIEL



Installation pas à pas de NEXTCLOUD avec Docker Compose



Exposition de NEXTCLOUD derrière Traefik 3.5



1^{ère} connexion à NEXTCLOUD et manipulations POST-INSTALLATION

Note :

Ce tutoriel suppose que vous avez une machine Debian 13 ou un conteneur LXC Proxmox à disposition et que vous avez mis en place le reverse proxy © Traefik 3.5, sur une autre machine, conformément au tutoriel présent sur notre site : <https://tutos-info.fr>.

L'objectif de ce tutoriel est d'exposer © NEXTCLOUD derrière le reverse proxy © Traefik. Nextcloud permet de gérer ses données sur son propre serveur Cloud, sans devoir céder la souveraineté des données à des entreprises comme Google ou Apple. Avec un mélange d'auto-hébergement et de Cloud computing privé, Nextcloud offre une protection des données meilleure que celle des fournisseurs américains commerciaux. Avec Nextcloud, les entreprises et les particuliers sans service informatique dédié bénéficient des avantages d'un Cloud en gestion autonome sur des serveurs privés ou hébergés.

Pour réaliser ce tutoriel il vous faut :

- Une machine Debian 13 fonctionnelle avec 1 Go de RAM et 20 Go d'espace disque
- Ou un conteneur LXC Debian avec les mêmes caractéristiques
- Le reverse proxy © Traefik 3.5.0 installé sur une machine (voir tutoriel sur notre site : <https://tutos-info.fr>)

1 – MISE A JOUR DES PAQUETS

- Connectez-vous à votre nouvelle machine Debian (ou conteneur LXC) avec un utilisateur disposant des droits « sudo » et mettez à jour les paquets avec la commande suivante :

```
sudo apt update && apt upgrade -y
```

2 – INSTALLATION DE DOCKER ET DOCKER-COMPOSE AVEC LE SCRIPT OFFICIEL – SAUF SI DÉJÀ FAIT

En « mode laboratoire », il est possible d'installer Docker et docker Compose avec le script officiel. L'utilisation du script rend l'installation rapide et simple. Pour cela, saisissez les commandes suivantes :

```
apt install curl -y  
curl -fsSL https://get.docker.com | sh
```

Note : Si vous travaillez en « production », il est recommandé d'ajouter votre utilisateur système au groupe docker. Ainsi, il ne sera plus nécessaire de saisir "sudo" avant toute commande. Pour cela, réalisez les étapes suivantes :

```
# Création du groupe "docker" :  
groupadd docker
```

```
# Attribution du groupe à un utilisateur :  
usermod -aG docker votre_nom_user
```

Déconnectez votre session Debian et reconnectez-vous.

Note : vous pouvez également installer Docker en vous aidant du site officiel et des sources officielles. Lien vers le site officiel : [Debian | Docker Docs](#)

Pour vérifier que Docker est bien installé sur votre machine Debian, saisissez la commande suivante :

```
sudo docker version
```

L'environnement Docker installé s'affiche :

```
debian@CT-DEBIAN-DOCKER:~$ sudo docker version
[sudo] password for debian:
Client: Docker Engine - Community
 Version:      28.3.3
 API version:  1.51
 Go version:   go1.24.5
 Git commit:   980b856
 Built:        Fri Jul 25 11:34:00 2025
 OS/Arch:      linux/amd64
 Context:      default

Server: Docker Engine - Community
 Engine:
  Version:      28.3.3
  API version:  1.51 (minimum version 1.24)
  Go version:   go1.24.5
  Git commit:   bea959c
  Built:        Fri Jul 25 11:34:00 2025
  OS/Arch:      linux/amd64
  Experimental: false
 containerd:
  Version:      1.7.27
  GitCommit:    05044ec0a9a75232cad458027ca83437aae3f4da
 runc:
  Version:      1.2.5
  GitCommit:    v1.2.5-0-g59923ef
 docker-init:
  Version:      0.19.0
  GitCommit:    de40ad0
```

Pour afficher la version Docker Compose installée, saisissez la commande suivante :

sudo docker compose version

```
debian@CT-DEBIAN-DOCKER:~$ sudo docker compose version
Docker Compose version v2.39.1
```

En cas de problème, vous pouvez désinstaller Docker et Docker Compose avec les commandes suivantes :

systemctl stop docker

apt purge docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

1^{ère} étape : création des dossiers et sous-dossiers

Dans cette étape, nous créons les dossiers et sous-dossiers utiles à la création de notre « stack » NEXTCLOUD avec les commandes suivantes (nous mettons les dossiers dans « /opt/containers ») :

```
sudo mkdir -p /opt/containers/nextcloud/{data,config,apps,db,php,backup}
```

On se place ensuite dans le dossier adéquat avec la commande suivante :

```
cd /opt/containers/nextcloud
```

2^{ème} étape : création d'un fichier « .env » (fichier caché avec les variables d'environnement)

- Créez le fichier « .env » dans « /opt/containers/nextcloud » à l'aide de la commande suivante :

```
sudo nano /opt/containers/nextcloud/.env
```

- Collez ou saisissez le contenu suivant :

```
# ---- Nextcloud -----
NC_DOMAIN=nextcloud.votre_domaine.fr
NC_ADMIN_USER=ncadmin
NC_ADMIN_PASSWORD=ChangeMe-Admin!

# ---- Base MariaDB -----
NC_DB_NAME=nextcloud
NC_DB_USER=ncuser
NC_DB_PASSWORD=ChangeMe-DB!
DB_ROOT_PASSWORD=ChangeMe-Root!

# ---- Redis (verrouillage fichiers/perf) -----
REDIS_PASSWORD=ChangeMe-Redis!

# ---- Divers -----
TZ=Europe/Paris
```

- Quittez et sauvegardez le fichier avec **CTRL + O** et pressez la touche **Entrée**.

3^{ème} étape : création du fichier « overrides.ini » (options PHP)

- Copiez la commande suivante :

```
sudo tee /opt/containers/nextcloud/php/overrides.ini >/dev/null <<'INI'
file_uploads = On
memory_limit = 1024M
upload_max_filesize = 2G
post_max_size = 2G
max_execution_time = 3600
max_input_time = 3600
output_buffering = Off
date.timezone = Europe/Paris
INI
```

4^{ème} étape : création du fichier « docker-compose.yml » avec la dernière version de WORDPRESS

- Créez le fichier « **docker-compose.yml** » dans « **/opt/containers/nextcloud** » à l'aide de la commande suivante :

sudo nano /opt/containers/nextcloud/docker-compose.yml

- Collez ou saisissez le contenu suivant (**le fichier commence par 3 tirets**) :

```
---

services:
  db:
    image: mariadb:10.11
    restart: unless-stopped
    environment:
      MARIADB_ROOT_PASSWORD: "${DB_ROOT_PASSWORD}"
      MARIADB_DATABASE: "${NC_DB_NAME}"
      MARIADB_USER: "${NC_DB_USER}"
      MARIADB_PASSWORD: "${NC_DB_PASSWORD}"
      TZ: "${TZ}"
    # Recos Nextcloud (isolation + row binlog)
    command: [
      "--transaction-isolation=READ-COMMITTED",
      "--binlog-format=ROW",
      "--character-set-server=utf8mb4",
      "--collation-server=utf8mb4_general_ci"
    ]
    volumes:
      - ./db:/var/lib/mysql
    healthcheck:
      test: ["CMD", "mysqladmin", "ping", "-h", "127.0.0.1", "-u", "${NC_DB_USER}", "--password=${NC_DB_PASSWORD}"]
      interval: 5s
      timeout: 3s
      retries: 30
      start_period: 20s

  redis:
    image: redis:7-alpine
    restart: unless-stopped
    command: ["redis-server", "--appendonly", "yes", "--requirepass", "${REDIS_PASSWORD}"]

  nextcloud:
    image: nextcloud:stable-apache
    restart: unless-stopped
    depends_on:
      db:
        condition: service_healthy
      redis:
        condition: service_started
    environment:
      MYSQL_HOST: db
      MYSQL_DATABASE: "${NC_DB_NAME}"
      MYSQL_USER: "${NC_DB_USER}"
      MYSQL_PASSWORD: "${NC_DB_PASSWORD}"
```

```

# Auto-install 1er démarrage
NEXTCLOUD_ADMIN_USER: "${NC_ADMIN_USER}"
NEXTCLOUD_ADMIN_PASSWORD: "${NC_ADMIN_PASSWORD}"

# Derrière Traefik (reverse proxy)
NEXTCLOUD_TRUSTED_DOMAINS: "${NC_DOMAIN}"
TRUSTED_PROXIES: "10.10.10.14"
OVERWRITEPROTOCOL: "https"

# Redis
REDIS_HOST: redis
REDIS_HOST_PASSWORD: "${REDIS_PASSWORD}"

TZ: "${TZ}"
volumes:
# Persistance recommandée
- ./data:/var/www/html/data
- ./config:/var/www/html/config
- ./apps:/var/www/html/custom_apps
# Overrides PHP
- ./php/overrides.ini:/usr/local/etc/php/conf.d/overrides.ini:ro
ports:
- "8802:80" # ← exposé pour Traefik (10.10.10.14 -> 10.10.10.15:8802)

# Tâches planifiées Nextcloud (toutes les 5 min)
nc-cron:
image: nextcloud:stable-apache
restart: unless-stopped
depends_on:
nextcloud:
condition: service_started
volumes:
- ./data:/var/www/html/data
- ./config:/var/www/html/config
- ./apps:/var/www/html/custom_apps
entrypoint: ["/bin/sh", "-c", "while true; do su -s /bin/sh -c 'php -f /var/www/html/cron.php' www-data; sleep 300; done"]

```

- Quittez et sauvegardez le fichier avec **CTRL + O** et pressez la touche **Entrée**.

5^{ème} étape : création du fichier de configuration dynamique sur Traefik

On crée maintenant le fichier de configuration dynamique (service et routeur) dans notre machine Traefik à l'aide de la commande suivante :

sudo nano /etc/traefik/dynamic/nextcloud.yml

- Copiez ou saisissez le contenu suivant :

```

http:
routers:
# Router HTTPS (principal)
nc-https:
rule: Host(`nextcloud.votre_domaine.fr`)
entryPoints: [websecure]

```

```

service: nc-svc
tls:
  certResolver: le
  middlewares: [nc-sec]

# Router HTTP -> redirection vers HTTPS
nc-http:
  rule: Host(`nextcloud.votre_domaine.fr`)
  entryPoints: [web]
  middlewares: [redirect-to-https]
  service: nc-svc

middlewares:
  # Redirection HTTP -> HTTPS
  redirect-to-https:
    redirectScheme:
      scheme: https

# En-têtes de sécurité recommandés par Nextcloud
nc-sec:
  headers:
    # HSTS (Nextcloud demande >= 15552000s ; ici ~2 ans)
    stsSeconds: 63072000
    stsIncludeSubdomains: true
    stsPreload: true

    # X-Frame-Options: SAMEORIGIN (ne pas utiliser frameDeny en même temps)
    customFrameOptionsValue: "SAMEORIGIN"

    # X-Content-Type-Options
    contentTypeNosniff: true

    # Referrer-Policy
    referrerPolicy: no-referrer

    # Quelques durcissements additionnels
    permissionsPolicy: "camera=(), microphone=(), geolocation=()"
    addVaryHeader: true

    # Laisse Nextcloud générer sa CSP ; n'ajoute pas de Content-Security-Policy ici.
    # Assure la bonne détection HTTPS côté app
    customRequestHeaders:
      "X-Forwarded-Proto": "https"

    # (Optionnel) quelques headers utiles supplémentaires côté réponse
    customResponseHeaders:
      "X-Download-Options": "noopen"
      "X-Permitted-Cross-Domain-Policies": "none"

services:
  # Backend Nextcloud (Apache dans le conteneur)
  nc-svc:
    loadBalancer:
      servers:
        - url: "http://ip_machine_nextcloud:8802"
      passHostHeader: true

```

- Quittez et sauvegardez le fichier avec **CTRL + O** et pressez la touche **Entrée**.

5^{ème} étape : lancement de la création de la stack « NEXTCLOUD »

Sur votre machine NEXTCLOUD, effectuez les manipulations suivantes :

```
cd /opt/containers/nextcloud
sudo docker compose up -d
```

Patientez pendant la création de la stack complète NEXTCLOUD :

```
[+] Running 4/5
✓ Network nextcloud_default          Created
✓ Container nextcloud-redis-1        Started
. Container nextcloud-db-1           Waiting
✓ Container nextcloud-nextcloud-1    Created
✓ Container nextcloud-nc-cron-1      Created
```

Une fois la stack créée, vous devez obtenir ceci :

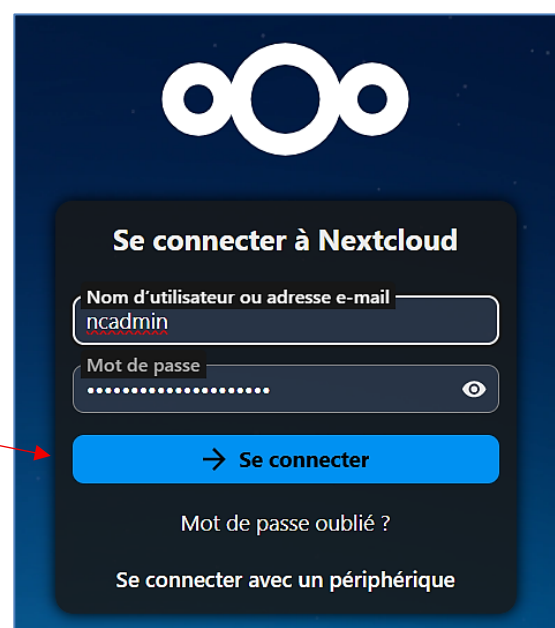
```
✓ Container nextcloud-db-1           Healthy
✓ Container nextcloud-nextcloud-1    Started
✓ Container nextcloud-nc-cron-1      Started
```

Portainer affiche les nouveaux conteneurs en lien avec notre stack NEXTCLOUD :

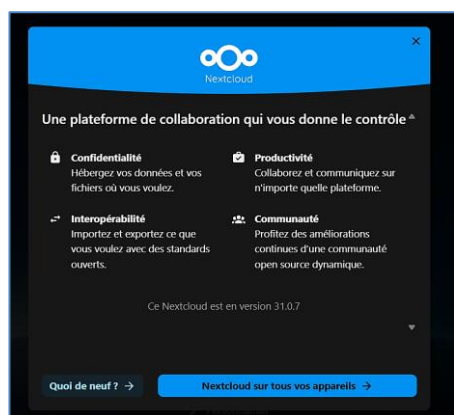
☐	nextcloud-db-1	healthy	   	nextcloud	mariadb:10.11	2025-08-17 11:45:28	172.20.0.2	-	 administrators
☐	nextcloud-nc-cron-1	running	   	nextcloud	nextcloud:stable-apache	2025-08-17 11:45:30	172.20.0.5	-	 administrators
☐	nextcloud-nextcloud-1	running	   	nextcloud	nextcloud:stable-apache	2025-08-17 11:45:29	172.20.0.4	8802:80	 administrators
☐	nextcloud-redis-1	running	   	nextcloud	redis:7-alpine	2025-08-17 11:45:28	172.20.0.3	-	 administrators

5 – TEST DE CONNEXION A NEXTCLOUD

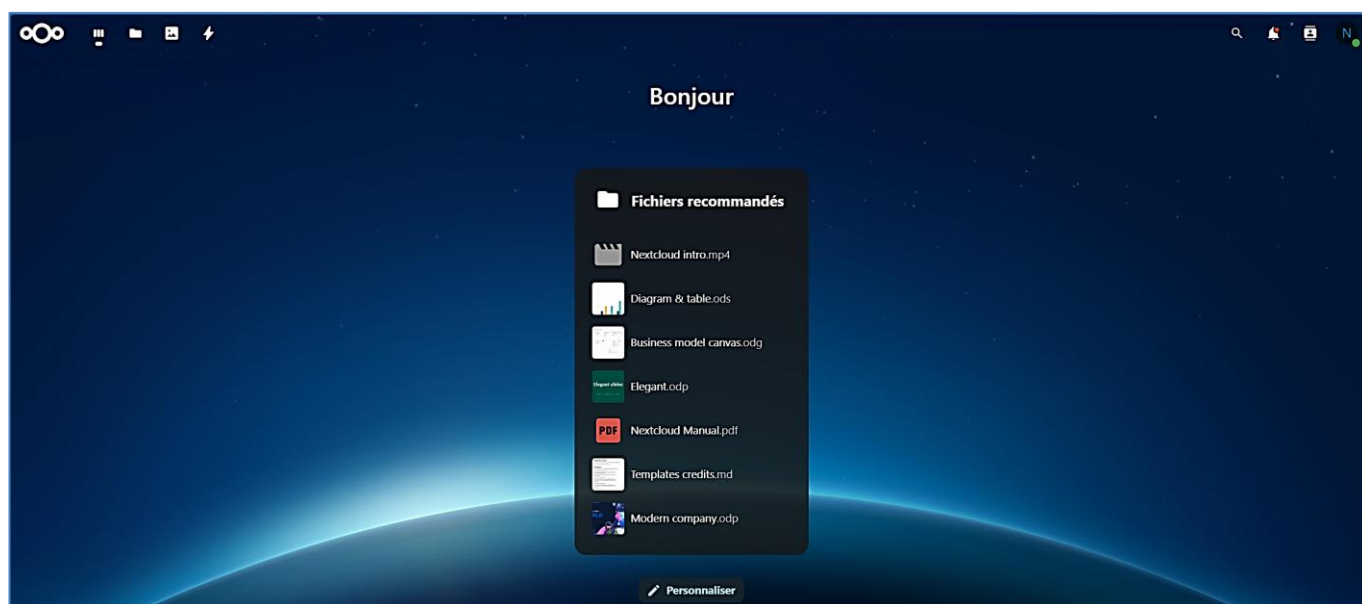
- Lancez votre navigateur et saisissez, dans la barre d'adresses, votre nom de domaine (ou votre WAN). Si vous avez tout configuré correctement, NEXTCLOUD affiche une première fenêtre vous demandant une authentification :
 - Saisissez « **ncadmin** » dans le nom d'utilisateur et le mot de passe que vous avez défini dans le fichier « .env » puis cliquez le bouton bleu « **Se connecter** »



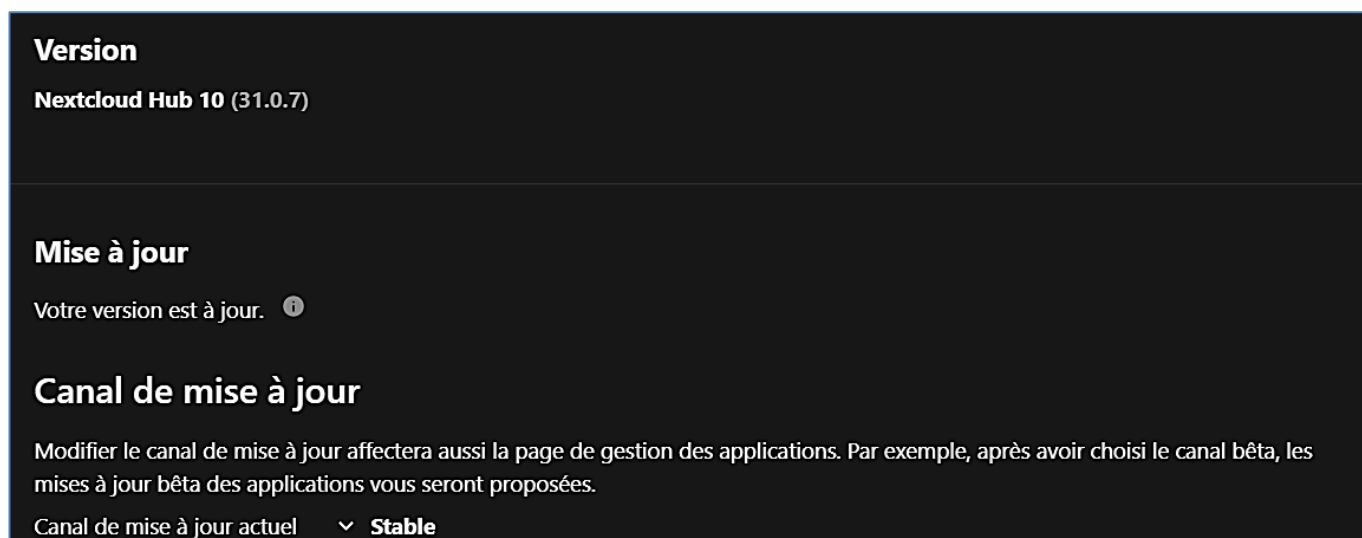
Le premier écran affiche un message de bienvenue ; vous pouvez le fermer en cliquant la croix en haut à droite :



L'interface de gestion (écran d'accueil) de Nextcloud s'affiche :



Vous pouvez personnaliser l'écran d'accueil de l'interface de gestion en cliquant le bouton « **Personnaliser** ». Si vous cliquez sur l'icône « **N** » en haut à droite et sur « **Paramètres d'administration** » vous pourrez contrôler la version de Nextcloud installée :



Lorsque vous affichez les paramètres d'administration, il est possible qu'un message de mise à jour soit affiché comme ici (nous avons installé la version 31.0.7 et la mise à jour 31.0.8 est disponible) :

Mise à jour

Une nouvelle version est disponible : **Nextcloud 31.0.8**

Toutes les applications ont une version compatible avec cette version de Nextcloud.

Télécharger maintenant

La mise à jour par l'interface web est désactivée. Veuillez utiliser la mise à jour par ligne de commande ou le mécanisme de mise à jour approprié pour votre méthode d'installation (par exemple, Docker pull) pour effectuer la mise à jour.

Le lien « Télécharger maintenant » ne doit pas être utilisé car il télécharge juste un paquet « zip » sur votre machine Windows. La procédure de mise à jour est la suivante :

Etape préparatoire (facultative mais conseillée) : préparation des sauvegardes

- Depuis votre machine Nextcloud, copiez les commandes suivantes :

```
cd /opt/containers/nextcloud
```

```
# Sauvegarde (dump) de la base de données
```

```
sudo docker exec $(sudo docker compose ps -q db) \
sh -lc 'mysqldump --single-transaction --default-character-set=utf8mb4 \
-u"$MARIADB_USER" -p"$MARIADB_PASSWORD" "$MARIADB_DATABASE" \
| sudo tee "backup/ncdb-$(date +%F-%H%M).sql" >/dev/null
```

```
# Données (data/config/apps)
```

```
sudo tar -C /opt/containers/nextcloud -czf \
backup/nextcloud-$(date +%F-%H%M).tgz data config apps
```

1^{ère} étape : passage en mode maintenance et arrêt de Cron

- Sur votre machine Nextcloud, collez les commandes suivantes :

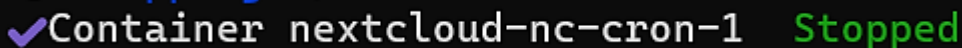
```
cd /opt/containers/nextcloud
```

```
sudo docker compose exec -u www-data -T nextcloud php occ maintenance:mode --on
```



```
Maintenance mode enabled
```

```
sudo docker compose stop nc-cron
```



```
✓ Container nextcloud-nc-cron-1 Stopped
```

2^{ème} étape : mise à jour du fichier « docker-compose.yml »

- Editez le fichier « **docker-compose.yml** » afin de **modifier le tag des services « nextcloud » et « nc-cron »** avec la commande suivante :

```
sudo nano /opt/containers/nextcloud/docker-compose.yml
```

- Modifiez les tags des services « nextcloud » et « nc-cron » :

```
nextcloud:
  image: nextcloud:31.0.8-apache
```

```
nc-cron:
  image: nextcloud:31.0.8-apache
```

- Quittez et sauvegardez le fichier avec **CTRL + O** et pressez la touche **Entrée**.

3^{ème} étape : récupérer la nouvelle image et la redéployer

- Sur la machine Nextcloud, copiez les commandes suivantes :

```
sudo docker compose pull nextcloud nc-cron
sudo docker compose up -d nextcloud nc-cron
```

4^{ème} étape : lancer la migration des applications

- Sur la machine Nextcloud, copiez les commandes suivantes :

```
# Upgrade Nextcloud
sudo docker compose exec -u www-data -T nextcloud php occ upgrade -vvv

# Finitions utiles
sudo docker compose exec -u www-data -T nextcloud php occ db:add-missing-indices
sudo docker compose exec -u www-data -T nextcloud php occ db:convert-filecache-bigint --no-interaction
sudo docker compose exec -u www-data -T nextcloud php occ maintenance:repair --include-expensive

# Mettre à jour toutes les apps
sudo docker compose exec -u www-data -T nextcloud php occ app:update --all
```

5^{ème} étape : sortir du mode maintenance et relancer le cron

- Sur la machine Nextcloud, copiez les commandes suivantes :

```
sudo docker compose exec -u www-data -T nextcloud php occ maintenance:mode --off
sudo docker compose start nc-cron
```

6^{ème} étape : vérification de la version Nextcloud

- Sur la machine Nextcloud, copiez la commande suivante :

```
sudo docker compose exec -u www-data -T nextcloud php occ -V
```

```
debian@CT-DEBIAN-DOCKER:/opt/containers/nextcloud$ sudo docker compose exec -u www-data -T nextcloud php occ -V
Nextcloud 31.0.8
```

```
sudo docker compose ps
```

NAME	PORTS	IMAGE	COMMAND	SERVICE	CREATED	STATUS
nextcloud-db-1		mariadb:10.11	"docker-entrypoint.s..."	db	2 hours ago	Up 2 hours (heal
thy)	3306/tcp					
nextcloud-nc-cron-1	80/tcp	nextcloud:31.0.8-apache	"/bin/sh -c 'while t..."	nc-cron	17 minutes ago	Up 16 minutes
nextcloud-nextcloud-1		nextcloud:31.0.8-apache	"/entrypoint.sh apac..."	nextcloud	28 minutes ago	Up 28 minutes
0.0.0.0:8802->80/tcp, [::]:8802->80/tcp						
nextcloud-redis-1	6379/tcp	redis:7-alpine	"docker-entrypoint.s..."	redis	2 hours ago	Up 2 hours

`sudo docker compose images | grep nextcloud`

nextcloud-db-1 292 years ago	mariadb	10.11	linux/amd64	9f59ffd33b9a	327MB
nextcloud-nc-cron-1 292 years ago	nextcloud	31.0.8-apache	linux/amd64	b5fcb9799d0d	1.45GB
nextcloud-nextcloud-1 292 years ago	nextcloud	31.0.8-apache	linux/amd64	b5fcb9799d0d	1.45GB
nextcloud-redis-1 292 years ago	redis	7-alpine	linux/amd64	f218e591b571	41.4MB

Vérification depuis les paramètres d'administration de l'interface de gestion de Nextcloud :

Version

Nextcloud Hub 10 (31.0.8)

Mise à jour

Votre version est à jour. 

FICHER « .env » - SUR LA MACHINE NEXTCLOUD

```
# ---- Nextcloud -----
NC_DOMAIN=nextcloud.votre_domaine.fr
NC_ADMIN_USER=ncadmin
NC_ADMIN_PASSWORD=ChangeMe-Admin!

# ---- Base MariaDB -----
NC_DB_NAME=nextcloud
NC_DB_USER=ncuser
NC_DB_PASSWORD=ChangeMe-DB!
DB_ROOT_PASSWORD=ChangeMe-Root!

# ---- Redis (verrouillage fichiers/perf) -----
REDIS_PASSWORD=ChangeMe-Redis!

# ---- Divers -----
TZ=Europe/Paris
```

FICHER « docker-compose.yml » - SUR LA MACHINE NEXTCLOUD

```
---

services:
  db:
    image: mariadb:10.11
    restart: unless-stopped
    environment:
      MARIADB_ROOT_PASSWORD: "${DB_ROOT_PASSWORD}"
      MARIADB_DATABASE: "${NC_DB_NAME}"
      MARIADB_USER: "${NC_DB_USER}"
      MARIADB_PASSWORD: "${NC_DB_PASSWORD}"
      TZ: "${TZ}"
    # Recos Nextcloud (isolation + row binlog)
    command: [
      "--transaction-isolation=READ-COMMITTED",
      "--binlog-format=ROW",
      "--character-set-server=utf8mb4",
      "--collation-server=utf8mb4_general_ci"
    ]
    volumes:
      - ./db:/var/lib/mysql
    healthcheck:
      test: ["CMD", "mysqladmin", "ping", "-h", "127.0.0.1", "-u", "${NC_DB_USER}", "--password=${NC_DB_PASSWORD}"]
      interval: 5s
      timeout: 3s
      retries: 30
      start_period: 20s

  redis:
    image: redis:7-alpine
    restart: unless-stopped
    command: ["redis-server", "--appendonly", "yes", "--requirepass", "${REDIS_PASSWORD}"]
```

```

nextcloud:
  image: nextcloud:stable-apache
  restart: unless-stopped
  depends_on:
    db:
      condition: service_healthy
    redis:
      condition: service_started
  environment:
    MYSQL_HOST: db
    MYSQL_DATABASE: "${NC_DB_NAME}"
    MYSQL_USER: "${NC_DB_USER}"
    MYSQL_PASSWORD: "${NC_DB_PASSWORD}"

# Auto-install 1er démarrage
NEXTCLOUD_ADMIN_USER: "${NC_ADMIN_USER}"
NEXTCLOUD_ADMIN_PASSWORD: "${NC_ADMIN_PASSWORD}"

# Derrière Traefik (reverse proxy)
NEXTCLOUD_TRUSTED_DOMAINS: "${NC_DOMAIN}"
TRUSTED_PROXIES: "10.10.10.14"
OVERWRITEPROTOCOL: "https"

# Redis
REDIS_HOST: redis
REDIS_HOST_PASSWORD: "${REDIS_PASSWORD}"

TZ: "${TZ}"
volumes:
  # Persistance recommandée
  - ./data:/var/www/html/data
  - ./config:/var/www/html/config
  - ./apps:/var/www/html/custom_apps
  # Overrides PHP
  - ./php/overrides.ini:/usr/local/etc/php/conf.d/overrides.ini:ro
ports:
  - "8802:80" # ← exposé pour Traefik (10.10.10.14 -> 10.10.10.15:8802)

```

```

# Tâches planifiées Nextcloud (toutes les 5 min)

```

```

nc-cron:
  image: nextcloud:stable-apache
  restart: unless-stopped
  depends_on:
    nextcloud:
      condition: service_started
  volumes:
    - ./data:/var/www/html/data
    - ./config:/var/www/html/config
    - ./apps:/var/www/html/custom_apps
  entrypoint: ["/bin/sh", "-c", "while true; do su -s /bin/sh -c 'php -f /var/www/html/cron.php' www-data; sleep 300; done"]

```

http:

routers:

Router HTTPS (principal)

nc-https:

rule: Host(`nextcloud.votre\_domaine.fr`)

entryPoints: [websecure]

service: nc-svc

tls:

certResolver: le

middlewares: [nc-sec]

Router HTTP -> redirection vers HTTPS

nc-http:

rule: Host(`nextcloud.votre\_domaine.fr`)

entryPoints: [web]

middlewares: [redirect-to-https]

service: nc-svc

middlewares:

Redirection HTTP -> HTTPS

redirect-to-https:

redirectScheme:

scheme: https

En-têtes de sécurité recommandés par Nextcloud

nc-sec:

headers:

HSTS (Nextcloud demande >= 15552000s ; ici ~2 ans)

stsSeconds: 63072000

stsIncludeSubdomains: true

stsPreload: true

X-Frame-Options: SAMEORIGIN (ne pas utiliser frameDeny en même temps)

customFrameOptionsValue: "SAMEORIGIN"

X-Content-Type-Options

contentTypeNosniff: true

Referrer-Policy

referrerPolicy: no-referrer

Quelques durcissements additionnels

permissionsPolicy: "camera=(), microphone=(), geolocation=()"

addVaryHeader: true

Laisse Nextcloud générer sa CSP ; n'ajoute pas de Content-Security-Policy ici.

Assure la bonne détection HTTPS côté app

customRequestHeaders:

"X-Forwarded-Proto": "https"

(Optionnel) quelques headers utiles supplémentaires côté réponse

customResponseHeaders:

"X-Download-Options": "noopen"

"X-Permitted-Cross-Domain-Policies": "none"

```
services:
  # Backend Nextcloud (Apache dans le conteneur)
  nc-svc:
    loadBalancer:
      servers:
        - url: "http://ip_machine_nextcloud:8802"
      passHostHeader: true
```