PortailSIO

NOUVELLE FORMULE

TUTORIEL

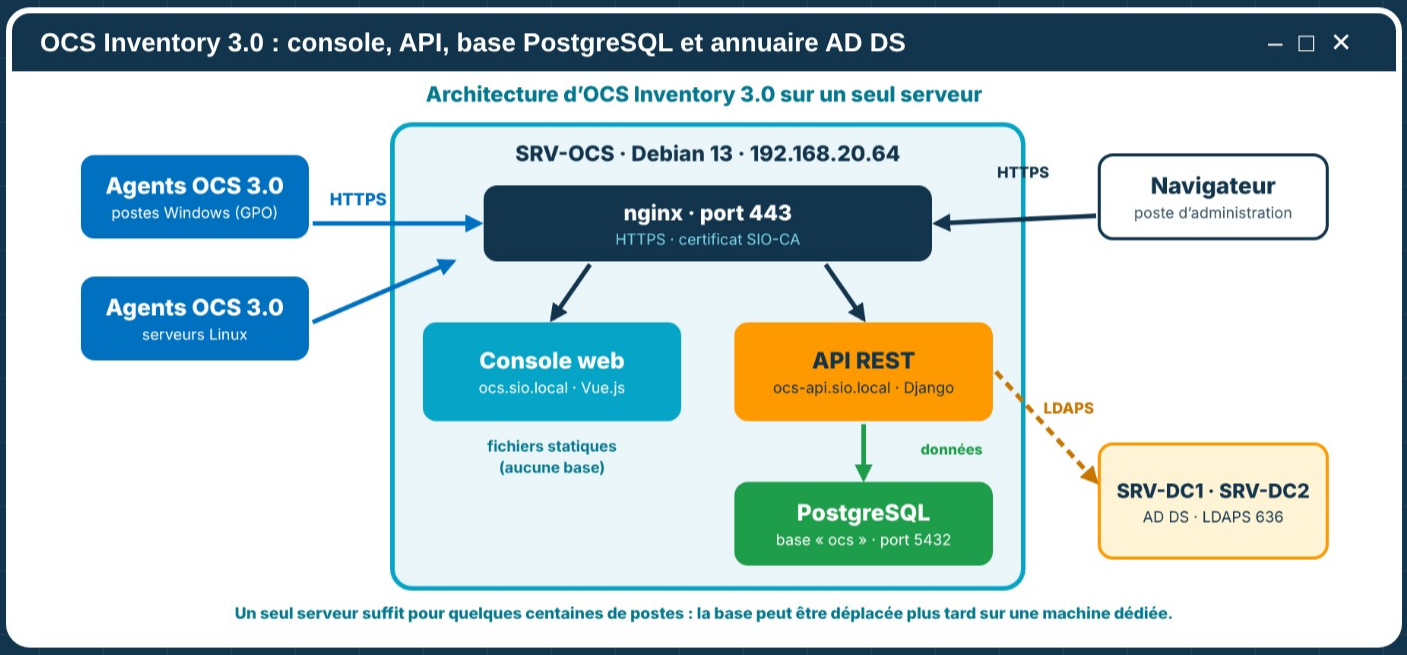
DEBIAN 13 · OCS INVENTORY 3.0

Installer et lier à AD/DS

OCS



NIVEAU
Intermédiaire



1 Préparer le serveur Debian 13 et le DNS

2 Installer PostgreSQL et créer la base

3 Installer OCS Inventory 3.0

4 Publier la console en HTTPS avec SIO-CA

5 Premiers réglages : comptes et premier actif

6 Lier OCS à l'annuaire AD DS

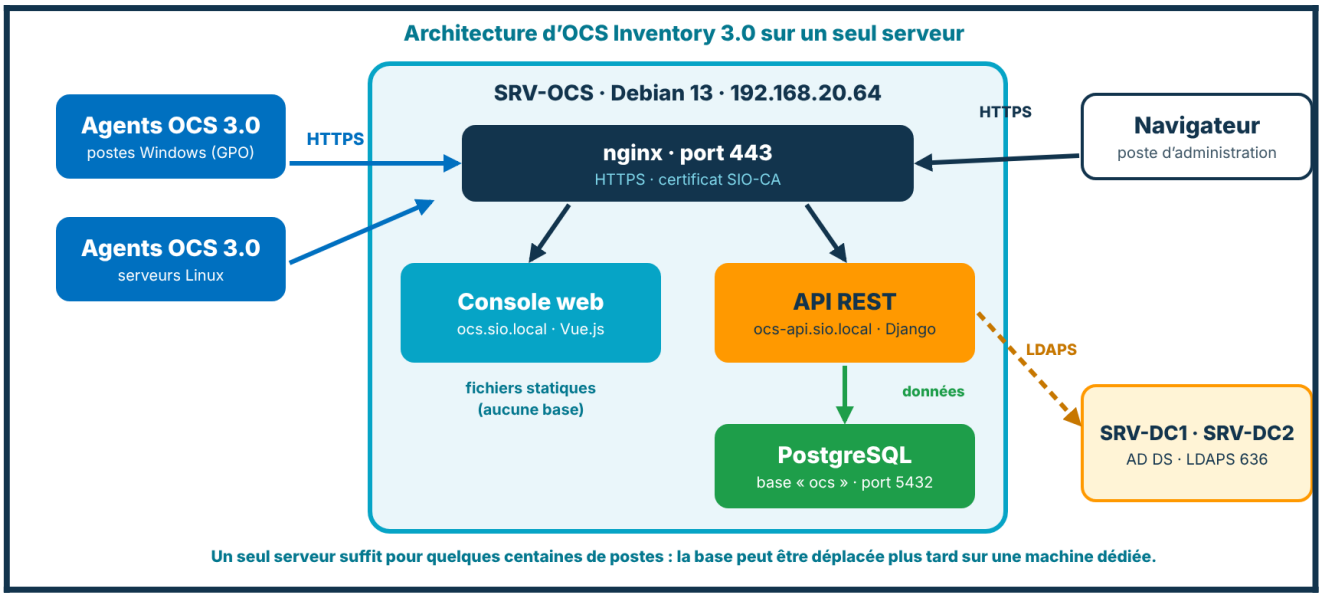
CC BY-NC
Utilisation commerciale interdite
© PortailSIO – 11/2026

Ce tutoriel en ligne
Scannez le QR code. Tous les tutoriels :
portail.sio-ndlp.fr/tutoriels.php

Ce tutoriel a été réalisé avec l'hyperviseur © Proxmox VE. Il peut être réalisé dans un environnement de virtualisation personnel à l'aide des logiciels © VirtualBox ou © VMware Player. Il **suppose que vous disposez d'une machine virtuelle Debian 13** (installation minimale avec serveur SSH) et du domaine **sio.local** construit dans les tutoriels précédents (contrôleurs **SRV-DC1** et **SRV-DC2** sous Windows Server 2025, avec l'autorité de certification **SIO-CA** et **LDAPS** activé : voir le tutoriel « Activer LDAPS sur AD DS »).

OCS Inventory est un logiciel libre d'**inventaire du parc informatique**. Un petit programme, l'**agent**, est installé sur chaque poste et chaque serveur : il envoie régulièrement au serveur la liste du matériel et des logiciels de la machine. L'administrateur consulte ensuite tout le parc depuis une **console web**. La **version 3.0** (octobre 2026) est une refonte complète : l'agent est réécrit (Windows, Linux et macOS), le serveur n'est plus un module Apache mais une **API** (Django, en Python) et la console est une application web moderne.

Le serveur se compose de **quatre éléments**, que nous installons ici sur **une seule machine** : l'API, la console, une base de données **PostgreSQL** et le serveur web **nginx** qui sert les deux. Les comptes de la console sont propres à OCS (un compte `admin` existe à l'installation), mais OCS peut aussi **interroger l'annuaire AD DS** : les administrateurs se connectent alors avec leur mot de passe Windows (étape 6).



Les rôles de chacun

- **Les agents** envoient leur inventaire à l'API en **HTTPS**, avec un compte de service OCS (groupe `AgentInventory`). Le tutoriel suivant les déploie sur les postes Windows par GPO
- **L'API (backend)** reçoit les inventaires, contrôle les droits, dialogue avec la base et avec l'annuaire. Elle tourne sous **uWSGI**, derrière nginx
- **La console (frontend)** est un simple site **statique** (pages et scripts) servi par nginx : elle ne contient aucune donnée et appelle l'API depuis le navigateur de l'administrateur
- **PostgreSQL** conserve les inventaires, les comptes et la configuration

Dans le cadre de ce tutoriel, nous utilisons les machines suivantes (VLAN SERVEURS, réseau 192.168.20.0 /24) :

Machine	Rôle	Système	Adresse IP
SRV-OCS	Serveur OCS Inventory 3.0 : nginx, API, console et PostgreSQL (nom `srv-ocs`)	Debian 13	192.168.20.64
SRV-DC1	Contrôleur de domaine, DNS, annuaire principal, autorité de certification SIO-CA	Windows Server 2025	192.168.20.250
SRV-DC2	Contrôleur de domaine, DNS, annuaire de secours	Windows Server 2025	192.168.20.251
Poste admin	Navigateur web (DNS = les deux contrôleurs, poste membre du domaine, réseau 192.168.10.0 /24)	Windows 11	DHCP

À titre indicatif, prévoyez **2 vCPU, 4 Go de mémoire et 40 Go de disque** pour SRV-OCS (la base grossit avec le nombre de machines et la fréquence des inventaires). Ces valeurs, comme les adresses et les mots de passe d'exemple, sont à adapter à votre environnement.

01

ÉTAPE 1 SUR 6

Préparer le serveur Debian 13 et le DNS

Le serveur reçoit un nom, une adresse fixe, le bon fuseau horaire et les DNS du domaine. La console et l'API seront joignables par **deux noms** différents, `ocs.sio.local` et `ocs-api.sio.local`, qui désignent la même machine : la raison en est expliquée à l'étape 4.

A – Préparer SRV-OCS

1. Mettez le système à jour : `sudo apt update && sudo apt full-upgrade -y`, puis installez les outils utiles aux contrôles : `sudo apt install -y curl wget ca-certificates`
2. Donnez son nom au serveur et déclarez-le dans le fichier `hosts` (nom complet et nom court) :

TERMINAL (SRV-OCS)

```
sudo hostnamectl set-hostname srv-ocs
echo "192.168.20.64  srv-ocs.sio.local  srv-ocs" | sudo tee -a /etc/hosts
sudo timedatectl set-timezone Europe/Paris
```

3. Donnez au serveur une **adresse IP fixe** (adaptez le nom de la carte, ici `ens18`, avec `ip -br address`) : éditez `/etc/network/interfaces`, indiquez les DNS du domaine dans `/etc/resolv.conf`, puis redémarrez le réseau avec `sudo systemctl restart networking` :

/etc/network/interfaces

```
auto ens18
iface ens18 inet static
    address 192.168.20.64/24
    gateway 192.168.20.254
```

/etc/resolv.conf

```
search sio.local
nameserver 192.168.20.250
nameserver 192.168.20.251
```

4 Vérifiez le résultat : nom, adresse IP, résolution DNS des contrôleurs, heure :

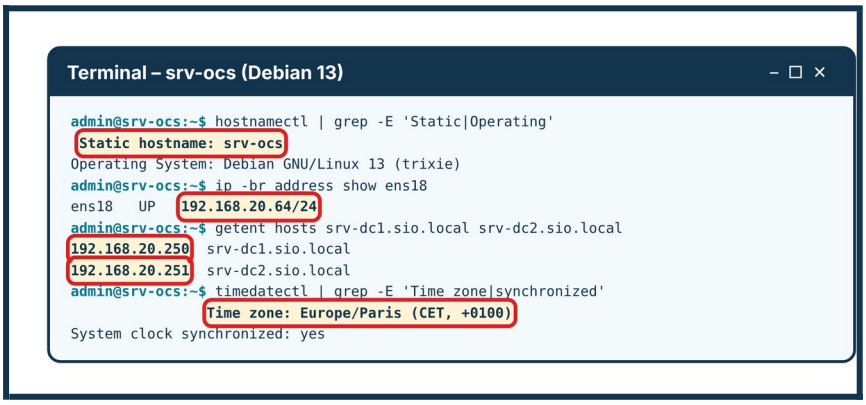


Figure 1 – Contrôle de la préparation de SRV-OCS

À SAVOIR

Si « System clock synchronized » indique « no », installez un service de temps (`sudo apt install -y chrony`). L'heure doit être juste : les certificats ont des dates de validité.

B – Déclarer les deux noms dans le DNS

Sur **SRV-DC1**, dans PowerShell (**administrateur**), créez les deux enregistrements. Ils pointent vers l'adresse de SRV-OCS ; ils sont répliqués sur SRV-DC2 avec la zone :

POWERSHELL (SRV-DC1)

```
Add-DnsServerResourceRecordA -ZoneName sio.local -Name ocs `
    -IPv4Address 192.168.20.64
Add-DnsServerResourceRecordA -ZoneName sio.local -Name ocs-api `
    -IPv4Address 192.168.20.64
Resolve-DnsName ocs.sio.local, ocs-api.sio.local -Type A | Select Name, Type, TTL,
IPAddress
```

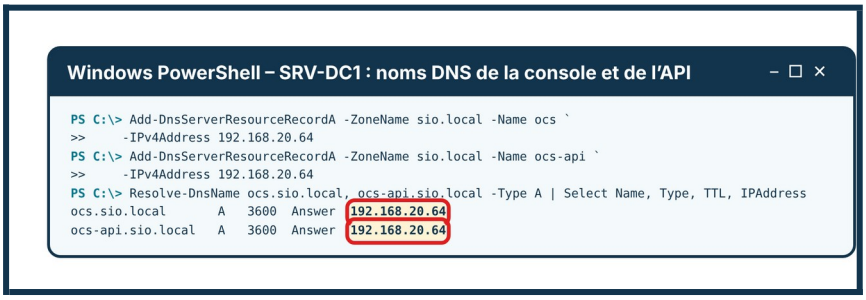


Figure 2 – PowerShell : noms DNS de la console et de l'API

À SAVOIR

Un seul serveur, deux noms : c'est possible parce que nginx choisit le site à servir d'après le nom demandé. Le certificat de l'étape 4 portera ces deux noms.

✔ POINT DE CONTRÔLE

- SRV-OCS répond au nom **srv-ocs** et à l'adresse **192.168.20.64**
- `ocs.sio.local` et `ocs-api.sio.local` sont résolus en **192.168.20.64**
- Le fuseau horaire est **Europe/Paris** et l'horloge est **synchronisée**

02 ÉTAPE 2 SUR 6 Installer PostgreSQL et créer la base

OCS Inventory 3.0 range tout (inventaires, comptes, configuration) dans une **base de données**. Il fonctionne avec **PostgreSQL** ou avec **MariaDB / MySQL** : le script de configuration de l'étape 3 propose les deux. Ce tutoriel retient

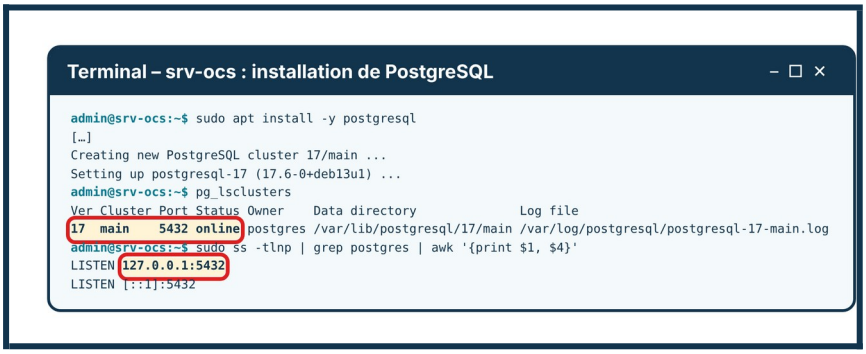
PostgreSQL, fourni par Debian 13 en **version 17**. Le serveur de base ne sera joignable que depuis SRV-OCS (adresse `127.0.0.1`).

A – Installer PostgreSQL

1 Installez le serveur, puis contrôlez que le cluster `main` est en ligne et qu'il n'écoute que sur la boucle locale :

TERMINAL (SRV-OCS)

```
sudo apt install -y postgresql
pg_lsclusters
sudo ss -tlnp | grep postgres | awk '{print $1, $4}'
```



À SAVOIR

L'installation crée automatiquement le cluster `17/main` et le démarre. Le compte système `postgres` est l'administrateur de la base : on s'y connecte avec `sudo -u postgres`.

Figure 3 – PostgreSQL 17 installé et démarré

B – Créer l'utilisateur et la base d'OCS

1 Créez un utilisateur **ocs** protégé par un mot de passe, puis une base **ocs** dont il est le propriétaire :

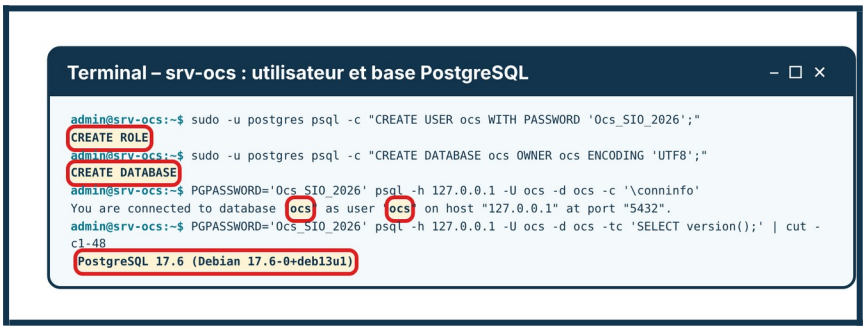
TERMINAL (SRV-OCS)

```
sudo -u postgres psql -c "CREATE USER ocs WITH PASSWORD 'Ocs_SIO_2026';"
sudo -u postgres psql -c "CREATE DATABASE ocs OWNER ocs ENCODING 'UTF8';"
```

2 Vérifiez que l'utilisateur peut se connecter **avec son mot de passe**, comme le fera l'API (connexion réseau sur `127.0.0.1`, pas par le compte système) :

TERMINAL (SRV-OCS)

```
PGPASSWORD='Ocs_SIO_2026' psql -h 127.0.0.1 -U ocs -d ocs -c '\conninfo'
PGPASSWORD='Ocs_SIO_2026' psql -h 127.0.0.1 -U ocs -d ocs -tc 'SELECT version();'
```



À SAVOIR

`CREATE ROLE` et `CREATE DATABASE` confirment les créations. Si la connexion est refusée (« password authentication failed »), reprenez la première commande avec le bon mot de passe : `ALTER USER ocs WITH PASSWORD '…';`.

Figure 4 – Création de l'utilisateur et de la base, test de connexion

REMARQUE IMPORTANTE

`Ocs_SIO_2026` n'est qu'un exemple : **choisissez un mot de passe solide** pour l'utilisateur **ocs** de la base. Vous le saisirez à l'étape 3-C ; il sera alors enregistré dans le fichier de configuration de l'API.

✔ **POINT DE CONTRÔLE**

- `pg_lsclusters` indique le cluster **17 main** en ligne
- La connexion `psql -h 127.0.0.1 -U ocs -d ocs` réussit avec le mot de passe choisi

03 ÉTAPE 3 SUR 6 Installer OCS Inventory 3.0

Le projet publie la version 3.0.0 sous forme de **paquets Debian** (fichiers `.deb`) sur sa page GitHub, sans dépôt apt à déclarer : on télécharge les fichiers puis on les installe avec `apt`, qui se charge des dépendances (nginx, uWSGI, Python, outils de compilation).

Paquet	Contenu
`ocsinventory-backend`	L'**API** : code Python, environnement virtuel, site nginx sur le port 8081, service uWSGI et minuterie des automatisations
`ocsinventory-frontend`	La **console** web : fichiers statiques et site nginx sur le port 80
`ocsinventory-server`	**Méta-paquet** : il n'apporte aucun fichier, il impose l'installation des deux paquets ci-dessus

À SAVOIR

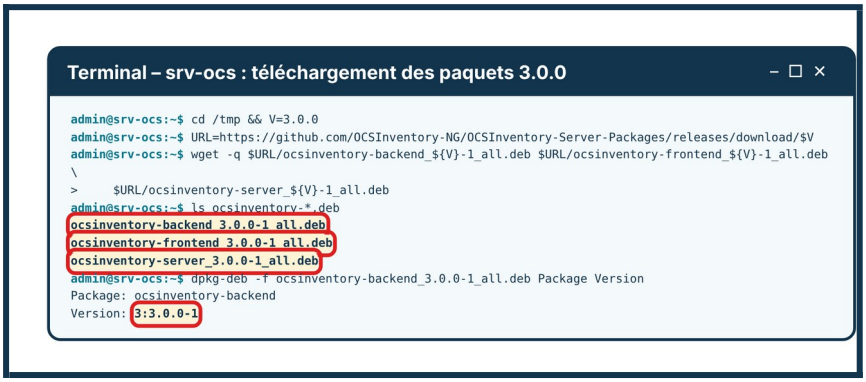
Les paquets de la version 3.0.0 sont publiés **sans signature** (c'est indiqué dans les notes de version). Téléchargez-les uniquement depuis la page officielle du projet, `github.com/OCSInventory-NG/OCSInventory-Server-Packages`, et comparez l'empreinte SHA-256 affichée par GitHub avec celle de vos fichiers (`sha256sum`).

A – Télécharger les paquets

1 Téléchargez les trois fichiers de la version 3.0.0 dans `/tmp` et contrôlez la version du paquet (le « 3: » est un numéro d'époque, il fait partie de la version) :

TERMINAL (SRV-OCS)

```
cd /tmp && V=3.0.0
URL=https://github.com/OCSInventory-NG/OCSInventory-Server-Packages/releases/download/$V
wget -q $URL/ocsinventory-backend_${V}-1_all.deb $URL/ocsinventory-frontend_${V}-
1_all.deb \
    $URL/ocsinventory-server_${V}-1_all.deb
ls ocsinventory-*.deb
dpkg-deb -f ocsinventory-backend_${V}-1_all.deb Package Version
```



À SAVOIR

Si `wget` ne peut pas joindre GitHub, vérifiez la sortie Internet du serveur (passerelle, proxy). Vous pouvez aussi télécharger les fichiers depuis un poste et les copier avec `scp`.

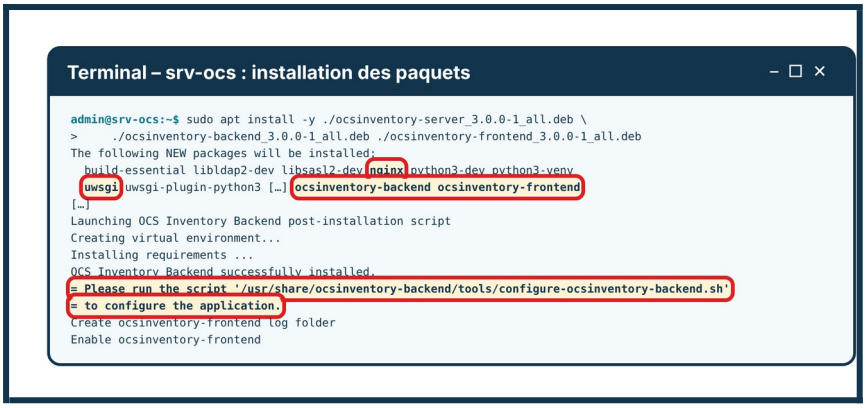
Figure 5 – Téléchargement des trois paquets

B – Installer les paquets

1. Installez les trois fichiers **en une seule commande**. Le préfixe `./` indique à apt qu'il s'agit de fichiers locaux ; il récupère lui-même les dépendances. L'opération dure plusieurs minutes : le paquet de l'API crée un environnement Python et compile certaines bibliothèques :

TERMINAL (SRV-OCS)

```
sudo apt install -y ./ocsinventory-server_3.0.0-1_all.deb \
    ./ocsinventory-backend_3.0.0-1_all.deb ./ocsinventory-frontend_3.0.0-1_all.deb
```



À SAVOIR

Le message encadré de la fin demande de lancer le script de configuration : c'est l'étape suivante. À ce stade, l'API n'a pas encore de base de données : elle ne peut pas fonctionner.

Figure 6 – Installation des paquets : dépendances et message final

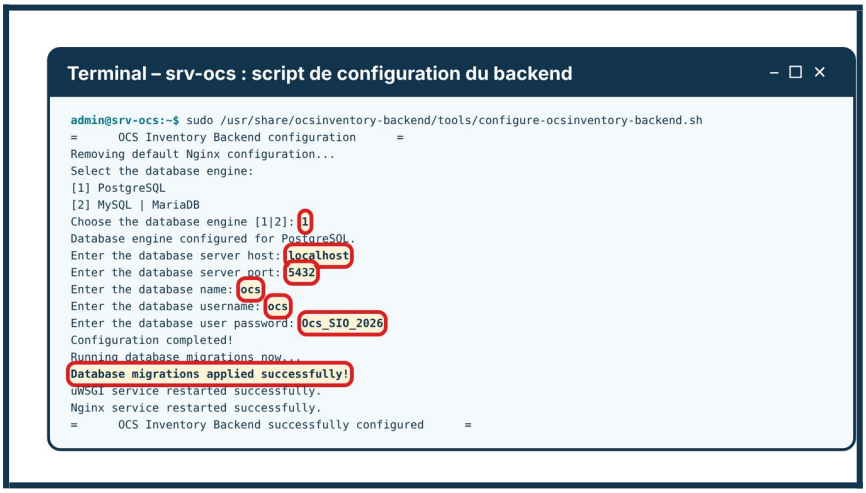
C – Configurer l'API avec le script fourni

Le paquet fournit un script qui renseigne le fichier de configuration de l'API (`/usr/share/ocsinventory-backend/.env`), **crée les tables** de la base, puis redémarre uWSGI et nginx. Il pose six questions : le moteur de base de données, puis l'adresse, le port, le nom, l'utilisateur et le mot de passe de la base :

TERMINAL (SRV-OCS)

```
sudo /usr/share/ocsinventory-backend/tools/configure-ocsinventory-backend.sh
```

Question du script	Réponse
Choose the database engine [1|2]	`1` (PostgreSQL)
Enter the database server host	`localhost`
Enter the database server port	`5432`
Enter the database name	`ocs`
Enter the database username	`ocs`
Enter the database user password	le mot de passe choisi à l'étape 2-B



À SAVOIR

Le script supprime au passage le site nginx par défaut de Debian. **Le mot de passe de la base s'affiche en clair pendant la saisie**, comme l'impose le script : ne lancez pas cette commande devant témoin. En cas d'échec, le détail est dans `/tmp/ocsinventory-backend-configuration.log`.

Figure 7 – Script de configuration de l'API

Les migrations ont créé les tables et les données de départ : les groupes `super-admin`, `admin`, `user`, `AgentInventory`, `AgentDeployment` et `SnmpScanner`, ainsi que le compte **admin** (mot de passe `admin`, à changer à l'étape 5).

D – Générer la clé de chiffrement

OCS **chiffre** dans la base les paramètres sensibles, comme le mot de passe du compte de liaison avec l'annuaire (étape 6). La clé est la variable `FIELD_ENCRYPTION_KEY` du fichier `.env` ; le paquet y laisse une valeur d'exemple, qu'il faut remplacer par une clé personnelle. La commande `encryption_key --generate` en fabrique une nouvelle ; on la place dans le fichier, puis on redémarre uWSGI, qui ne lit `.env` qu'au démarrage :

TERMINAL (SRV-OCS)

```
KEY=$(sudo /usr/lib/ocsinventory-backend/venv/bin/python \
    /usr/share/ocsinventory-backend/manage.py encryption_key --generate)
sudo sed -i "s|^FIELD_ENCRYPTION_KEY=.*|FIELD_ENCRYPTION_KEY='$KEY'|" \
    /usr/share/ocsinventory-backend/.env
sudo grep -c "^FIELD_ENCRYPTION_KEY='" /usr/share/ocsinventory-backend/.env
sudo systemctl restart uwsgi
sudo /usr/lib/ocsinventory-backend/venv/bin/python \
    /usr/share/ocsinventory-backend/manage.py encryption_key
```



Figure 8 – Clé de chiffrement générée et prise en compte

À SAVOIR

Le « 1 » confirme que la ligne existe dans `.env`. La dernière commande n'affiche que l'**empreinte** de la clé en service (jamais la clé) et le nombre de secrets lisibles : 0 pour l'instant, puisqu'aucun secret n'est encore enregistré.

REMARQUE IMPORTANTE

La clé de l'exemple n'est pas la vôtre : n'en recopiez pas la valeur. **Sauvegardez le fichier** `.env` avec la base : sans la même clé, les secrets enregistrés (mot de passe de l'annuaire) deviennent illisibles et doivent être ressaisis.

E – Démarrer les automatisations et tester en local

OCS exécute ses tâches planifiées (règles, nettoyages) grâce à une **minuterie systemd** (`timer`). Le paquet l'active mais ne la démarre pas, la base n'étant pas prête au moment de l'installation. Démarrez-la, puis testez les deux sites en local : l'API sur le port **8081**, la console sur le port **80** :

TERMINAL (SRV-OCS)

```
sudo systemctl enable --now ocsinventory-backend-automation.timer
systemctl is-active uwsgi nginx ocsinventory-backend-automation.timer
sudo ss -tlnp | grep nginx | awk '{print $1, $4}'
curl -s http://127.0.0.1:8081/api-check/
curl -sI http://127.0.0.1/ | head -1
```

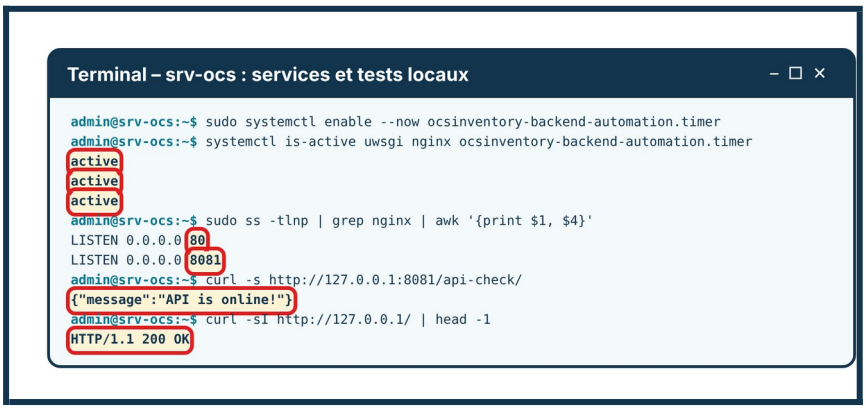


Figure 9 – Services actifs et tests locaux

À SAVOIR

`/api-check/` est une adresse de contrôle de l'API, accessible sans compte. Si elle répond une erreur 502, consultez `sudo tail -n 30 /var/log/uwsgi/ocsinventory-backend.log` : la cause est presque toujours dans `.env` (base ou clé).

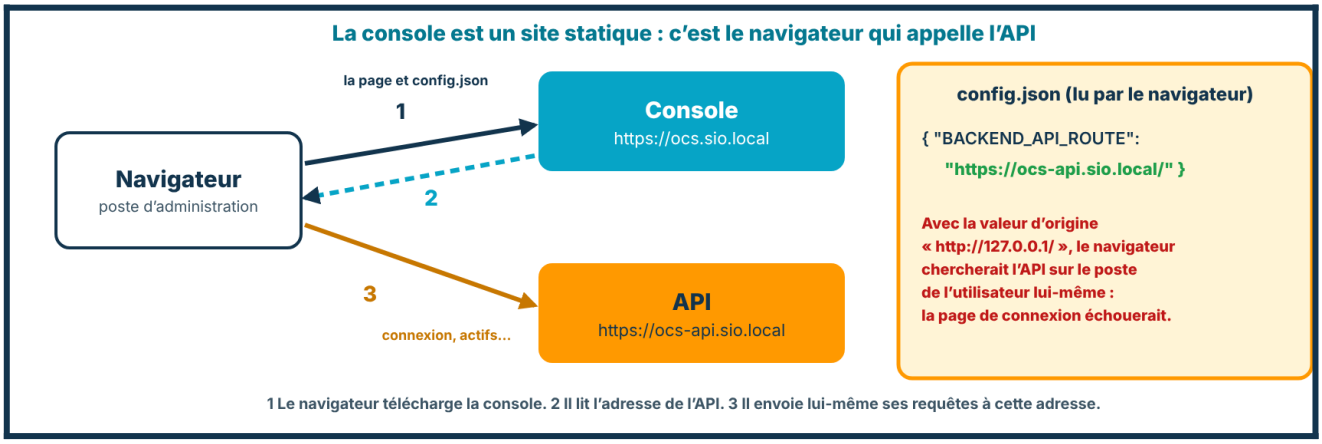
✔ POINT DE CONTRÔLE

- Le script a répondu **Database migrations applied successfully!**
- `encryption_key` affiche une **empreinte** (la clé est valide)
- `curl http://127.0.0.1:8081/api-check/` répond **API is online!** et la console répond **200 OK** sur le port 80

04 ÉTAPE 4 SUR 6
Publier la console en HTTPS avec SIO-CA

Pour l'instant, tout est en HTTP : les mots de passe des administrateurs et des agents circuleraient en clair. Nous plaçons les deux sites derrière **HTTPS**, avec un certificat délivré par **SIO-CA** (auquel les postes du domaine font déjà confiance), directement dans **nginx** : aucun autre composant n'est nécessaire.

Il faut **deux noms** parce que la console et l'API sont deux sites distincts. La console est un site statique : le navigateur la télécharge, lit le fichier `config.json` qui contient l'**adresse de l'API**, puis envoie **lui-même** ses requêtes à cette adresse. Cette adresse doit donc être joignable **depuis le poste de l'administrateur** : la valeur d'origine (`http://127.0.0.1/`) désignerait le poste lui-même, pas le serveur.



A – Créer la clé privée et la demande de certificat

1 Créez le fichier de description `sudo nano /tmp/ocs.cnf` : le certificat doit porter **les deux noms** (champ « Subject Alternative Name »), les navigateurs ignorant le nom principal seul :

/tmp/ocs.cnf

```
[req]
distinguished_name = dn
req_extensions = ext
prompt = no

[dn]
CN = ocs.sio.local

[ext]
subjectAltName = DNS:ocs.sio.local, DNS:ocs-api.sio.local
```

2 Générez la clé privée (RSA 2048) et la demande de certificat, puis contrôlez le contenu de la demande :

TERMINAL (SRV-OCS)

```
sudo mkdir -p /etc/ssl/ocs
sudo openssl req -new -newkey rsa:2048 -nodes -config /tmp/ocs.cnf \
    -keyout /etc/ssl/ocs/ocs.key -out /tmp/ocs.csr
openssl req -in /tmp/ocs.csr -noout -subject -verify
openssl req -in /tmp/ocs.csr -noout -text | grep -A1 Alternative
```



Figure 10 – Clé privée et demande de certificat avec ses deux noms

À SAVOIR

Le fichier `.key` est la clé privée : elle reste sur SRV-OCS, avec des droits restreints (point C). Le fichier `.csr` ne contient que la clé **publique** et les noms : il peut circuler librement.

B – Faire signer la demande par SIO-CA

Sur **SRV-DC1** (qui héberge l'autorité de certification), dans PowerShell **administrateur**, exportez le certificat de l'autorité (utile à l'étape D), récupérez la demande, soumettez-la avec le modèle « **Serveur Web** » (nom interne `WebServer`) puis renvoyez les deux certificats vers SRV-OCS :

POWERSHELL (SRV-DC1)

```
certutil -ca.cert C:\Temp\sio-ca.cer
scp admin@192.168.20.64:/tmp/ocs.csr C:\Temp\
certreq -submit -config "SRV-DC1.sio.local\SIO-CA" `
    -attrib "CertificateTemplate:WebServer" `
    C:\Temp\ocs.csr C:\Temp\ocs.cer
scp C:\Temp\ocs.cer C:\Temp\sio-ca.cer admin@192.168.20.64:/tmp/
```



Figure 11 – Signature de la demande par l'autorité SIO-CA

À SAVOIR

Si `certreq` répond « Refusé » ou « En attente », le modèle « Serveur Web » n'est pas activé sur l'autorité (console certsrv > Modèles de certificats > Nouveau > Modèle de certificat à délivrer) ou le compte n'a pas le droit « Inscrire » dessus.

C – Installer le certificat sur SRV-OCS

1. Convertissez le certificat reçu au format PEM, protégez la clé (lisible par `root` seul : nginx la lit au démarrage), puis vérifiez que le certificat correspond bien à la clé :

TERMINAL (SRV-OCS)

```
sudo openssl x509 -in /tmp/ocs.cer -out /etc/ssl/ocs/ocs.crt
sudo chmod 600 /etc/ssl/ocs/ocs.key
sudo openssl x509 -in /etc/ssl/ocs/ocs.crt -noout -subject -issuer -enddate
sudo openssl x509 -in /etc/ssl/ocs/ocs.crt -noout -ext subjectAltName
cmp <(sudo openssl x509 -in /etc/ssl/ocs/ocs.crt -noout -pubkey) \
    <(sudo openssl pkey -in /etc/ssl/ocs/ocs.key -pubout) \
    && echo "Clé et certificat correspondent"
```

À SAVOIR

Si OpenSSL répond « Could not read certificate » (ou « unable to load »), le fichier reçu est au format binaire : ajoutez l'option `-inform der` à la première commande.



À SAVOIR

Le certificat vaut deux ans environ (durée du modèle). Notez sa date de fin : **un certificat expiré rend la console inaccessible et bloque les agents**. Pour un renouvellement, répétez les points A à C, remplacez les deux fichiers puis rechargez nginx.

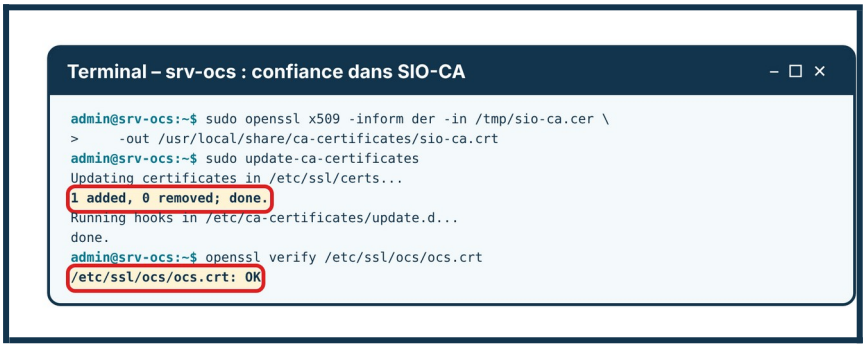
Figure 12 – Certificat installé : émetteur, noms et correspondance avec la clé

D – Faire confiance à SIO-CA sur SRV-OCS

Le serveur lui-même doit reconnaître SIO-CA : l'agent Linux de l'étape 5 et, plus tard, la connexion à l'annuaire (étape 6) vérifient les certificats avec le **magasin de confiance du système**. Convertissez le certificat de l'autorité (exporté au point B, au format binaire) et ajoutez-le au magasin :

TERMINAL (SRV-OCS)

```
sudo openssl x509 -inform der -in /tmp/sio-ca.cer \
    -out /usr/local/share/ca-certificates/sio-ca.crt
sudo update-ca-certificates
openssl verify /etc/ssl/ocs/ocs.crt
```



À SAVOIR

`openssl verify` répond **OK** quand la chaîne est valide : le certificat de SRV-OCS est bien signé par une autorité de confiance. Cette étape est faite **une seule fois** pour ce serveur.

Figure 13 – SIO-CA ajoutée au magasin de confiance

E – Écrire la configuration nginx

Les paquets ont installé deux sites nginx en HTTP (`ocsinventory-backend` sur le port 8081 et `ocsinventory-frontend` sur le port 80). Plutôt que de modifier ces fichiers (une mise à jour pourrait les remplacer), nous créons **un seul fichier personnel** qui reprend leur contenu avec le HTTPS, et nous désactivons les deux d'origine. Créez `sudo nano /etc/nginx/sites-available/ocs.conf`, en trois blocs. Le premier **redirige** le HTTP vers HTTPS :

/etc/nginx/sites-available/ocs.conf (1/3 : redirection)

```
server {
    listen 80 default_server;
    server_name ocs.sio.local ocs-api.sio.local;
    return 301 https://$host$request_uri;
}
```

1 Le deuxième sert la **console** : les fichiers statiques de `/usr/share/ocsinventory-frontend`, avec le certificat. Toute adresse inconnue renvoie vers `index.html` (l'application gère elle-même ses pages) :

/etc/nginx/sites-available/ocs.conf (2/3 : console)

```
server {
    listen 443 ssl;
    http2 on;
    server_name ocs.sio.local;
    ssl_certificate     /etc/ssl/ocs/ocs.crt;
    ssl_certificate_key /etc/ssl/ocs/ocs.key;
    ssl_protocols TLSv1.2 TLSv1.3;
    add_header Strict-Transport-Security "max-age=31536000" always;
    root /usr/share/ocsinventory-frontend;
    index index.html;
    access_log /var/log/ocsinventory-frontend/access.log;
    error_log  /var/log/ocsinventory-frontend/error.log;
    location / { try_files $uri $uri/ /index.html; }
}
```

2 Le troisième sert l'**API** : il transmet les requêtes à uWSGI par son socket. L'option `client_max_body_size` relève la taille maximale d'une requête (1 Mo par défaut), car l'inventaire d'un poste très chargé en logiciels peut la dépasser :

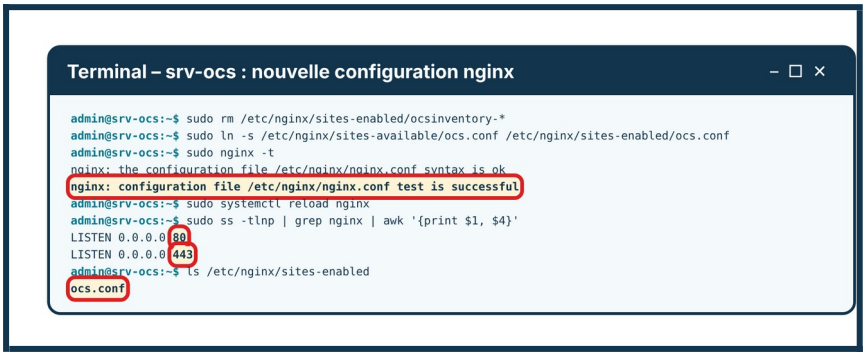
/etc/nginx/sites-available/ocs.conf (3/3 : API)

```
server {
    listen 443 ssl;
    http2 on;
    server_name ocs-api.sio.local;
    ssl_certificate     /etc/ssl/ocs/ocs.crt;
    ssl_certificate_key /etc/ssl/ocs/ocs.key;
    ssl_protocols TLSv1.2 TLSv1.3;
    server_tokens off;
    client_max_body_size 100m;
    location = /favicon.ico { access_log off; log_not_found off; }
    location /static/ { root /usr/share/ocsinventory-backend/; }
    location / {
        include         uwsgi_params;
        uwsgi_pass      unix:/var/run/uwsgi/app/ocsinventory-backend/socket;
        uwsgi_param     UWSGI_SCHEME $scheme;
        uwsgi_param     SERVER_SOFTWARE nginx/$nginx_version;
        uwsgi_param     HTTP_HOST $host;
        uwsgi_param     REQUEST_URI $request_uri;
        uwsgi_param     DOCUMENT_ROOT $document_root;
    }
}
```

3 Désactivez les deux sites d'origine, activez le nouveau, vérifiez la syntaxe et rechargez nginx :

TERMINAL (SRV-OCS)

```
sudo rm /etc/nginx/sites-enabled/ocsinventory-*
sudo ln -s /etc/nginx/sites-available/ocs.conf /etc/nginx/sites-enabled/ocs.conf
sudo nginx -t
sudo systemctl reload nginx
sudo ss -tlnp | grep nginx | awk '{print $1, $4}'
ls /etc/nginx/sites-enabled
```



À SAVOIR

`nginx -t` doit répondre « test is successful » **avant** le rechargement : en cas d'erreur, il indique le fichier et la ligne fautifs. Le port 8081 n'est plus ouvert : l'API n'est désormais joignable qu'en HTTPS.

Figure 14 – Nouvelle configuration nginx

F – Indiquer à la console l'adresse de l'API

1 Éditez `sudo nano /usr/share/ocsinventory-frontend/config/config.json` et remplacez l'adresse d'origine par celle de l'API, **avec la barre oblique finale** (la console y ajoute directement les chemins de l'API) :

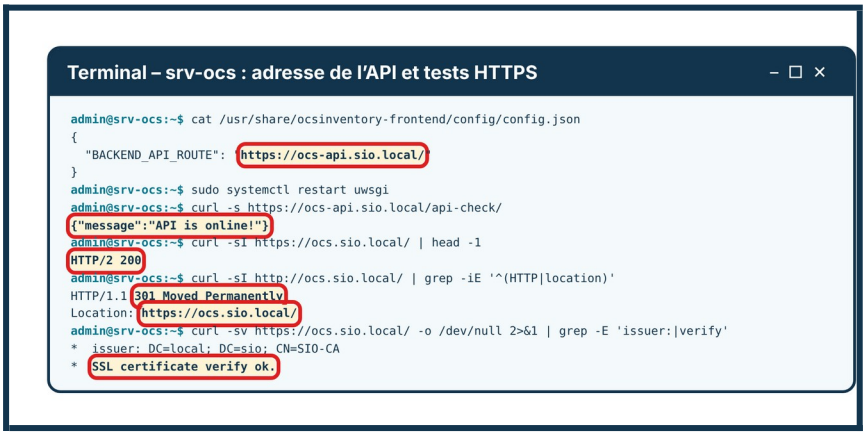
/usr/share/ocsinventory-frontend/config/config.json

```
{
  "BACKEND_API_ROUTE": "https://ocs-api.sio.local/"
}
```

2 Dans le fichier de l'API, la variable `FRONTEND_REDIRECT` (adresse de la console, utilisée par les connexions SSO) vaut encore `http://localhost:3000`. Corrigez-la, puis redémarrez uWSGI et testez la chaîne complète :

TERMINAL (SRV-OCS)

```
sudo sed -i "s|^FRONTEND_REDIRECT=.*|FRONTEND_REDIRECT='https://ocs.sio.local'|" \
    /usr/share/ocsinventory-backend/.env
sudo systemctl restart uwsgi
curl -s https://ocs-api.sio.local/api-check/
curl -sI https://ocs.sio.local/ | head -1
curl -sI http://ocs.sio.local/ | grep -iE '^(HTTP|location)'
curl -sv https://ocs.sio.local/ -o /dev/null 2>&1 | grep -E 'issuer:|verify'
```



À SAVOIR

Le dernier test confirme que le certificat est reconnu **depuis le serveur** (étape D). Le navigateur de l'administrateur, lui, reconnaît SIO-CA parce que son poste est membre du domaine.

Figure 15 – Adresse de l'API et tests HTTPS

G – Ouvrir la console

1. Sur le poste d'administration, ouvrez `https://ocs.sio.local` : la page de connexion s'affiche, sans avertissement de certificat. Si le navigateur a gardé l'ancienne adresse de l'API en mémoire, forcez le rechargement avec **Ctrl + F5** :

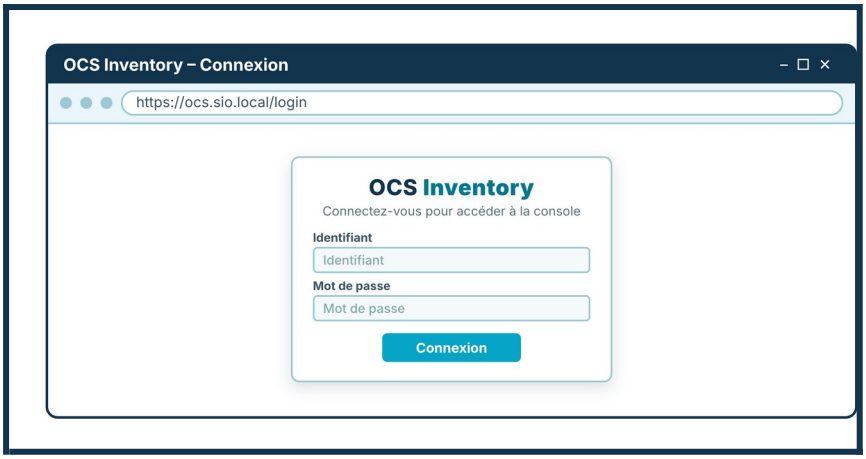


Figure 16 – Page de connexion d'OCS Inventory 3.0

À SAVOIR

Si la page s'affiche mais que la connexion échoue (« Network Error »), ouvrez les outils du navigateur (**F12**, onglet Réseau) : la requête en erreur indique l'adresse appelée. C'est presque toujours `BACKEND_API_ROUTE` ou un certificat non reconnu sur le poste.

✔ **POINT DE CONTRÔLE**

- `https://ocs.sio.local` affiche la page de connexion, **sans avertissement** de certificat
- `curl https://ocs-api.sio.local/api-check/` répond **API is online!** et `http://ocs.sio.local` redirige vers HTTPS
- `/etc/nginx/sites-enabled` ne contient que **ocs.conf**

05 ÉTAPE 5 SUR 6 Premiers réglages : comptes et premier actif

OCS Inventory 3.0 distingue des **utilisateurs**, qui regroupent les administrateurs de la console **et** les comptes de service des agents, et des **groupes**, qui portent les droits. Cette étape sécurise le compte d'origine, crée le compte que les agents utiliseront, puis vérifie que l'ensemble fonctionne en inventoriant le serveur lui-même.

A – Changer le mot de passe du compte admin

1 Connectez-vous avec l'identifiant **admin** et le mot de passe **admin** (valeurs d'origine, créées par les migrations), puis ouvrez le menu « **Gestion des utilisateurs** » > « **Utilisateurs** » :

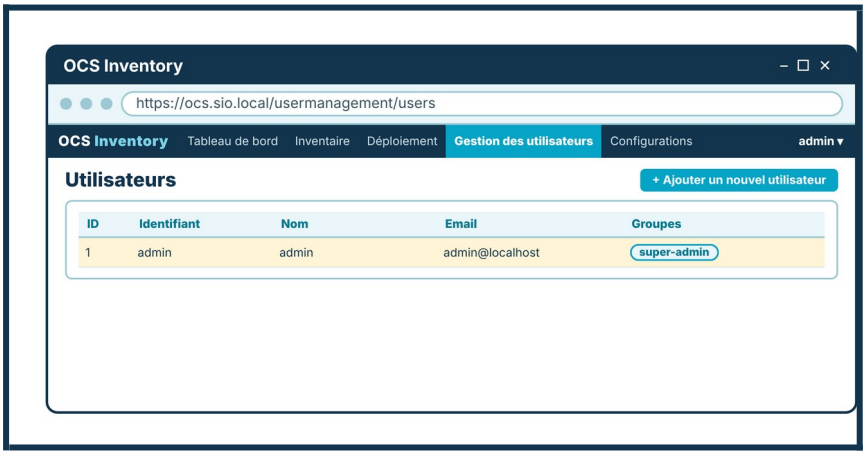


Figure 17 – Gestion des utilisateurs : le compte d'origine

À SAVOIR

Le groupe `super-admin` possède tous les droits (ajout, modification, suppression, consultation). Chaque groupe apparaît sous la forme d'une pastille dans la colonne « Groupes ».

2 Cliquez sur le crayon de la ligne **admin** pour ouvrir « **Modifier un utilisateur** », saisissez un **nouveau mot de passe** et validez avec « **Sauvegarder** » :

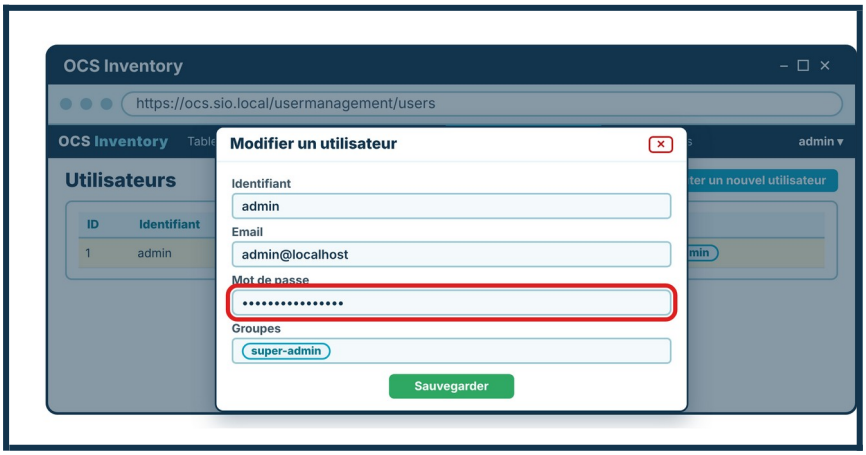


Figure 18 – Modification du mot de passe du compte admin

À SAVOIR

Ce compte est **local** à OCS : il n'existe pas dans l'annuaire AD DS. Il reste utile en secours si l'annuaire est injoignable (étape 6).

REMARQUE IMPORTANTE

Changez ce mot de passe immédiatement : `admin` / `admin` est un couple d'identifiants public, identique sur toutes les installations d'OCS Inventory 3.0.

B – Comprendre les groupes

Les droits ne sont jamais donnés à un utilisateur directement, mais à un **groupe** auquel il appartient. Six groupes existent à l'installation (menu « Gestion des utilisateurs » > « Groupes ») :

Groupe	Droits	Usage
`super-admin`	Tous : ajout, modification, suppression, consultation	Administrateurs de la console
`admin`	Ajout, modification, consultation (pas de suppression)	Techniciens
`user`	Consultation seule	Lecteurs
`AgentInventory`	Envoyer des inventaires, lire sa configuration et les modèles	Compte de service des agents
`AgentDeployment`	Lire les actions de déploiement et en rapporter le résultat	Compte de service des agents (déploiement)
`SnmpScanner`	Droits du scanner SNMP	Compte du scanner SNMP (hors tutoriel)

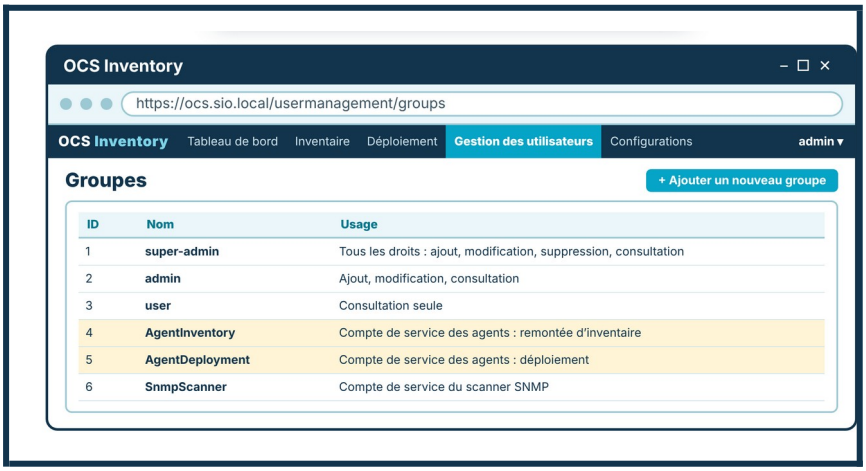


Figure 19 – Les six groupes créés par l'installation

À SAVOIR

Les groupes `AgentInventory` et `AgentDeployment` sont **protégés** : ils servent aux machines, pas aux personnes. Un compte d'agent ne doit appartenir qu'à ces groupes, jamais à `admin`.

C – Créer le compte de service des agents

Tous les agents du parc utiliseront le **même compte**, dont l'identifiant et le mot de passe seront écrits dans leur configuration. Ce compte ne doit donc avoir que le droit d'envoyer des inventaires. Dans « Utilisateurs », cliquez sur « **+ Ajouter un nouvel utilisateur** », puis renseignez l'identifiant **agent-inventory**, un e-mail, un mot de passe et le groupe **AgentInventory** :



Figure 20 – Création du compte de service des agents

À SAVOIR

Choisissez un mot de passe **solide et propre à ce compte** : il sera lisible dans la configuration de chaque agent, et dans le script de déploiement du tutoriel suivant. L'exemple de la suite utilise `Agent_SIO_2026!`.

D – Inventorier le serveur avec l'agent Linux

Pour vérifier que toute la chaîne fonctionne (HTTPS, API, base, droits), installons l'agent sur SRV-OCS lui-même. Le projet publie une archive pour les systèmes Debian ; son script `install.sh` accepte le mode **silencieux** (`-S`), où l'on indique l'adresse de l'API, le compte et son mot de passe :

TERMINAL (SRV-OCS)

```
mkdir /tmp/agent && cd /tmp/agent
A=OCSInventory-Agent-Setup-Linux-Deb-3.0.0.tar.xz
wget -q https://github.com/OCSInventory-NG/OCSInventory-Agent-Rework/releases/download/
3.0.0/$A
tar -xJf $A --strip-components=1 && ls
sudo ./install.sh -S -U https://ocs-api.sio.local -u agent-inventory \
    -p 'Agent_SIO_2026!' -n -s
systemctl is-active ocsinventory-agent
```

Option	Rôle
`-S`	Mode silencieux : aucune question, `-U`, `-u` et `-p` sont alors obligatoires
`-U https://ocs-api.sio.local`	Adresse de l'**API** (pas celle de la console), **sans barre oblique finale**
`-u` / `-p`	Compte de service de l'agent (étape C)
`-n`	Lance un inventaire **immédiatement** après l'installation
`-s`	Enregistre l'agent comme **service systemd** (démarrage automatique, envois réguliers)



Figure 21 – Installation de l'agent Linux en mode silencieux

À SAVOIR

Le script n'ajoute aucune option de certificat : l'agent utilise le magasin de confiance du système, où SIO-CA a été ajoutée à l'étape 4-D. Son journal est `/var/log/ocsinventory-agent.log` : c'est la première chose à lire en cas de problème.

E – Vérifier l'inventaire dans la console

1 Menu « **Inventaire** » > « **Actifs** » : le serveur apparaît avec son système d'exploitation et la date de sa dernière mise à jour (comptez une minute après le lancement de l'agent, puis rafraîchissez la page) :



Figure 22 – Liste des actifs : srv-ocs

À SAVOIR

« Actif » est le terme d'OCS 3.0 pour une machine inventoriée. Cliquez sur son nom pour consulter le détail : matériel, logiciels, réseau.

2 Le « **Tableau de bord** » compte désormais un actif, contacté aujourd'hui :

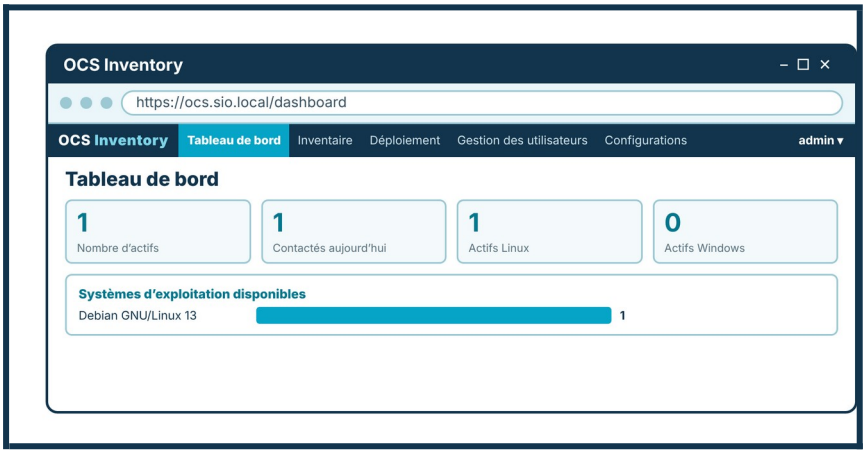


Figure 23 – Tableau de bord après le premier inventaire

À SAVOIR

Le tableau de bord se personnalise (déplacement et ajout de graphiques). Il se remplira à mesure que les postes Windows remonteront leur inventaire.

✔ POINT DE CONTRÔLE

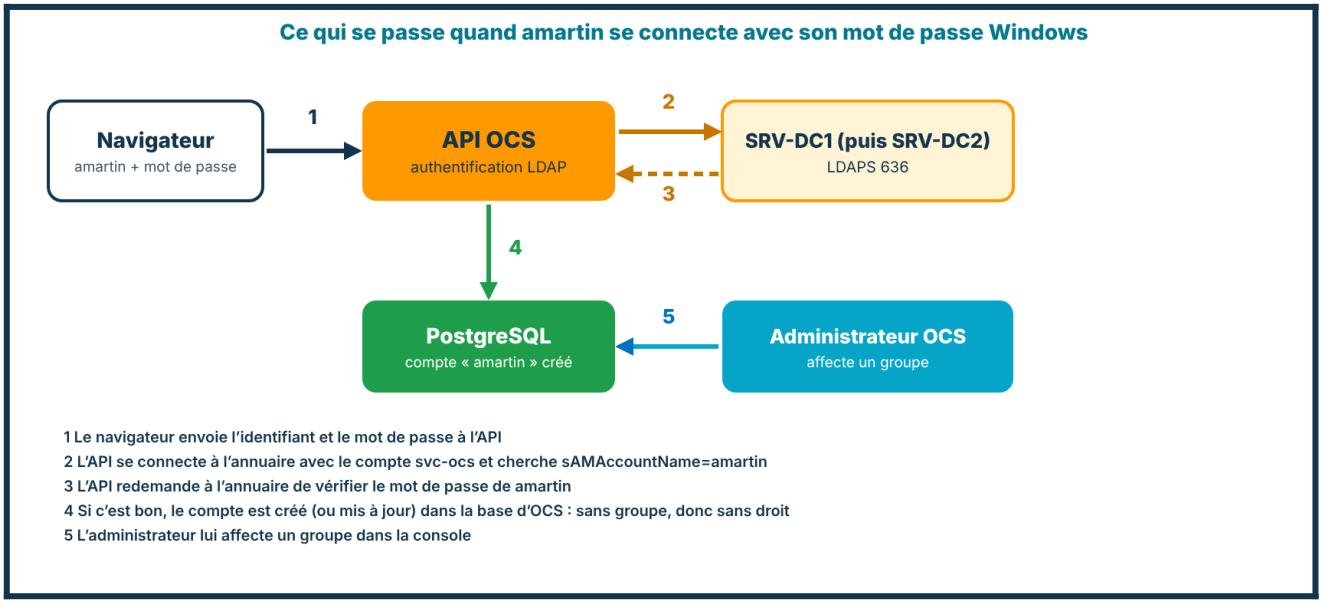
- Le mot de passe de **admin** n'est plus `admin`
- Le compte **agent-inventory** existe et appartient **uniquement** au groupe `AgentInventory`
- Le serveur **srv-ocs** figure dans « Inventaire > Actifs » avec Debian GNU/Linux 13

06

ÉTAPE 6 SUR 6

Lier OCS à l'annuaire AD DS

Jusqu'ici, les administrateurs de la console ont des comptes propres à OCS. Avec la liaison à l'annuaire, ils se connectent avec leur **mot de passe Windows** : un seul mot de passe à retenir, et la désactivation d'un compte dans Active Directory lui retire l'accès. Voici ce qui se passe à chaque connexion :



Deux points sont propres à OCS. D'une part, le compte est **créé à la première connexion réussie**, mais **sans groupe**, donc sans aucun droit : un administrateur doit lui en affecter un (point F). D'autre part, OCS essaie les annuaires configurés **dans l'ordre de leur priorité** : en déclarant SRV-DC1 puis SRV-DC2, on obtient la **tolérance de panne**.

A – Préparer AD DS : le compte de service

OCS se connecte à l'annuaire avec un **compte de liaison** (`BIND_DN`), qui lui permet de rechercher l'utilisateur avant de vérifier son mot de passe. Ce compte n'a besoin d'aucun droit particulier : tout utilisateur du domaine peut lire l'annuaire. Sur **SRV-DC1**, dans PowerShell (**administrateur**), créez-le dans l'unité d'organisation des comptes de service :

POWERSHELL (SRV-DC1)

```
New-ADUser -Name "svc-ocs" -SamAccountName "svc-ocs" `
    -UserPrincipalName "svc-ocs@sio.local" `
    -Path "OU=Comptes de service,DC=sio,DC=local" `
    -AccountPassword (Read-Host -AsSecureString "Mot de passe") `
    -PasswordNeverExpires $true -CannotChangePassword $true -Enabled $true
Get-ADUser svc-ocs | Select DistinguishedName, Enabled
```



Figure 24 – Création du compte de service svc-ocs

À SAVOIR

Le **nom distinctif** (DN) affiché est celui que vous saisirez dans OCS. Si l'unité d'organisation `Comptes de service` n'existe pas (voir les tutoriels GLPI et Nextcloud), créez-la d'abord : `New-ADOrganizationalUnit -Name "Comptes de service" -Path "DC=sio,DC=local"`.

REMARQUE IMPORTANTE

Le mot de passe est saisi à l'invite (il ne reste pas dans l'historique). **Choisissez-le solide** : il sera enregistré, chiffré, dans la base d'OCS. `PasswordNeverExpires` évite qu'une expiration ne coupe silencieusement les connexions.

B – Tester LDAPS depuis SRV-OCS

Avant de configurer OCS, vérifiez en ligne de commande que SRV-OCS joint l'annuaire en LDAPS, fait confiance à son certificat, et que le compte de service retrouve un utilisateur. SIO-CA est déjà dans le magasin de confiance (étape 4-D) :

TERMINAL (SRV-OCS)

```
sudo apt install -y ldap-utils
openssl s_client -connect srv-dc1.sio.local:636 -verify_return_error \
    </dev/null 2>&1 | grep -E 'subject=|Verify return code'
ldapsearch -H ldaps://srv-dc1.sio.local \
    -D "CN=svc-ocs,OU=Comptes de service,DC=sio,DC=local" -W \
    -b "OU=Utilisateurs,DC=sio,DC=local" "(sAMAccountName=amartin)" dn mail
```

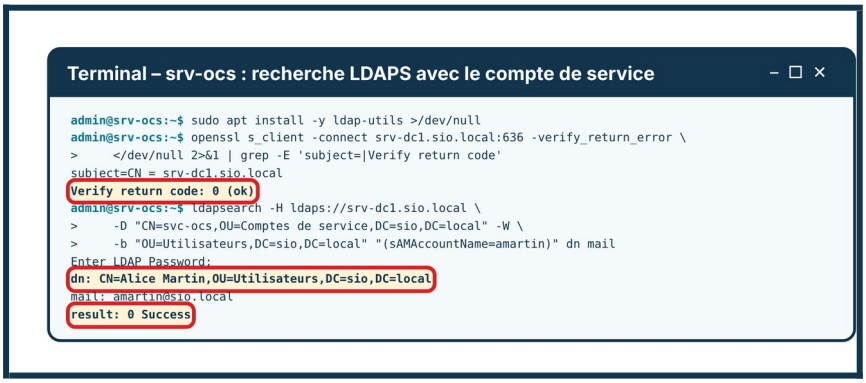


Figure 25 – Test de la connexion LDAPS et de la recherche d'un utilisateur

À SAVOIR

Utilisez le **nom complet** du contrôleur (`srv-dc1.sio.local`), pas son adresse IP : le certificat est établi au nom du serveur. « Verify return code: 0 (ok) » prouve que la chaîne de confiance est complète. Si ce test échoue, OCS échouera aussi : reprenez l'étape 4-D.

C – Activer l'authentification LDAP dans OCS

1 Connectez-vous en **admin**, ouvrez « **Configurations** » > « **Authentification** » : l'onglet **Global** liste les méthodes (LOCAL, LDAP, OIDC, CAS), chacune avec un interrupteur. Activez **LDAP** ; l'authentification **locale** reste activée, ce qui conserve le compte `admin` en secours :

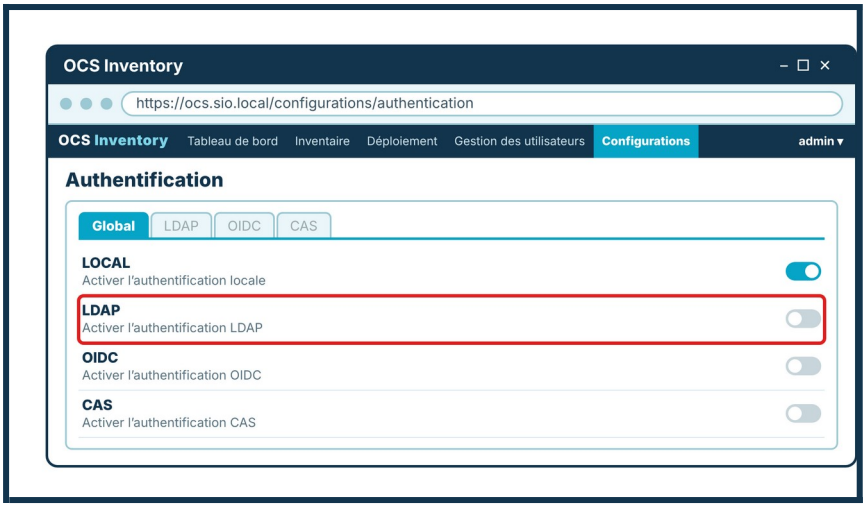


Figure 26 – Authentification : activation de la méthode LDAP

À SAVOIR

Une fois LDAP activé, l'onglet **LDAP** (grisé jusque-là) devient accessible. Les méthodes OIDC et CAS (authentification unique) sont hors de ce tutoriel.

D – Configurer l'annuaire SRV-DC1, puis SRV-DC2

Dans l'onglet **LDAP**, OCS propose déjà une configuration d'exemple, « **LDAP Default Config** » (adresse `ldap://127.0.0.1:3268`, valeurs fictives) : modifiez-la avec le **crayon** de sa ligne (« Éditer la connexion LDAP ») et renseignez les champs suivants :

Champ	Valeur	Rôle
Nom	`SIO-DC1`	Nom affiché dans la liste
SERVER_URI	`ldaps://srv-dc1.sio.local:636`	Annuaire **chiffré** (LDAPS), nom complet du contrôleur
BIND_DN	`CN=svc-ocs,OU=Comptes de service,DC=sio,DC=local`	Compte de liaison (étape A)
BIND_PASSWORD	le mot de passe de svc-ocs	Enregistré **chiffré** avec la clé de l'étape 3-D
BASE_DN	`OU=Utilisateurs,DC=sio,DC=local`	Où chercher les utilisateurs (sous-arbre compris)
USER_LOGIN_FIELD	`sAMAccountName`	Attribut comparé à l'identifiant saisi (amartin)
PROTOCOL_VERSION	`v3`	Version du protocole LDAP

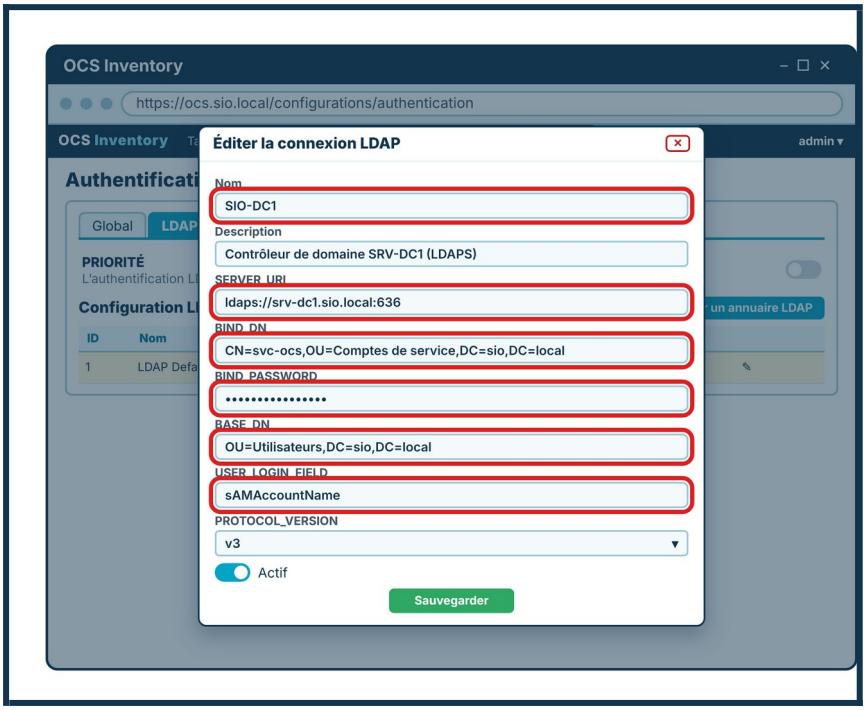


Figure 27 – Édition de la configuration LDAP de SRV-DC1

À SAVOIR

Laissez la case « **Actif** » cochée et validez avec « **Sauvegarder** ». Limitez `BASE_DN` à l'unité d'organisation des utilisateurs : on évite ainsi que des comptes d'autres unités (comptes de service, par exemple) puissent se connecter.

1 Cliquez ensuite sur « **+ Ajouter un annuaire LDAP** » et renseignez les mêmes champs pour le second contrôleur (nom **SIO-DC2**, `SERVER_URI` = `ldaps://srv-dc2.sio.local:636`). OCS interrogera SIO-DC1, puis SIO-DC2 si le premier ne répond pas :

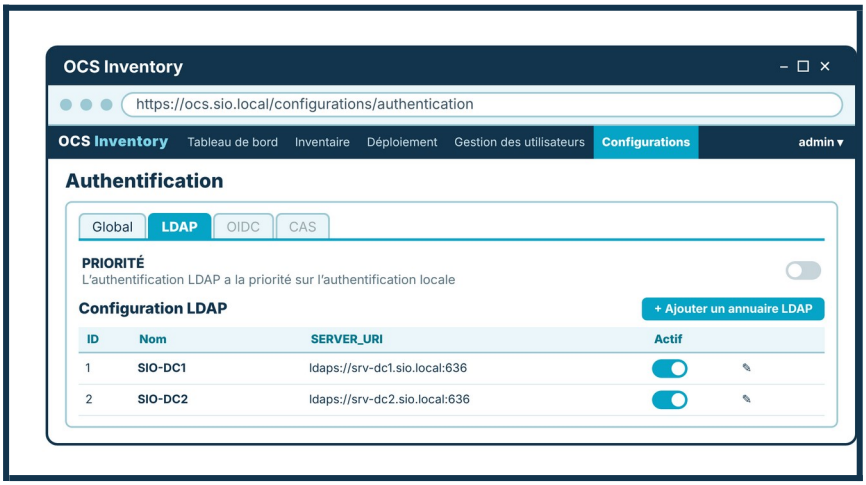


Figure 28 – Deux annuaires LDAP configurés

À SAVOIR

L'interrupteur « **PRIORITÉ** » du haut sert à tester les annuaires **avant** la base locale. Laissez-le désactivé : le compte `admin` est vérifié d'abord en local, sans interroger l'annuaire.

E – Récupérer le nom et l'adresse e-mail

Sans **mappage** des attributs, un compte venant de l'annuaire est créé avec son seul identifiant. Cliquez sur l'icône « **Configurer le mappage** » de la ligne de l'annuaire (une fois pour chaque annuaire) et indiquez l'attribut de l'annuaire qui alimente chaque champ OCS :

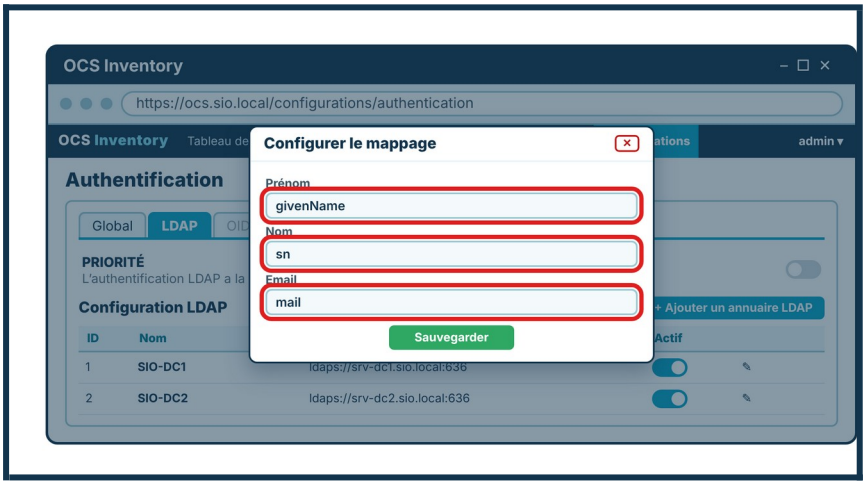


Figure 29 – Mappage des attributs de l'annuaire

À SAVOIR

`givenName` (prénom), `sn` (nom de famille) et `mail` (adresse e-mail) sont les noms standard d'Active Directory. Les valeurs sont recopiées à chaque connexion de l'utilisateur.

F – Se connecter avec un compte Windows et lui donner un groupe

1. Ouvrez une **fenêtre de navigation privée** sur `https://ocs.sio.local` et connectez-vous avec l'identifiant **amartin** et son mot de passe de domaine. La connexion crée le compte dans OCS, sans groupe : le tableau de bord ne montre rien d'utile, c'est normal
2. Revenez à la session **admin**, ouvrez « Gestion des utilisateurs » > « Utilisateurs » : le compte **amartin** est apparu, avec son nom et son e-mail (grâce au mappage) et la mention « aucun groupe » :

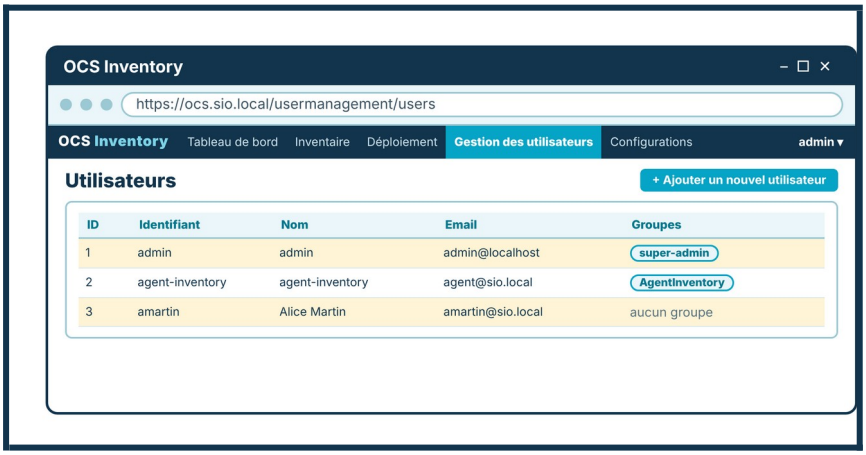


Figure 30 – Utilisateurs : le compte amartin créé à la première connexion

À SAVOIR

Les comptes de l'annuaire sont créés **à la demande** : un utilisateur qui ne s'est jamais connecté n'apparaît pas dans la liste. Les mots de passe ne sont pas copiés dans OCS : ils sont vérifiés par l'annuaire à chaque connexion.

3 Cliquez sur le crayon de la ligne **amartin**, choisissez le groupe voulu dans « **Groupes** » (ici `admin`, ou `user` pour une consultation seule) et sauvegardez :

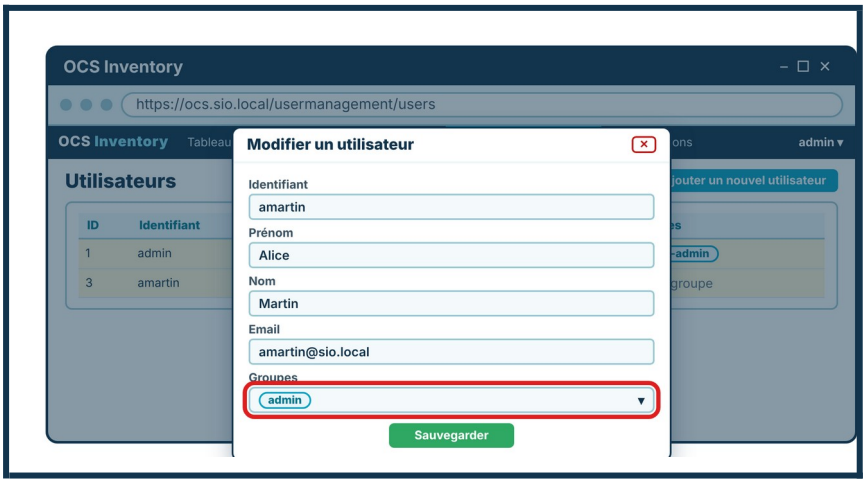


Figure 31 – Affectation d'un groupe à l'utilisateur amartin

À SAVOIR

Le groupe donne les droits : pour un technicien qui doit pouvoir modifier mais pas supprimer, `admin` convient ; pour un simple lecteur, `user`. Préférez des comptes **nominatifs** plutôt qu'un compte partagé.

4 **amartin** se déconnecte puis se reconnecte : il accède maintenant aux données selon son groupe :

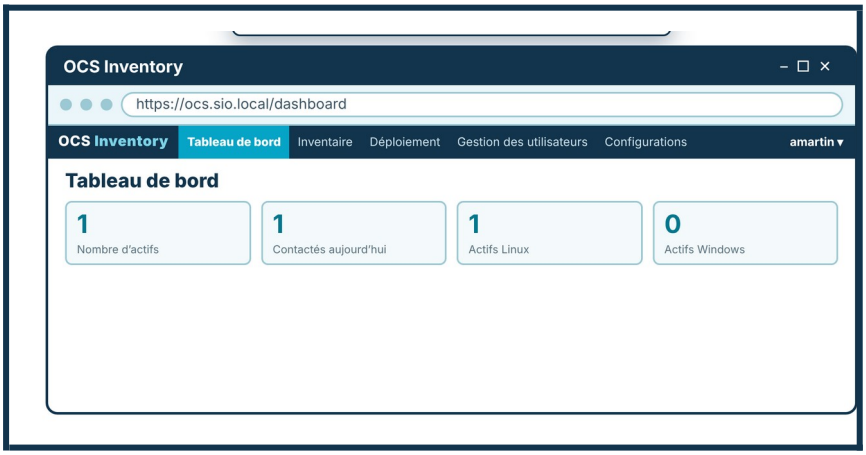


Figure 32 – Tableau de bord ouvert par amartin

À SAVOIR

Pour tester la tolérance de panne, arrêtez SRV-DC1 : la connexion d'amartin doit continuer de fonctionner grâce à SIO-DC2. Désactivez ensuite le compte dans Active Directory : sa connexion à OCS est refusée.

✔ POINT DE CONTRÔLE

- `ldapsearch` renvoie le **dn** d'amartin et `openssl s_client` indique **Verify return code: 0 (ok)**
- L'onglet LDAP de la configuration liste **SIO-DC1** et **SIO-DC2**, tous deux **actifs**
- **amartin** ouvre une session OCS avec son **mot de passe Windows** et dispose des droits de son groupe
- Le compte local **admin** fonctionne toujours (secours si l'annuaire est injoignable)

Votre serveur **OCS Inventory 3.0** est **opérationnel** : console et API en HTTPS avec le certificat de l'autorité interne, base PostgreSQL, un compte de service pour les agents et des administrateurs authentifiés par l'annuaire. Il reste à **déployer l'agent sur les postes Windows** (tutoriel « Déployer l'agent OCS Inventory sur les postes Windows par GPO ») et à prévoir les sauvegardes de la base et du fichier `.env`.

Dépannage : les symptômes les plus fréquents

Symptôme	Cause probable	Solution
Le script : « Error during database migrations »	Mot de passe ou nom de base erroné, PostgreSQL arrêté	Lire `/tmp/ocsinventory-backend-configuration.log` ; tester `psql -h 127.0.0.1 -U ocs -d ocs` (étape 2-B) ; relancer le script
Page de connexion affichée mais « Network Error »	`BACKEND_API_ROUTE` encore à `127.0.0.1`, sans barre finale, ou nom de l'API non résolu depuis le poste	Corriger `config.json` (étape 4-F), **Ctrl + F5**, relire la requête en erreur (F12 > Réseau) ; `nslookup ocs-api.sio.local`
Avertissement de certificat, ou « Network Error » alors que `curl` fonctionne	Poste hors domaine (SIO-CA inconnue) ou nom absent des noms alternatifs	Importer SIO-CA sur le poste ; contrôler `openssl x509 -noout -ext subjectAltName`
« 502 Bad Gateway » sur l'API	uWSGI arrêté ou erreur dans `.env` (base, clé)	`sudo systemctl status uwsgi` ; `sudo tail -n 30 /var/log/uwsgi/ocsinventory-backend.log`
`nginx -t` échoue, ou le port 80 est déjà utilisé	Un site d'origine est encore activé	`ls /etc/nginx/sites-enabled` ne doit montrer que `ocs.conf` ; supprimer les liens `ocsinventory-*`
« 413 Request Entity Too Large » dans le journal de l'agent	Inventaire plus gros que la limite nginx	Augmenter `client_max_body_size` dans le bloc de l'API, `nginx -t` puis `reload`
Un agent ne remonte pas d'inventaire	SIO-CA non reconnue, adresse avec barre finale, mauvais compte ou mot de passe	Lire `/var/log/ocsinventory-agent.log` ; adresse **sans** barre finale ; `curl https://ocs-api.sio.local/api-check/` depuis la machine
L'agent reçoit un refus (401 ou 403)	Le compte n'est pas dans le groupe `AgentInventory`	Vérifier le groupe du compte (étape 5-C)
Après la liaison LDAP, l'onglet LDAP est grisé	Méthode LDAP non activée	Configurations > Authentification > Global : activer LDAP
La connexion d'un utilisateur de l'annuaire échoue	CA non reconnue, `BIND_DN` ou mot de passe faux, utilisateur hors de `BASE_DN`	Refaire le test de l'étape 6-B ; `sudo systemctl restart uwsgi` ; lire le journal d'uWSGI
amartin se connecte mais ne voit rien ou reçoit des erreurs	Compte sans groupe, donc sans droit	L'administrateur lui affecte un groupe (étape 6-F)
Mot de passe de l'annuaire illisible après une restauration	Le fichier `.env` restauré contient une autre `FIELD_ENCRYPTION_KEY`	Restaurer la clé d'origine, ou ressaisir le mot de passe dans la configuration LDAP
Plus de HTTPS après une mise à jour des paquets	Le paquet de la console a recréé son site d'origine	Supprimer à nouveau `sites-enabled/ocsinventory-frontend` puis `nginx -t` et `reload`

Pense-bête : commandes et emplacements

Commande / emplacement	Utilité
`sudo systemctl status uwsgi nginx`	État de l'API et du serveur web (`restart uwsgi` après toute modification de `.env`)
`/usr/share/ocsinventory-backend/.env`	Configuration de l'API : base, clé de chiffrement, adresse de la console. **À sauvegarder**
`/usr/share/ocsinventory-frontend/config/config.json`	Adresse de l'API appelée par la console
`/etc/nginx/sites-available/ocs.conf`	Sites HTTPS (console et API) et certificat
`/var/log/uwsgi/ocsinventory-backend.log`	Journal de l'API (erreurs de base, de droits, d'annuaire)
`/var/log/ocsinventory-frontend/error.log`	Journal du site de la console
`systemctl list-timers 'ocsinventory*'`	Minuterie des automatisations d'OCS
`sudo -u postgres pg_dump ocs > ocs.sql`	Sauvegarde de la base (à faire avec `.env` et les certificats)
`curl https://ocs-api.sio.local/api-check/`	Contrôle rapide de l'API (réponse « API is online! »)
`/var/log/ocsinventory-agent.log`	Journal de l'agent Linux installé avec `install.sh`
`openssl x509 -in /etc/ssl/ocs/ocs.crt -noout -enddate`	Date d'expiration du certificat : à surveiller, renouvellement en suivant l'étape 4 (A à C)