Navidrome

Référence : Navidrome 0.60.3 — Ubuntu 24.04 — conteneur Incus

Guide de reconstruction. Il décrit comment rebâtir cette machine et pourquoi elle est réglée ainsi.

Le fichier `.md` de cette page : ouvrir le `.md`

Ce que cette page couvre : un serveur de musique posé dans un conteneur, qui donne accès à la bibliothèque du NAS depuis l'extérieur de la maison — sur un téléphone, et dans la voiture par Android Auto. L'installation, les deux montages qui rendent la bibliothèque visible, la configuration, le mandataire inverse qui l'expose en HTTPS, le transcodage, et la circulation des listes de lecture entre ce serveur et celui de la maison.

Contexte : un hôte Incus portant plusieurs conteneurs en macvlan, chacun avec son adresse sur le réseau domestique ; un NAS sur la même machine ; et un conteneur web qui sert déjà de mandataire inverse pour d'autres services. Les adresses, les noms d'hôtes et les noms de dossiers sont des exemples à transposer.

Trois choses valent le détour même sans suivre le guide de bout en bout. Dans le fichier de configuration, un titre de section avale tout ce qui le suit, et un réglage général écrit trop bas cesse d'exister sans qu'aucun message ne le dise — section *La configuration*. La bibliothèque n'est visible qu'au prix de deux montages et non d'un, pour une raison qui ne se devine pas — section *Rendre la bibliothèque visible*. Et le scan quotidien tourne dans un processus distinct, ce qui fait apparaître au journal un second numéro de processus que l'on prend volontiers pour un redémarrage — section *Vérifier*.

Convention : chaque bloc de commandes indique la machine où il s'exécute. Un bloc unique se copie tel quel ; des blocs séparés signifient qu'une édition ou une décision intervient entre eux. Les commandes `incus` se tapent sur l'hôte,

dans une session ouverte par `ssh hostadmin@192.168.0.11`.

À quoi sert cette machine

La bibliothèque musicale vit sur le NAS et se joue dans la maison par le serveur Lyrion, qui alimente les platines. Rien de tout cela ne sort du réseau local.

Ce conteneur est la sortie. Il expose la même bibliothèque — ou plutôt une sélection de celle-ci — derrière une adresse publique en HTTPS, avec un compte et un mot de passe, et il transcode à la volée ce qui doit l'être pour un téléphone en 4G. L'application Symfonium s'y connecte par le protocole Subsonic, et Android Auto affiche la bibliothèque sur l'écran de la voiture.

Il ne modifie jamais la bibliothèque. Ses deux montages sont en lecture seule ; tout ce qu'il produit — sa base, ses caches, ses vignettes — vit dans son propre stockage. C'est ce qui permet de l'exposer sur Internet sans ouvrir un chemin d'écriture vers le NAS.

Le conteneur

La création d'un conteneur, les profils et le stockage sont décrits dans Incus — hôte et conteneur & serveur NAS. Ce qui suit ne retient que ce qui est propre à cette machine.

Le conteneur est non privilégié, démarre avec l'hôte, et porte les deux profils du parc : `default` et `macvlan`, ce dernier lui donnant sa propre adresse sur le réseau local plutôt qu'une adresse privée derrière l'hôte.

Sur l'hôte — la création :

```
incus launch images:ubuntu/24.04/cloud navidrome -p default -p macvlan
```

```
incus config set navidrome boot.autostart true
```

□□ Sans les deux profils, le conteneur naît sans réseau. Si le profil `macvlan` n'existe pas encore sur l'hôte, il se crée en deux commandes : `incus profile create macvlan`, puis `incus profile device add macvlan eth0 nic nictype=macvlan parent=eno1`.

L'adresse et les résolveurs

Le conteneur porte une adresse fixe, posée dans son propre netplan — elle voyage donc avec lui si l'instance est exportée puis réimportée ailleurs.

Sur l'hôte — ouvrir l'éditeur :

```
nano /tmp/50-static-public-ip.yaml
```

Contenu du fichier, à coller dans l'éditeur ouvert ci-dessus :

```
network:
  version: 2
  renderer: networkd
  ethernets:
    eth0:
      dhcp4: no
      dhcp6: no
      addresses: [192.168.0.8/24]
      routes:
        - to: default
          via: 192.168.0.1
      nameservers:
        addresses: [192.168.0.5, 192.168.0.1]
```

Sur l'hôte — la pose et la relecture :

```
incus file push /tmp/50-static-public-ip.yaml \
  navidrome/etc/netplan/50-static-public-ip.yaml \
```

```
  --uid 0 --gid 0 --mode 0600
incus file pull navidrome/etc/netplan/50-static-public-ip.yaml -
incus exec navidrome -- netplan apply
```

⚠ La relecture n’est pas décorative. Un chemin de dépôt fautif ne produit pas d’erreur : le fichier est simplement créé ailleurs, et la machine démarre avec une configuration réseau qui n’est pas celle qu’on croit avoir posée. Le mode `0600` n’est pas un raffinement non plus — netplan se plaint d’un fichier plus ouvert que cela.

Le second résolveur est le routeur, et ce n’est pas un choix par défaut. Sans lui, le conteneur perd toute résolution dès que le filtre DNS s’arrête. Il ne peut pas être un résolveur public : le routeur détourne vers le filtre toute requête adressée hors du réseau local, si bien qu’une adresse extérieure renverrait à la machine en panne.

Le conteneur ne configure pas son réseau par cloud-init, qui est neutralisé par un fichier posé à la construction :

```
CFG=/etc/cloud/cloud.cfg.d/99-disable-network-config.cfg
incus exec navidrome -- sh -c "echo 'network: {config: disabled}' > $CFG"
incus exec navidrome -- rm -f /etc/netplan/50-cloud-init.yaml
```

Vérifications :

```
incus list navidrome
incus exec navidrome -- ping -c2 192.168.0.1
incus exec navidrome -- getent hosts github.com
```

Rendre la bibliothèque visible

Le dossier exposé au serveur n’est pas la bibliothèque elle-même : c’est un dossier de liens symboliques qui pointent vers les dossiers réels de la

musique. C’est ce qui permet de choisir ce qui sort de la maison — les formats très lourds, par exemple, n’ont simplement pas de lien.

 Cette sélection est la seule chose qui limite ce qui est exposé sur Internet. Créer un lien de plus, c’est publier un dossier de plus, et rien ne le redemande.

Deux montages sont nécessaires, et non un. Le premier donne le dossier de liens ; le second donne leur destination. Incus ne suit pas un lien symbolique qui sort de son espace de montage : sans le second, les liens sont visibles et ne mènent nulle part. Et il doit être monté au même chemin que sur l’hôte, puisque c’est ce chemin que les liens contiennent.

Sur l’hôte :

```
incus config device add navidrome musique disk \
  source=/media/nas1/Audio/Navidrome \
  path=/mnt/musique \
  readonly=true
incus config device add navidrome musique-source disk \
  source=/media/nas1/Audio/Musique \
  path=/media/nas1/Audio/Musique \
  readonly=true
incus restart navidrome
```

Les deux sont en lecture seule. Le serveur n’écrit jamais dans la bibliothèque, et un montage inscriptible ouvrirait vers le NAS un chemin d’écriture depuis un service exposé sur Internet.

⚠ Un troisième montage est une fausse bonne idée. Donner « accès aux listes de lecture » par un montage dédié revient à monter une seconde fois un sous-dossier déjà visible, et en écriture. Le dossier des listes est atteint par le premier montage ; il n’y a rien à ajouter.

Sur l’hôte — relire ce que porte l’instance :

```
incus config device show navidrome
```

Le décalage des propriétaires

Les fichiers de la bibliothèque appartiennent à `uid=1000` et `gid=150` sur l'hôte. Dans un conteneur non privilégié, ces numéros sont décalés d'un million : sans correspondance explicite, le serveur voit toute la bibliothèque en `nobody`/`nogroup` et ne peut rien lire.

Sur l'hôte — l'arrêt, puis l'éditeur :

```
incus stop navidrome
incus config edit navidrome
```

Dans l'éditeur ouvert ci-dessus, la clé `raw.idmap` doit se présenter ainsi, dans la section `config:` :

```
  raw.idmap: |
    uid 1000 1000
    gid 150 150
```

Sur l'hôte — le démarrage et le contrôle décisif :

```
incus start navidrome
incus exec navidrome -- ls -la /mnt/musique | head
```

Des propriétaires nommés, et non des `nobody`.

⚠ `raw.idmap` se pose par `incus config edit`, jamais par `incus config set` : la valeur tient sur plusieurs lignes, et la barre verticale du YAML est ce qui la rend lisible par Incus.

Cette correspondance sert à lire. Elle ne rend rien inscriptible : ce sont les montages en lecture seule qui décident de cela.

Installer le serveur

Le serveur est un binaire unique, sans dépendances, déposé dans `/opt` et lancé sous le compte `www-data`.

Dans le conteneur :

```
install -d -o www-data /opt/navidrome /var/lib/navidrome
cd /tmp
BASE=https://github.com/navidrome/navidrome/releases/latest/download
wget $BASE/navidrome_linux_amd64.tar.gz
tar -xzf navidrome_linux_amd64.tar.gz -C /opt/navidrome navidrome
chown -R www-data:www-data /opt/navidrome
rm navidrome_linux_amd64.tar.gz
```

On entre dans le conteneur par `incus exec navidrome -- bash` depuis l'hôte. Le compte `www-data` est retenu parce qu'il existe déjà et qu'il ne sert à rien d'autre ici ; ce qui compte est qu'il ne s'agisse pas de root.

Le transcodage a besoin de ffmpeg, qui vient de l'archive Ubuntu :

```
incus exec navidrome -- apt-get update
incus exec navidrome -- apt-get -y install ffmpeg
```

Cette machine n'a aucun dépôt tiers ni PPA. C'est ce qui rend ses montées de version sans histoire, le serveur lui-même n'étant pas un paquet.

La configuration

Le fichier de configuration vit à côté des données du serveur, et non dans `/etc` : c'est le dossier que le service reçoit en propre.

Dans le conteneur — ouvrir l’éditeur :

```
nano /var/lib/navidrome/navidrome.toml
```

Contenu du fichier, à coller dans l’éditeur ouvert ci-dessus :

```
MusicFolder = "/mnt/musique"
DataFolder = "/var/lib/navidrome"
Port = 4533
LogLevel = "info"

# Listes de lecture : chemin RELATIF au MusicFolder.
# Le serveur importe les .m3u, il n'en écrit pas.
PlaylistsPath = "Playlists-Navidrome"

DefaultLanguage = "fr"
EnableFavourites = true
EnableReplayGain = true

# Rend publiques les listes créées dans l'interface.
DefaultPlaylistPublicVisibility = true

# Adresse publique du service, derrière le mandataire inverse.
# Sert à fabriquer les liens de partage.
BaseURL = "https://musique.example.com"

# Tout ce qui suit appartient à la section [Scanner] jusqu'à la fin
# du fichier : aucun réglage général ne peut être placé plus bas.
[Scanner]
# Scan complet chaque 24 h, à l'heure du dernier démarrage du service.
Schedule = "@every 24h"
```

□ Un titre de section avale tout ce qui le suit

C’est le piège central de ce fichier, et il ne produit aucun message.

Dans ce format, `[Scanner]` ouvre une section qui se referme seulement à la fin du fichier. Un réglage général écrit sous cette ligne — l’adresse publique, par

exemple — n’est plus le réglage général : il devient un réglage du scanner, que le serveur ne connaît pas et ignore en silence. Le service démarre, l’interface répond, et le réglage n’existe pas.

Le symétrique existe aussi : à l’intérieur de la section, les clés ne reprennent pas son nom. Écrire `Scanner.FollowSymlinks` sous `[Scanner]` demande en réalité `Scanner.Scanner.FollowSymlinks`, que personne ne lit.

La règle tient en une phrase : les réglages généraux d’abord, les sections en dernier, et à l’intérieur d’une section, le nom court.

Les liens symboliques sont suivis sans qu’aucun réglage ne le demande — ce sont les deux montages qui font ce travail. Un réglage à ce sujet n’a rien à faire ici.

Ce que chaque valeur décide

`PlaylistsPath` est relatif au dossier de musique, jamais absolu. Laissé vide, le serveur importe tout `.m3u` qu’il trouve n’importe où dans la bibliothèque ; la valeur explicite restreint la recherche au seul dossier voulu.

`DefaultPlaylistPublicVisibility` rend publiques les listes créées dans l’interface — tous les comptes du serveur les voient. À passer à `false` là où plusieurs personnes ont chacune les leurs.

`BaseURL` est découpé par le serveur en trois morceaux : le chemin, l’hôte et le protocole. Le chemin est vide ici, le service vivant à la racine de son domaine et non sous un sous-chemin ; l’hôte et le protocole servent à fabriquer les adresses des partages. C’est aussi pourquoi la valeur se lit `""` dans la configuration que l’interface expose : c’est le chemin qui s’y affiche, et il est vide à bon droit.

`Schedule` fixe l’intervalle entre deux scans complets, pas l’heure. Le compte à rebours part du dernier démarrage du service : redémarrer à 20 h 26 déplace le scan quotidien à 20 h 26.

Le service

Dans le conteneur — ouvrir l'éditeur :

```
nano /etc/systemd/system/navidrome.service
```

Contenu du fichier, à coller dans l'éditeur ouvert ci-dessus :

```
[Unit]
Description=Navidrome Music Server
After=network.target

[Service]
User=www-data
WorkingDirectory=/var/lib/navidrome
#Environment=ND_ENABLETRANSCODINGCONFIG=true
ExecStart=/opt/navidrome/navidrome --configfile /var/lib/navidrome/navidrome.toml
ExecStartPost=/bin/bash -c \
  'sleep 2 && mkdir -p /tmp/go-taglib-wasm \
  && chown -R www-data:www-data /tmp/go-taglib-wasm'
Restart=on-failure
RestartSec=5

[Install]
WantedBy=multi-user.target
```

Dans le conteneur — la mise en marche :

```
systemctl daemon-reload
systemctl enable --now navidrome
systemctl status navidrome --no-pager
```

⚠ La ligne `ExecStartPost` n'est pas un ornement. Le lecteur d'étiquettes déballe une bibliothèque dans `/tmp/go-taglib-wasm`, dossier recréé par root à chaque

démarrage. Sans cette correction de propriétaire, le service démarre normalement et échoue à lire les étiquettes, sans message clair. Elle est coupée ici sur trois lignes pour rester recopiable ; systemd recolle à la lecture toute ligne terminée par une barre oblique inverse, et l'écrire d'un seul tenant revient au même.

La ligne `Environment=` est livrée commentée : elle ne sert qu'à ouvrir temporairement l'écran des profils de transcodage, et se recommente ensuite.

Les permissions

Sur l'hôte :

```
incus exec navidrome -- groupadd -g 150 nasgroup
incus exec navidrome -- usermod -aG nasgroup www-data
incus exec navidrome -- chown -R www-data:www-data /var/lib/navidrome
incus exec navidrome -- chmod 1777 /tmp
incus exec navidrome -- systemctl restart navidrome
```

Le groupe `150` porte le même numéro que le groupe des fichiers du NAS : c'est lui, combiné au décalage des propriétaires, qui donne la lecture de la bibliothèque.

L'exposer en HTTPS

Le serveur écoute en clair sur le port `4533` de son adresse locale. Ce qui sort de la maison passe par le conteneur web, qui termine le TLS et relaie.

Dans ISPConfig, site `musique.example.com`, onglet Options, champ Apache Directives :

```
ProxyPreserveHost On
ProxyPass        / http://192.168.0.8:4533/
ProxyPassReverse / http://192.168.0.8:4533/
```

```
RequestHeader set X-Forwarded-Proto "https"
RequestHeader set X-Forwarded-For "%{REMOTE_ADDR}s"
```

Sur l’hôte — activer les modules nécessaires :

```
incus exec web-srv -- a2enmod proxy proxy_http headers
incus exec web-srv -- systemctl reload apache2
```

□□ Juste après l’enregistrement dans ISPConfig, une requête peut renvoyer une 404 d’Apache, puis la suivante la page attendue : le rechargement en douceur est encore en cours. Attendre quelques secondes avant de conclure à une erreur de configuration.

Le certificat

Le certificat générique `*.example.com` couvre déjà ce sous-domaine : ni port 80 ni validation externe ne sont nécessaires. Dans ISPConfig, onglet SSL du site, coller le certificat, sa clé et le paquet d’autorité, puis SSL Action = Save, Let’s Encrypt décoché.

⚠□ Auto-Subdomain : None. Autrement, ISPConfig ajoute `www.musique.example.com` au site, nom que le certificat générique ne couvre pas — un certificat générique ne vaut que pour un seul niveau.

Côté extérieur, il faut que le nom public pointe vers l’adresse publique de la maison et que le routeur redirige le 443 vers le conteneur web.

Écouter

Dans Symfonium, sur le téléphone :

- Fournisseur : OpenSubsonic

- Adresse : `https://musique.example.com`, sans numéro de port
- Compte et mot de passe : ceux créés dans l’interface du serveur
- Débit maximal : 192 kbit/s convient pour la voiture

Android Auto s’active de lui-même une fois la connexion établie.

Le transcodage

Les profils de transcodage se modifient dans l’interface, mais l’écran qui les porte n’est accessible que si le service a été démarré avec la variable qui l’autorise. La manœuvre est donc en trois temps : décommenter, régler, recommenter.

Dans le conteneur — ouvrir l’éditeur et retirer le `#` devant la ligne `Environment=` :

```
nano /etc/systemd/system/navidrome.service
```

Dans le conteneur — appliquer :

```
systemctl daemon-reload
systemctl restart navidrome
```

Dans l’interface, Settings → Transcoding, les deux profils s’écrivent avec le chemin absolu de ffmpeg :

```
/usr/bin/ffmpeg -i %s -ss %t -map 0:a:0 -b:a %bk -v 0 -f mp3 -
```

```
/usr/bin/ffmpeg -i %s -ss %t -map 0:a:0 -b:a %bk -v 0 -c:a libopus -f opus -
```

Le premier produit du MP3, le second de l'Opus, meilleur à débit égal mais moins universellement accepté.

⚠ Sans le chemin absolu, le transcodage échoue en silence : le serveur cherche `ffmpeg` dans un environnement qui n'a pas de `PATH` utile et répond `executable file not found in $PATH`, message qui ne remonte pas jusqu'à l'auditeur — la lecture s'arrête, sans plus.

La ligne `Environment=` se recommente une fois les profils enregistrés, avec le même couple `daemon-reload` et `restart`.

Les listes de lecture

Le serveur importe, il n'exporte pas. Il lit les fichiers `.m3u` du dossier désigné et en fait des listes dans sa base ; une liste créée ou modifiée depuis un téléphone vit dans cette base et ne produit aucun fichier.

Le dossier des listes est un sous-dossier du dossier de musique :

```
/media/nas1/Audio/Navidrome/
  Playlists-Navidrome/
    Playlist - Club.m3u
    Playlist - Jazz.m3u
  24-192/   -> lien vers /media/nas1/Audio/Musique/24-192/
  Flac/     -> lien vers /media/nas1/Audio/Musique/Flac/
```

Les chemins écrits dans les fichiers sont donc relatifs, et remontent d'un cran :

```
#EXTM3U
#EXTINF:217,Music And Lights
```

```
../Flac/Various Artists - 100 Hits Disco/09 - Imagination - Music And Lights.flac
#EXTINF:293,Billie utilisateur
../24-88/Michael Jackson - Thriller/6 - Michael Jackson - Billie utilisateur.flac
```

⚠ Un chemin absolu, ou un chemin sans `../`, ne mène à rien : le serveur les résout depuis le dossier qui contient la liste.

Du serveur de la maison vers celui-ci

L'outil Music Tools Suite récupère les listes du serveur Lyrion, convertit les chemins à la forme ci-dessus et dépose les fichiers dans `Playlists-Navidrome/`. Le déclenchement est manuel ; l'importation, elle, se fait au scan suivant.

⚠ Un avertissement revient régulièrement au journal — `Path in playlist not found` — pour les entrées en `tmp://`. Lyrion emploie cette forme pour une piste lue hors de sa bibliothèque ; la piste est de toute façon hors du dossier exposé. Rien à corriger.

Sans cet outil, la conversion se fait à la main. Sur l'hôte :

```
for f in /media/nas1/Audio/Navidrome/Playlists-Navidrome/*.m3u; do
  sed -i 's|^\([^#]\)|../\1|g' "$f"
  touch "$f"
done
```

Le `touch` est ce qui force la relecture au prochain scan : le serveur se fie à la date de modification.

De celui-ci vers le serveur de la maison

La fonction d'exportation de Symfonium produit un `.m3u` que Music Tools Suite réajuste pour Lyrion. Le cycle fonctionne donc dans les deux sens, à condition de passer par l'outil pour la conversion des chemins.

Vérifier

Où tester

⚠ Un service en macvlan ne se teste pas depuis la machine qui le porte : les paquets partent par l'interface parente, où le macvlan interdit le dialogue entre l'hôte et ses conteneurs. Les trois points de mesure sont donc l'intérieur du conteneur, le conteneur web, et une autre machine du réseau.

Sur l'hôte — de l'intérieur :

```
incus exec navidrome -- systemctl is-active navidrome
incus exec navidrome -- curl -s -o /dev/null \
  -w "%{http_code}\n" http://localhost:4533/ping
```

`active` et `200`.

Sur l'hôte — par le chemin réel du mandataire inverse :

```
incus exec web-srv -- curl -s -o /dev/null \
  -w "%{http_code}\n" http://192.168.0.8:4533/ping
```

Depuis une autre machine du réseau — bout en bout, TLS compris :

```
curl -s -o /dev/null -w "%{http_code}\n" https://musique.example.com/ping
```

Ce que le journal dit, et ce qu'il ne dit pas

Sur l'hôte :

```
incus exec navidrome -- journalctl -u navidrome -n 30 --no-pager
```

Un démarrage sain se termine par une ligne annonçant le serveur prêt sur `0.0.0.0:4533`, suivie d'un scan.

□□ Le scan tourne dans un processus distinct. Le journal montre donc deux numéros de processus : celui du serveur, qui reste, et celui du scan, qui apparaît puis disparaît. Ce second numéro n’est pas un redémarrage, et sa ligne `Loaded configuration` non plus.

□ Un réglage ignoré ne s’écrit nulle part. Le fichier de configuration n’est pas validé : une clé mal placée ou mal nommée est simplement absente, sans avertissement au journal. Le seul contrôle qui tranche est de demander au serveur ce qu’il a retenu.

Sur l’hôte — la configuration effective, telle que l’interface la reçoit :

```
incus exec navidrome -- curl -s http://localhost:4533/app/ \
  | grep -o 'APP_CONFIG__.\{0,300\}'
```

On y lit notamment la langue par défaut — preuve que les réglages généraux du fichier sont bien lus — et le chemin de l’adresse publique, vide lorsque le service vit à la racine de son domaine.

La bibliothèque

Sur l’hôte :

```
incus exec navidrome -- ls /mnt/musique
incus exec navidrome -- ls /mnt/musique/Flac | head
incus exec navidrome -- id www-data
```

La première commande liste les liens exposés, la deuxième prouve qu’ils sont suivis — elle montre du contenu réel —, la troisième doit porter le groupe `150`.

Entretenir

La montée de version

Le serveur est un binaire : sa montée consiste à remplacer le fichier et à relancer le service. Un instantané pris avant la manœuvre permet le retour en arrière.

Sur l’hôte :

```
incus snapshot create navidrome avant-montee
```

Dans le conteneur :

```
cd /tmp
BASE=https://github.com/navidrome/navidrome/releases/latest/download
wget $BASE/navidrome_linux_amd64.tar.gz
systemctl stop navidrome
tar -xzf navidrome_linux_amd64.tar.gz -C /opt/navidrome navidrome
chown www-data:www-data /opt/navidrome/navidrome
systemctl start navidrome
rm navidrome_linux_amd64.tar.gz
```

⚠ La base de données est migrée automatiquement au premier démarrage de la nouvelle version, et cette migration ne se défait pas. Revenir en arrière après coup, c’est revenir à l’instantané, pas seulement à l’ancien binaire.

Ce qui n’a pas à être sauvegardé

Rien dans ce conteneur n’est unique. La bibliothèque vit sur le NAS, les listes viennent du serveur de la maison, et le reste — base, caches, vignettes — se reconstruit par un scan. Ce qui mérite d’être conservé est le fichier de configuration et l’unité du service, tous deux reproduits en entier dans cette page.

Les comptes et les habitudes d’écoute, eux, vivent dans la base. Ils sont couverts par l’export mensuel du conteneur, décrit dans l’article de la sauvegarde — pas par une sauvegarde propre à cette machine.

Le bruit pendant les mises à jour

Un `apt upgrade` dans le conteneur fait apparaître des lignes de ce genre :

```
host0: Failed to write 'change' to '/.../uevent': Permission denied
```

Des paquets liés au matériel demandent au système de re-parcourir les disques ; le conteneur non privilégié voit ces chemins mais n'a pas le droit d'y écrire. Sans effet sur le service.

Ces paquets n'ont aucune raison d'être ici. Leur retrait supprime le bruit et allège l'image.

Sur l'hôte — le filet, puis la simulation :

```
incus snapshot create navidrome avant-allegement
incus exec navidrome -- apt-get purge --dry-run \
  open-vm-tools udisks2 multipath-tools fwupd
```

Sur l'hôte — la purge, puis les orphelins :

```
incus exec navidrome -- apt-get purge -y \
  open-vm-tools udisks2 multipath-tools fwupd
incus exec navidrome -- apt-get autoremove --purge --dry-run
```

⚠ Lire la simulation avant de l'appliquer. Si `ffmpeg`, `netplan.io`, `systemd`, `curl` ou `wget` y apparaissait, s'arrêter : le cas normal ne concerne que la famille matériel, disques et modems.

Sur l'hôte — appliquer, puis revalider :

```
incus exec navidrome -- apt-get autoremove --purge -y
incus exec navidrome -- systemctl is-active navidrome
incus exec navidrome -- curl -s -o /dev/null \
  -w "%{http_code}\n" http://localhost:4533/ping
```

Ce qui n’est pas dans cette page

La création du conteneur, les profils, le stockage et les sauvegardes sont dans Incus — hôte et conteneur & serveur NAS.

Le mandataire inverse et ISPConfig sont décrits dans Serveur web — Apache multi-PHP et ISPConfig, dont cette page ne reprend que le strict nécessaire.

Le serveur de musique de la maison et ses platines ont leurs propres pages ; c’est de lui que viennent les listes de lecture.

Aide-mémoire

Élément	Valeur
Adresse du conteneur	`192.168.0.8`, macvlan sur `eno1`
Profils Incus	`default` + `macvlan`
Binaire	`/opt/navidrome/navidrome`
Configuration	`/var/lib/navidrome/navidrome.toml`
Données, base et caches	`/var/lib/navidrome/`
Unité du service	`/etc/systemd/system/navidrome.service`
Compte du service	`www-data`, membre du groupe `150`
Port d’écoute	`4533`
Bibliothèque exposée	`/mnt/musique` ← `/media/nas1/Audio/Navidrome`
Destination des liens	`/media/nas1/Audio/Musique`, au même chemin
Correspondance d’identifiants	`uid 1000 1000`, `gid 150 150`
Adresse publique	`https://musique.example.com`