

Docker Hermes Container 系統架構圖解

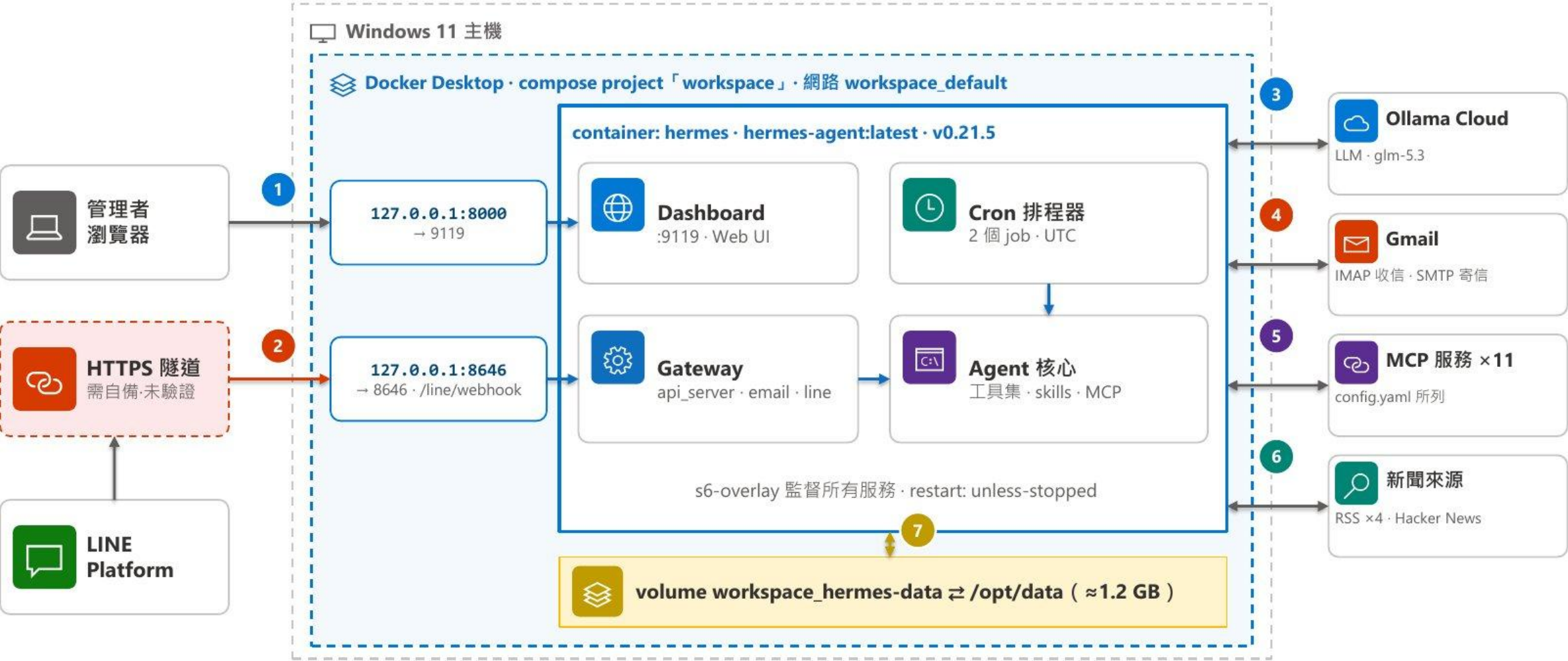
容器運行架構、通道、排程與維運 — 以 Microsoft Azure
架構圖風格繪製

nousresearch/hermes-agent:latest · Hermes Agent v0.21.5
盤點日期 2026-10-02 (依 docker-compose.yml 與運行中容器實際檢視)



整體架構：一個容器，三個通道，一個資料 volume

Windows 11 主機 → Docker Desktop → hermes 容器；外部服務透過本機埠或出站連線互動



① 管理者瀏覽器 → Dashboard ② LINE webhook 經隧道 → Gateway ③ LLM 推論 ④ Email 收寄 ⑤ MCP 工具 ⑥ 新聞抓取 ⑦ 狀態與排程寫入 volume

部署層：docker-compose.yml 一份檔案定義整個運行環境

單一服務 hermes；映像、啟動指令、埠、volume 與環境變數皆在此宣告

docker-compose.yml

services:

hermes:

image: nousresearch/hermes-agent:latest

container_name: hermes

restart: unless-stopped

command: gateway run

ports:

- "127.0.0.1:8000:9119"

- "127.0.0.1:8646:8646" # LINE webhook

volumes:

- hermes-data:/opt/data

environment:

- HERMES_DASHBOARD=1

- HERMES_DASHBOARD_BASIC_AUTH_USERNAME=\${...}

- HERMES_DASHBOARD_BASIC_AUTH_PASSWORD=\${...}

volumes:

hermes-data:

\${...} 為省略；實際值由同目錄 .env 注入，.env 已列入 .gitignore



image

官方映像 hermes-agent:latest · 運行版本 v0.21.5 (Python 3.13 venv 位於 /opt/hermes)



restart

unless-stopped：Docker Desktop 啟動後自動拉起；排程只在容器運行時才會觸發



command

gateway run：啟動 Messaging Gateway，含 cron 排程器、email、LINE 與 API server



ports

兩個埠都只綁 127.0.0.1，僅本機可連；對外需另行建立 HTTPS 隧道



volumes

命名 volume 掛到 /opt/data (HERMES_HOME)，容器重建後設定與排程仍在

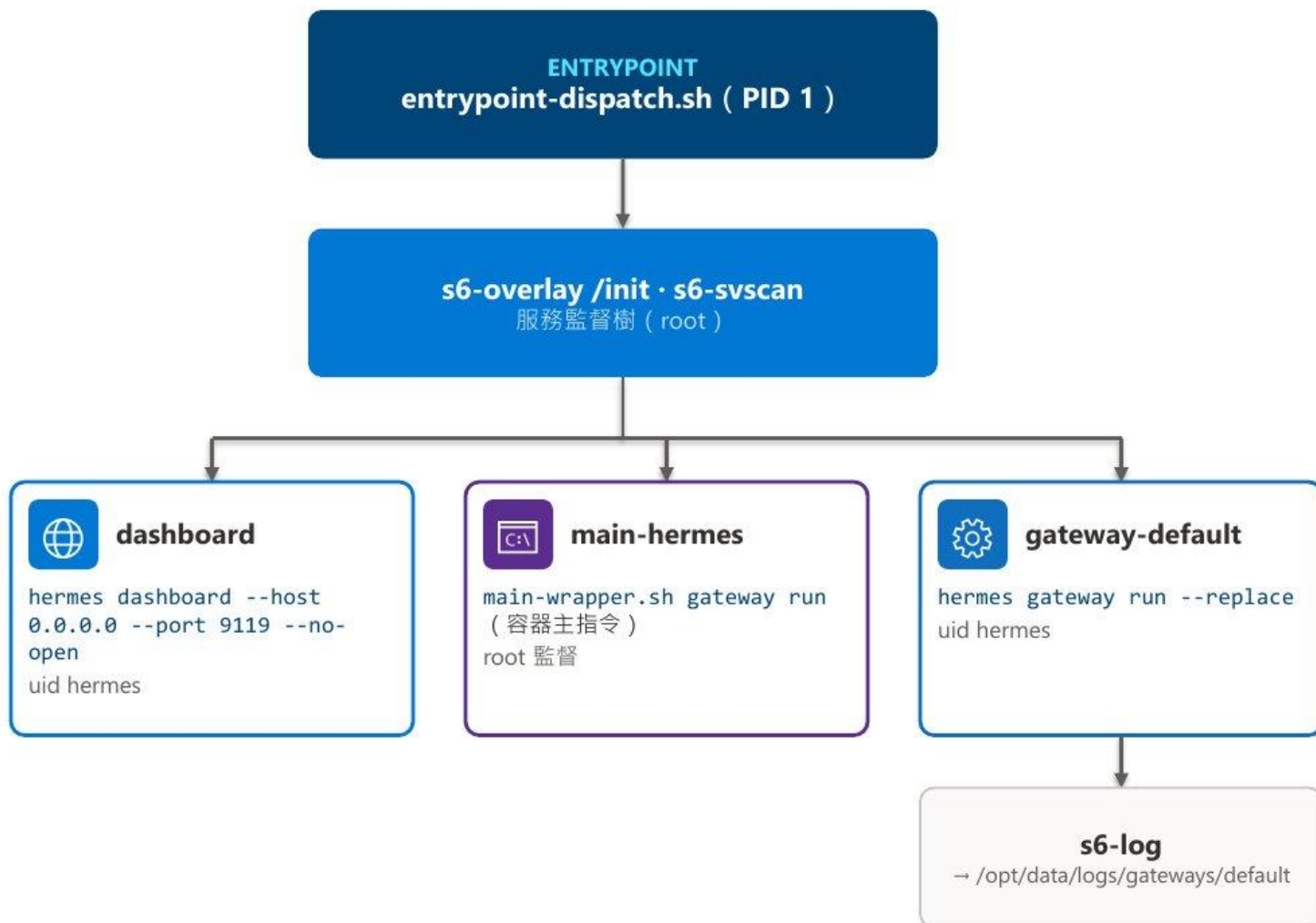


environment

開啟 Dashboard 並以 Basic Auth 保護；帳密不寫死在 compose

容器內部：s6-overlay 監督的程序樹

PID 1 交給 s6-overlay ; Dashboard 與 Gateway 是兩個獨立受監督的長駐服務



1 單容器、雙服務

同一個容器長駐 Dashboard (Web UI) 與 Gateway (通道 + 排程)，由 s6 一併監督

2 崩潰自動重啟

s6-supervise 重新拉起失效服務；compose 的 restart: unless-stopped 再提供第二層保護

3 最小權限

服務程序以 hermes 使用者執行 (ps 檢視)，root 只負責監督與初始化

4 非 PID 1 備援

若由外部 init 啟動 (如 docker run --init)，改走直接啟動模式，不含監督樹

網路與連接埠：只有本機回環入口，其餘皆為出站連線

hermes 單獨接在 workspace_default，與同機其他專案容器的網路彼此隔離

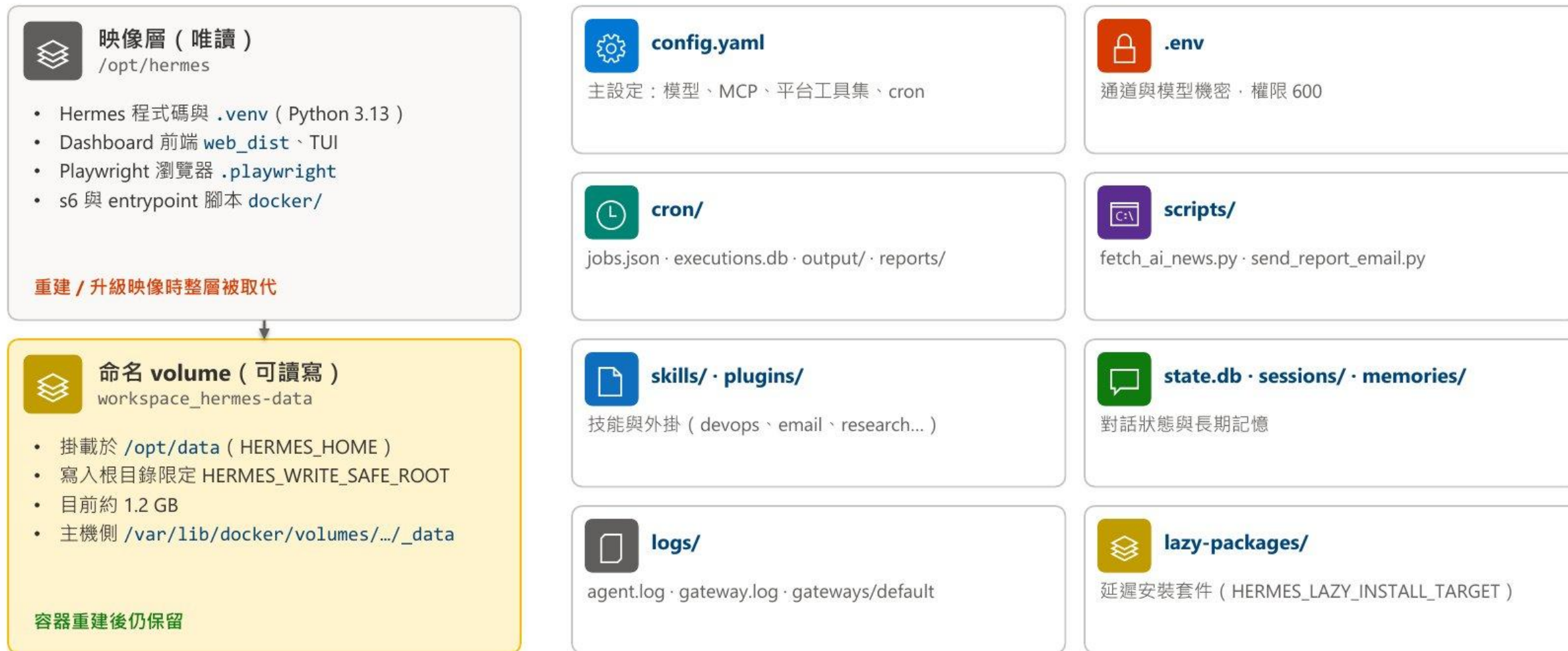


入口	主機綁定	認證與保護	備註
Dashboard	127.0.0.1:8000	Basic Auth (帳密由 .env 注入)	只在本機瀏覽器開啟；未對區網開放
LINE webhook	127.0.0.1:8646	LINE Channel Secret、LINE_ALLOWED_USERS 白名單	LINE 需 HTTPS 回呼，必須自備隧道；本檢視未驗證隧道是否存在
容器出站	– (無入站)	各服務金鑰存放於 /opt/data/.env	Email、LLM、MCP、RSS 皆為容器主動連出

同機其他專案的容器 (資料庫、API、n8n 等) 位於各自的 bridge 網路，不與 hermes 互通。

資料持久化：映像是唯讀程式碼，`/opt/data` 才是狀態

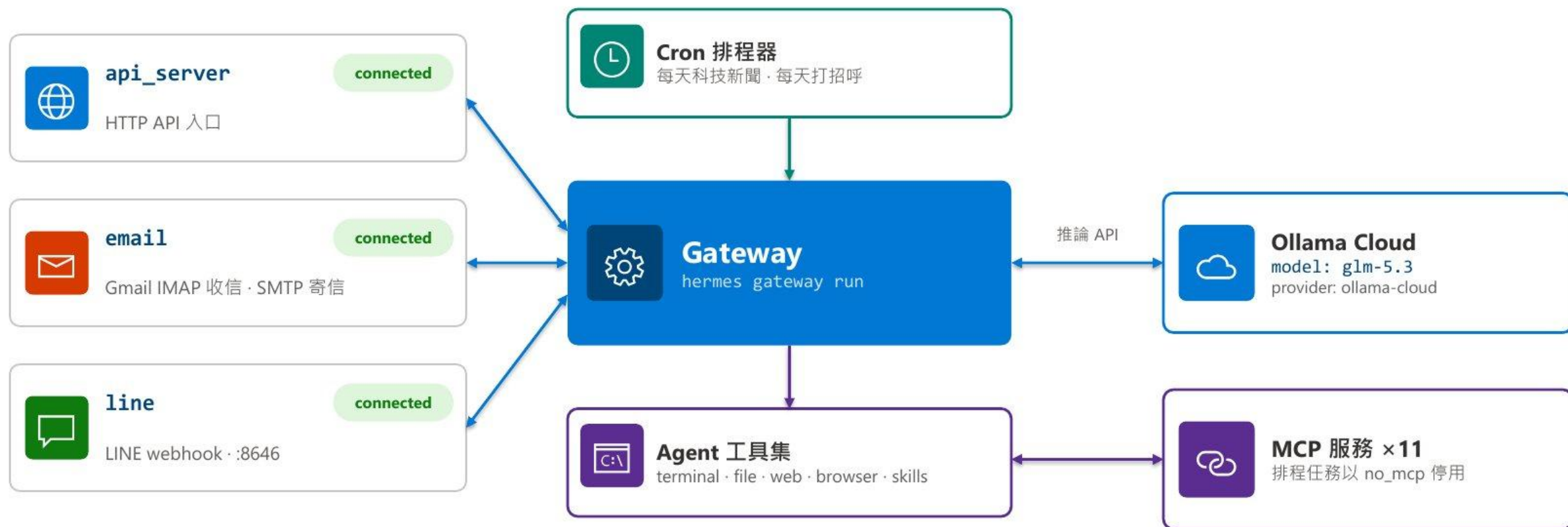
升級映像或重建容器時，只有 volume 內的設定、排程、腳本與對話記錄會被保留



備份重點：cron/jobs.json、.env、scripts/；job 定義只存在 volume，沒有異地備份。

Gateway 樞紐：三個通道已連線，推論交給雲端模型

狀態取自 gateway_state.json (2026-10-02 14:33 UTC) : api_server、email、line 皆為 connected

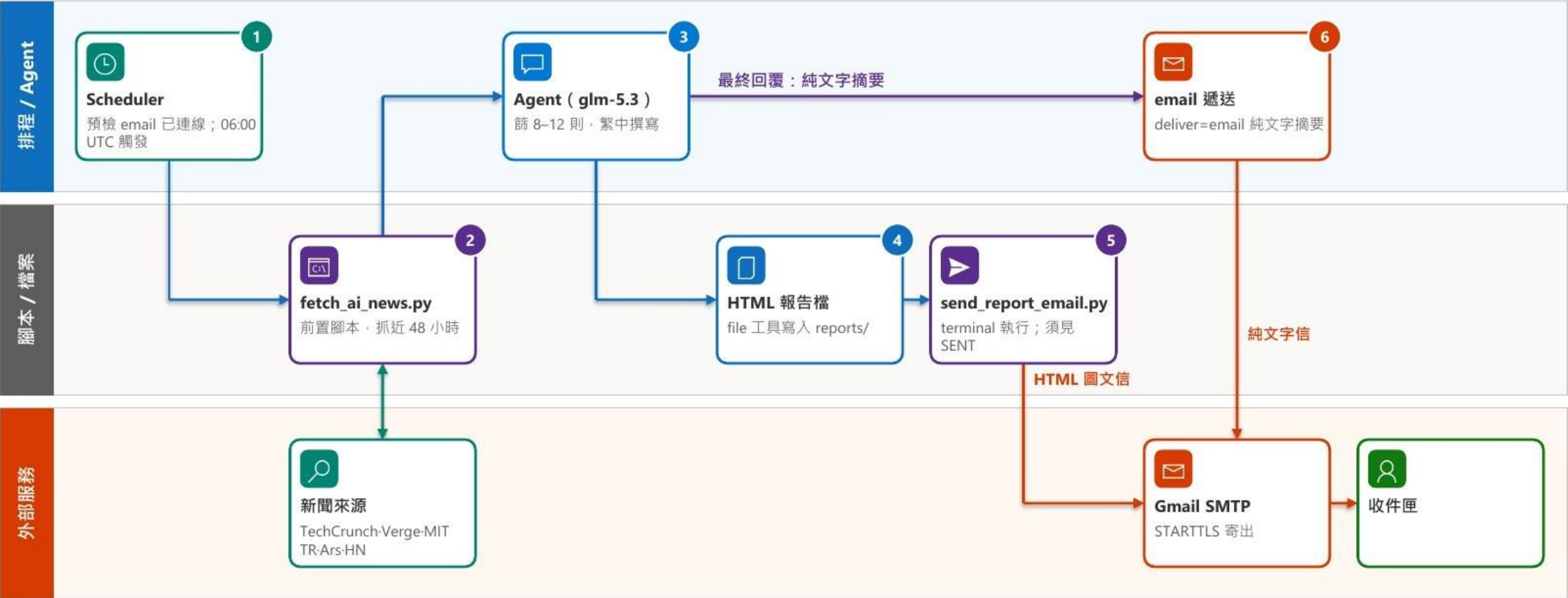


config.yaml 所列 MCP 服務 (其中 7 個尚未互動登入，會每 5 分鐘輸出 OAuth 警告，不影響排程)



排程流程 ①：每天科技新聞 (b71a65ee0d9c)

cron 0 6 * * * (UTC) → 前置爬取 → Agent 撰寫 → 寄出 HTML 圖文信，並再寄一封純文字摘要



成功條件：send_report_email.py 輸出 SENT，否則回覆第一行必須是 [CRON_FAILURE]。結果：每天收到兩封信（HTML 圖文信 + 純文字摘要）。

工具白名單 terminal · file · web · browser · code_execution · skills · no_mcp；execute_code 在排程中一律被拒。

排程流程 ②：每天打招呼 (4c9b336c55e5) 與時區換算

cron 0 22 * * * (UTC) = 台北每天 06:00；不需要工具，Agent 直接寫信，由 Hermes 遞送



時區換算：台北 UTC+8，cron 以 UTC 計

台北時間
06:00

- 8 小時

UTC (前一天)
22:00

→ cron 運算式 0 22 * * *

prompt 重點

- 繁體中文 (台灣用語)，朋友口吻，依日期 / 星期 / 季節變化
- 結構：問候、2-4 句鼓勵、今日小行動、結尾祝福；名言不得捏造
- 只輸出信件本身，不要前言與後記 (多餘前言會被當信件內容寄出)

項目	每天科技新聞	每天打招呼
Job ID	b71a65ee0d9c	4c9b336c55e5
cron (UTC)	0 6 * * *	0 22 * * *
台北時間	14:00 (要 06:00 須改 0 22 * * *)	06:00
前置腳本	fetch_ai_news.py	無
工具	terminal、file、web、browser、code_execution、skills、no_mcp	不使用工具
遞送	email：HTML 信 + 純文字信	email：純文字信
下次執行	2026-10-03 06:00 UTC	2026-10-02 22:00 UTC

安全與信任邊界：誰能碰到什麼

從網際網路到機密 volume 的五道邊界，與目前已落實及待改善的項目



✓ 已落實的控制

- 兩個連接埠都只綁 127.0.0.1，不暴露到區網
- Dashboard 啟用 Basic Auth；帳密由 `.env` 注入，`.env` 已列入 `.gitignore`
- Email、LINE、模型金鑰只放在 volume 內 `/opt/data/.env` (600)
- 排程中 `execute_code` 一律拒絕 (`approvals.cron_mode: deny`)
- 新聞 job 以 `no_mcp` 與工具白名單縮小權限；寫入限於 `HERMES_WRITE_SAFE_ROOT`

⚠ 待改善與風險

- LINE 需對外 HTTPS 回呼；隧道本身未驗證，公開前請確認簽章驗證與 `LINE_ALLOWED_USERS`
- Gmail 帳戶密碼與應用程式密碼曾出現在對話記錄，建議撤銷並重建
- 映像使用 `latest`，版本可能漂移；建議固定明確 tag 或 digest
- job 與腳本只在 volume，沒有異地備份
- 7 個 MCP 服務 OAuth 未登入，持續輸出警告

維運速查與已知限制

PowerShell 指令；操作排程一律加 -u hermes

需求	指令
啟動 / 更新	<code>docker compose up -d</code>
查看容器 log	<code>docker logs -f hermes</code>
改 .env 後重啟	<code>docker restart hermes</code>
列出排程	<code>docker exec -u hermes hermes hermes cron list</code>
立即執行一次	<code>docker exec -u hermes hermes hermes cron run <job_id></code>
執行記錄	<code>docker exec -u hermes hermes hermes cron runs <job_id></code>
健康檢查	<code>docker exec -u hermes hermes hermes cron doctor</code>
暫停 / 恢復	<code>docker exec -u hermes hermes hermes cron pause resume <job_id></code>
Dashboard	<code>http://127.0.0.1:8000</code>

已知限制



-u hermes

避免以 root 寫入 jobs.json 造成權限錯亂



.env 變更

需重啟容器才會載入



失敗偵測

Hermes 只認回覆第一行 [CRON_FAILURE]；以「是否收到信」為準，別只看 last_status



時區

cron 以 UTC 計算；台北 = UTC+8



容器依賴

Docker Desktop 與容器必須持續運行才會觸發排程

重點回顧

Docker Hermes Container 運行架構一頁帶走

1

個容器

hermes · 由 s6-overlay 監督 Dashboard 與 Gateway

2

個本機埠

8000 Dashboard、8646 LINE，皆綁 127.0.0.1

1

個 volume

/opt/data 保存設定、排程、腳本與記憶 (≈1.2 GB)

3

個通道連線

api_server · email · line

2

個排程 job

每天科技新聞、每天打招呼，皆遞送到 email

運行重點：容器不死、排程才會跑；狀態全在 /opt/data；對外只有 LINE webhook 需要 HTTPS 隧道。

資料來源：docker-compose.yml、docker inspect、容器內程序與 gateway_state.json 檢視，以及先前兩份排程配置報告（2026-10-02）。