Intégration du Scanner Epson ET4856

Scanner Epson ET-4856 — Intégration Home Assistant & Paperless-ngx

---

1. Vue d'ensemble

L'imprimante Epson ET-4856 (identifiée comme ET-4850 Series) est pilotée en local via le protocole **eSCL** (AirScan/AirPrint) en HTTPS. Un script bash exécuté depuis le container Home Assistant déclenche les scans et dépose les fichiers directement dans le répertoire `consume` de Paperless-ngx pour ingestion automatique (OCR + indexation).

Architecture

```
Bouton Dashboard HA → script.scanner_epson → shell_command.epson_scan
    → POST eSCL /ScanJobs (HTTPS:443) sur l'imprimante
    → GET /NextDocument → fichier PDF
    → /media/scans/ (= consume Paperless-ngx)
    → Paperless ingère → OCR → indexation → archivage
```

Composants impliqués

Composant	Rôle	Emplacement
Epson ET-4856	Scanner (eSCL via HTTPS)	192.168.10.25:443
Home Assistant	Orchestration, dashboard, déclenchement	Container Docker sur Unraid
Paperless-ngx	Ingestion, OCR, indexation	Container Docker sur Unraid
epson_scan.sh	Script bash de scan eSCL	/config/scripts/epson_scan.sh

---

2. Protocole eSCL — Détails techniques

Découverte

Le scanner n'expose **pas** eSCL en HTTP (port 80). Il faut utiliser **HTTPS sur le port 443** avec `-k` (certificat auto-signé SEIKO EPSON CORP.).

```
# Vérifier le statut du scanner
curl -s -k https://192.168.10.25/eSCL/ScannerStatus

# Voir les capabilities
curl -s -k https://192.168.10.25/eSCL/ScannerCapabilities
```

Capabilities du scanner

Paramètre	Vitre (Platen)	Chargeur (ADF)
Résolutions	100, 200, 300, 600, 1200 DPI	100, 200, 300, 600 DPI
Modes couleur	RGB24, Grayscale8, BlackAndWhite1	RGB24, Grayscale8, BlackAndWhite1
Formats sortie	PDF, JPEG	PDF, JPEG
Duplex	N/A	**Non supporté** (SimplexInputCaps uniquement)
Capacité	1 page	50 feuilles max
Dimensions max	2550 x 3510 (≈A4)	2550 x 4200

Workflow eSCL (séquence HTTP)

1. `POST /eSCL/ScanJobs` avec XML de paramètres → Réponse 201 + header `Location` contenant l'URL du job
2. `GET {Location}/NextDocument` → Télécharge le fichier scanné (PDF ou JPEG)
3. `GET {Location}/NextDocument` (2ème appel) → Renvoie 404, signale la fin au scanner
4. `DELETE {Location}` → Libère le job et remet le scanner en Idle

⚠ **Important :** Sans les étapes 3 et 4, le scanner reste bloqué en "Scan en cours" sur l'écran LCD.

Exemple de XML ScanSettings

```
<?xml version="1.0" encoding="UTF-8"?>
<scan:ScanSettings xmlns:pwg="http://www.pwg.org/schemas/2010/12/sm"
  xmlns:scan="http://schemas.hp.com/imaging/escl/2011/05/03">
  <pwg:Version>2.0</pwg:Version>
  <pwg:ScanRegions>
    <pwg:ScanRegion>
      <pwg:Width>2550</pwg:Width>
      <pwg:Height>3510</pwg:Height>
      <pwg:ContentRegionUnits>escl:ThreeHundredthsOfInches</pwg:ContentRegionUnits>
      <pwg:XOffset>0</pwg:XOffset>
      <pwg:YOffset>0</pwg:YOffset>
    </pwg:ScanRegion>
  </pwg:ScanRegions>
  <pwg:InputSource>Platen</pwg:InputSource>
  <scan:ColorMode>RGB24</scan:ColorMode>
  <scan:XResolution>300</scan:XResolution>
  <scan:YResolution>300</scan:YResolution>
  <pwg:DocumentFormat>application/pdf</pwg:DocumentFormat>
  <scan:Intent>Document</scan:Intent>
</scan:ScanSettings>
```

---

3. Configuration du container Home Assistant

Volume monté

Le répertoire `consume` de Paperless est monté dans le container HA :

```
-v '/mnt/user/Sauvegarde_Jean-Michel/Documents_Importants/consume/':'/media/scans':'rw'
```

Dépendance : poppler-utils

Le package `poppler-utils` (fournit `pdfunite`) est nécessaire pour la fusion PDF multi-pages ADF. Comme le container HA est basé sur Alpine Linux (BusyBox), le package est perdu à chaque redémarrage.

Solution : Un script d'installation + une automation HA au démarrage.

Fichier : `/config/scripts/install_deps.sh`

```
#!/bin/bash
apk add --no-cache poppler-utils 2>/dev/null
```

Automation : dans `automations.yaml`

```
- alias: "Install scanner dependencies"
  trigger:
    - trigger: homeassistant
      event: start
  action:
    - action: shell_command.install_deps
```

---

4. Script epson_scan.sh (v2)

Emplacement

`/config/scripts/epson_scan.sh` (dans le container HA)
`/mnt/user/appdata/homeassistant/scripts/epson_scan.sh` (côté Unraid)

Paramètres

Position	Paramètre	Valeurs possibles	Défaut
$1	Résolution	100, 200, 300, 600	300
$2	Source	Platen, Feeder	Platen
$3	Mode couleur	RGB24, Grayscale8, BlackAndWhite1	RGB24
$4	Format	pdf, jpeg	pdf
$5	Intent	Document, Photo, TextAndGraphic	Document
$6	Nom document	Texte libre (optionnel)	scan_[date]

Comportement

- **Mode Vitre (Platen)** : Scanne 1 page, enregistre directement dans `/media/scans/`

- **Mode ADF (Feeder)** : Boucle de scan page par page jusqu'à ce que le chargeur soit vide, puis fusionne tous les PDF en un seul via `pdfunite`. Maximum 50 pages par sécurité.

Codes de retour

- **Succès :** `OK:/chemin/fichier.pdf:taille:N pages` (exit 0)
- **Erreur :** `ERREUR: description` (exit 1)

Compatibilité BusyBox

Le container HA utilise Alpine Linux avec BusyBox. Le script évite :

- `grep -P` (Perl regex) → remplacé par `sed -n 's/.../p'`
- `stat` → remplacé par `wc -c`
- heredoc `cat << EOF` pour le XML → chaîne dans une variable

Test en ligne de commande

```
# Scan vitre, 200 DPI, couleur, PDF
docker exec homeassistant bash /config/scripts/epson_scan.sh 200 Platen RGB24 pdf Document

# Scan ADF multi-pages, 300 DPI, niveaux de gris, PDF
docker exec homeassistant bash /config/scripts/epson_scan.sh 300 Feeder Grayscale8 pdf
Document
```

---

5. Configuration Home Assistant

configuration.yaml — Sections ajoutées

Input selects (options de scan)

```
input_select:
  scan_source:
    name: Source de scan
    options:
      - "Vitre"
      - "Chargeur (ADF)"
    initial: "Vitre"
```

```
    icon: mdi:scanner
  scan_resolution:
    name: Résolution
    options:
      - "100"
      - "200"
      - "300"
      - "600"
    initial: "300"
    icon: mdi:resize
  scan_color:
    name: Mode couleur
    options:
      - "Couleur"
      - "Niveaux de gris"
      - "Noir et blanc"
    initial: "Couleur"
    icon: mdi:palette
  scan_format:
    name: Format de sortie
    options:
      - "PDF"
      - "JPEG"
    initial: "PDF"
    icon: mdi:file-document
```

Input boolean (indicateur scan en cours)

```
input_boolean:
  scan_en_cours:
    name: Scan en cours
    icon: mdi:progress-clock
```

Shell commands

```
shell_command:
  epson_scan: "bash /config/scripts/epson_scan.sh {{ states('input_select.scan_resolution') }}
{{ 'Feeder' if states('input_select.scan_source') == 'Chargeur (ADF)' else 'Platen' }} {{
{'Couleur':'RGB24','Niveaux de gris':'Grayscale8','Noir et
blanc':'BlackAndWhite1'}[states('input_select.scan_color')] }} {{
```

```
states('input_select.scan_format') | lower }} Document"
  install_deps: "bash /config/scripts/install_deps.sh"
```

Command line sensors (statut scanner)

```
command_line:
  - sensor:
      name: "Epson Scanner Status"
      unique_id: epson_et4856_scanner_status
      icon: mdi:scanner
      command: "curl -s -k https://192.168.10.25/eSCL/ScannerStatus 2>/dev/null"
      scan_interval: 30
      value_template: >
        {% set matches = value | regex_findall('<pwg:State>(.*?)</pwg:State>') %}
        {% if matches %}
          {% set map = {'Idle':'Prêt','Processing':'En
cours','Testing':'Test','Stopped':'Arrêté'} %}
          {{ map.get(matches[0], matches[0]) }}
        {% else %}
          Indisponible
        {% endif %}
  - sensor:
      name: "Epson ADF Status"
      unique_id: epson_et4856_adf_status
      icon: mdi:tray-full
      command: "curl -s -k https://192.168.10.25/eSCL/ScannerStatus 2>/dev/null"
      scan_interval: 30
      value_template: >
        {% set matches = value | regex_findall('<scan:AdfState>(.*?)</scan:AdfState>') %}
        {% if matches %}
          {% set map = {'ScannerAdfEmpty':'Vide','ScannerAdfLoaded':'Papier
chargé','ScannerAdfJam':'Bourrage'} %}
          {{ map.get(matches[0], matches[0]) }}
        {% else %}
          Indisponible
        {% endif %}
```

Note : On utilise `command_line` sensor au lieu de `rest` sensor car le client HTTP Python de HA gère mal le certificat auto-signé de l'imprimante malgré `verify_ssl: false`.

scripts.yaml — Script scanner

```
scanner_epson:
  alias: "Scanner un document"
  icon: mdi:scanner
  mode: single
  sequence:
    - action: input_boolean.turn_on
      target:
        entity_id: input_boolean.scan_en_cours
    - action: shell_command.epson_scan
      response_variable: scan_result
    - action: input_boolean.turn_off
      target:
        entity_id: input_boolean.scan_en_cours
    - choose:
        - conditions:
            - condition: template
              value_template: "{{ scan_result.returncode == 0 }}"
          sequence:
            - action: persistent_notification.create
              data:
                title: "Scan terminé"
                message: >
                  Document scanné avec succès !
                  {{ scan_result.stdout }}
                notification_id: scan_result
        - conditions:
            - condition: template
              value_template: "{{ scan_result.returncode != 0 }}"
          sequence:
            - action: persistent_notification.create
              data:
                title: "Erreur de scan"
                message: >
                  Erreur lors du scan :
                  {{ scan_result.stderr }}
                notification_id: scan_result
```

6. Dashboard

L'onglet Scan se trouve dans le dashboard `lovelace.dashboard_roborock`. Il utilise les cards Mushroom (mushroom-template-card, mushroom-select-card) et state-switch pour la cohérence visuelle avec le reste du dashboard.

Entités utilisées dans le dashboard

Entité	Rôle
sensor.epson_scanner_status	État du scanner (Prêt / En cours)
sensor.epson_adf_status	État du chargeur ADF (Vide / Papier chargé)
input_select.scan_source	Choix Vitre / Chargeur ADF
input_select.scan_resolution	Choix résolution DPI
input_select.scan_color	Choix couleur / gris / N&B
input_select.scan_format	Choix PDF / JPEG
input_boolean.scan_en_cours	Indicateur scan en cours
script.scanner_epson	Script déclenché par le bouton

---

7. Dépannage

Le scanner répond HTTP 503

Un job précédent est resté bloqué. Vérifier le statut et attendre que le scanner passe en `Idle`, ou éteindre/rallumer l'imprimante :

```
docker exec homeassistant curl -s -k https://192.168.10.25/eSCL/ScannerStatus
```

Le scanner reste bloqué sur "Scan en cours" (écran LCD)

Le job n'a pas été fermé proprement. Le script v2 inclut la fermeture (GET NextDocument + DELETE), mais si un scan a été interrompu :

```
# Récupérer l'UUID du job en cours
docker exec homeassistant curl -s -k https://192.168.10.25/eSCL/ScannerStatus

# Supprimer le job manuellement (remplacer UUID)
docker exec homeassistant curl -s -k -X DELETE "https://192.168.10.25/eSCL/ScanJobs/UUID-DU-JOB"
```

pdfunite not found

Le package `poppler-utils` n'a pas été installé au démarrage. Installer manuellement :

```
docker exec homeassistant apk add --no-cache poppler-utils
```

Vérifier que l'automation "Install scanner dependencies" est active et sans erreur.

grep: unrecognized option: P

Le script utilise `grep -P` (Perl regex) incompatible avec BusyBox. Vérifier que le script v2 est bien déployé (utilise `sed` à la place).

```
docker exec homeassistant head -5 /config/scripts/epson_scan.sh
# Doit afficher "v2"
```

Sensor scanner "Indisponible" ou "unavailable"

Vérifier que curl fonctionne depuis le container :

```
docker exec homeassistant curl -s -k https://192.168.10.25/eSCL/ScannerStatus
```

Si ça répond du XML mais le sensor ne fonctionne pas, vérifier les logs :

```
docker exec homeassistant grep -i "command_line\|epson" /config/home-assistant.log | tail -10
```

Timeout sur scan ADF multi-pages

Le `shell_command` de HA a un timeout de **60 secondes**. En 300 DPI, chaque page prend ~10-15 secondes. Au-delà de ~4 pages, le timeout peut être atteint. Solutions :

- Réduire la résolution (200 DPI)
- Passer en mode asynchrone (script en background + notification webhook)

8. Fichiers — Récapitulatif

Fichier (Unraid)	Fichier (container HA)	Rôle
/mnt/user/appdata/homeassistant/scripts/epson_scan.sh	/config/scripts/epson_scan.sh	Script de scan eSCL v2
/mnt/user/appdata/homeassistant/scripts/install_deps.sh	/config/scripts/install_deps.sh	Installation poppler-utils au démarrage
/mnt/user/appdata/homeassistant/configuration.yaml	/config/configuration.yaml	Config HA (input_select, shell_command, sensors)
/mnt/user/appdata/homeassistant/scripts.yaml	/config/scripts.yaml	Script HA scanner_epson
/mnt/user/appdata/homeassistant/automations.yaml	/config/automations.yaml	Automation install_deps au démarrage
/mnt/user/Sauvegarde_Jean-Michel/Documents_Importants/consume/	/media/scans/	Répertoire de sortie (= consume Paperless)

9. Améliorations possibles

- **Trigger Telegram** : Déclencher un scan via commande Telegram bot
- **Bouton Zigbee physique** : Bouton à côté de l'imprimante pour scanner en un appui
- **Nommage depuis le dashboard** : Ajouter un champ input_text éditable (non résolu — incompatibilité avec les vues sections)
- **Mode asynchrone** : Pour les gros lots ADF, lancer le script en background et notifier via webhook quand c'est terminé
- **Niveaux d'encre** : Intégration HACS `ha-epson-workforce` — vérifier `http://192.168.10.25/PRESENTATION/HTML/TOP/PRTINFO.HTML`
- **Duplex manuel** : Workflow en 2 passes (recto puis verso) avec interleaving des pages