03 管理者

備份與還原

> 版本：v1.2 | 更新日期：2026-06-16

本指南說明如何備份、還原和遷移 Config Server 的設備資料，適用於本地端（`SERVER_ROLE=local`）和雲端（`SERVER_ROLE=cloud`）兩種角色。

01

1. 資料結構總覽

Config Server 的資料儲存在 `config-server/data/` 目錄下，採用**檔案系統**作為持久層（無外部資料庫）。

```
config-server/data/
├── licenses.json          ← 授權記錄（Cloud-only）
├── <device-id>/           ← 每台裝置一個子目錄
│   ├── config.json        ← 設備配置快照（心跳上傳）
│   ├── web/               ← Web UI 檔案（index.html.gz）
│   └── firmware/           ← OTA 韌體暫存
```

關鍵檔案說明

檔案	用途	來源
`config.json`	設備的完整配置（網路、IO、MQTT、規則等）	裝置心跳自動上傳
`licenses.json`	所有裝置的授權啟用記錄	Cloud 授權 API
`web/`	（舊）設備 Web UI 上傳快取	**已停用** —— `upload_web.py` 已廢除，Web 資產改隨 `.mesb` 一檔投遞（見 `guide-ota-embedded-web.md`）
`firmware/`	設備 OTA 韌體暫存	Dashboard 上傳

02

2. 備份

2.1 完整備份（推薦）

備份整個 `data/` 目錄即可涵蓋所有設備資料：

```bash
# 建立帶時間戳的備份
cd config-server
tar -czf backup-$(date +%Y%m%d-%H%M%S).tar.gz data/
```

2.2 單一設備備份

若只需備份特定設備：

```bash
# 替換 <DEVICE_ID> 為實際設備 ID
tar -czf backup-<DEVICE_ID>.tar.gz data/<DEVICE_ID>/
```

2.3 透過 API 匯出設備配置

```bash
# 從 Config Server 匯出設備當前配置
curl -s http://localhost:8888/api/devices/<DEVICE_ID>/config | jq . > device_bacl
```

2.4 建議的備份頻率

環境	頻率	方式
開發/測試	手動	功能開發前後各一次
生產(本地)	每日	cron + tar
生產(雲端)	每小時	cron + rsync 或雲端快照

03

3. 還原

3.1 完整還原

```bash
cd config-server

# 停止伺服器
# pm2 stop config-server 或 Ctrl+C

# 還原
tar -xzf backup-20260324-120000.tar.gz

# 重新啟動
node server.js
# 或 pm2 restart config-server
```

3.2 單一設備還原

```bash
tar -xzf backup-<DEVICE_ID>.tar.gz -C config-server/
```

3.3 透過 API 匯入設備配置

```bash
curl -s -X POST http://localhost:8888/api/devices/<DEVICE_ID>/config \
  -H "Content-Type: application/json" \
  -d @device_backup.json
```

04

4. 遷移

4.1 本地端 → 本地端（換機器）

1. **停止**舊機器上的 Config Server。
2. **複製**整個 `config-server/` 目錄到新機器：

```bash
rsync -avz config-server/ new-host:/path/to/config-server/
```

3. 在新機器上安裝依賴並啟動：

```bash
cd config-server
npm install
SERVER_ROLE=local PORT=8888 node server.js
```

4. **更新裝置**的 Config Server URL（透過 Web UI「設定 → 伺服器」卡片）。

4.2 本地端 → 雲端

1. **備份** `data/` 目錄。
2. 將 `data/` 複製到雲端伺服器。
3. 設定環境變數：

```bash
export SERVER_ROLE=cloud
export PORT=3001
export FIREBASE_SERVICE_ACCOUNT=/path/to/firebase-key.json
```

4. 啟動伺服器。雲端角色會額外載入 `license` 模組。
5. 裝置端需更新 Config Server URL 為雲端地址。

4.3 雲端 → 本地端

1. **匯出** `data/` 和 `licenses.json`。
2. 複製到本地機器。
3. 使用 `SERVER_ROLE=local` 啟動。
    - **注意**：`licenses.json` 在本地模式下不會被使用，但保留以備未來遷回雲端。

05

5. 注意事項

5.1 裝置端配置同步

> [!IMPORTANT] Config Server 儲存的是裝置**上傳的快照**，而非「唯一真實來源」。裝置的 QSPI Flash（`/fs/config.json`）才是運行中的主配置。若兩者不一致，以裝置端為準。

5.2 授權資料（Cloud-only）

`licenses.json` 包含所有裝置的授權狀態記錄。遷移時**必須**一併複製此檔案，否則已啟用的裝置會在雲端顯示為未授權。

5.3 韌體檔案

`config-server/firmware/` 目錄儲存透過 Dashboard 上傳的韌體版本。這些檔案較大（約 500KB-1MB），備份時請確認磁碟空間充足。

5.4 敏感資訊

`.env` 檔案包含 Firebase 金鑰等敏感資訊，**不應**納入備份壓縮檔中。建議單獨管理：

```bash
# 備份時排除 .env
tar -czf backup.tar.gz --exclude='.env' data/
```

06

6. 自動備份腳本範例

```bash
#!/bin/bash
# config-server-backup.sh
# 每日執行：crontab -e → 0 2 * * * /path/to/config-server-backup.sh

BACKUP_DIR="/backups/config-server"
DATA_DIR="/path/to/config-server/data"
TIMESTAMP=$(date +%Y%m%d-%H%M%S)

mkdir -p "$BACKUP_DIR"
tar -czf "$BACKUP_DIR/backup-$TIMESTAMP.tar.gz" -C "$(dirname $DATA_DIR)" data/

# 保留最近 30 天的備份
find "$BACKUP_DIR" -name "backup-*.tar.gz" -mtime +30 -delete

echo "[$(date)] 備份完成: backup-$TIMESTAMP.tar.gz"
```

07

7. 雙路徑備援：device push 到雲端＋地端兩個 cs（v5.9.114+）

每台 device 同時支援 push config 到：

- **Cloud cs**（內建 `https://opta.smms.com.tw`，總是 active，無法關閉）
- **Local cs**（`config.configServer.url` 有設才 active）

兩條路徑彼此獨立，任何一條成功都會更新 `lastPushAt`。Cloud 斷線時 local 仍累積 snapshot 歷史，反之亦然。

設定 device 指向 local cs

device 端 admin token 起 curl：

```bash
DEV=192.168.0.31; AUTH="Authorization: Bearer <device-admin-token>"
curl -X POST -H "$AUTH" -H "Content-Type: application/json" \
  -d '{"url":"http://192.168.0.120:8888","token":"admin-token","enabled":true,"au
  http://$DEV/api/configserver
```

之後 device 改 config / 手動 `POST /api/configserver/push` / 心跳 autoPush 都會同時打兩邊。

觀察 push 結果（fw v5.9.119+）

```bash
curl http://$DEV/api/configserver | jq .push
```

回傳 `push.cloud` 跟 `push.local` 各自的 `{result, httpCode, ageS, failCount, cooldownMsLeft}`。連續失敗 N 次（PUSH_FAIL_THRESHOLD）會進 5 min cooldown，cooldown 期間 skip 該路徑，另一條仍正常推。

注意

- Local cs 不在線時 device 仍會嘗試 push、進 cooldown — 這是「fail-fast」設計，不會 hang 主迴圈。
- Local cs 啟動後不會自動被 device 發現，必須由人工 `POST /api/configserver` 設 URL。
- 把 `enabled=false` 或 `url=""` 即可關閉 local push，cloud push 不受影響。

08

8. PG 備份監控（v2.2.6 cs+）

`opta-backup` 容器每日 `pg_dump`，預設 `BACKUP_INTERVAL_SEC=86400` (24h)、`BACKUP_KEEP_DAYS=14`。

不必 SSH 就能查健康狀態：

```bash
curl -H "Authorization: Bearer <admin-jwt>" \
  https://opta.smms.com.tw/api/admin/backup-status
```

回傳：

```json
{
  "mounted": true,
  "dumpDir": "/var/backups/pg",
  "dumpCount": 11,
  "lastDumpFile": "optacs-20260519T065122Z.dump",
  "lastDumpAt": "2026-05-19T06:51:23.139Z",
  "hoursSinceLastDump": 22.7,
  "oldestDumpAt": "2026-05-09T06:51:20.218Z",
  "totalSize": 228239,
  "healthy": true
}
```

`healthy: false` 表示最後一次 dump 超過 30 小時 — 該檢查 backup container 是否還活著（`docker ps opta-backup`）。