📦 Illumio PCE Monitor - 開發移交文件包

移交對象: Antigravity Development Team

目前版本: V70.0 (Stable)

日期: 2026-02-15

---

1. 專案概觀 (Project Overview)

本專案是一個 **Python-based、無 Agent (Agentless)** 的監控工具，用於透過 REST API 監控 Illumio Core (PCE) 的健康狀態、資安事件與流量行為。

核心檔案結構

檔案名稱	用途說明
illumio_pce_monitor.py	**主程式**。包含所有邏輯 (API Client, 運算引擎, UI, SMTP)。
illumio_pce_config.json	**設定檔**。儲存 API 憑證、SMTP 設定與監控規則 (Rules)。
illumio_pce_state.json	**狀態檔**。儲存 last_check 時間戳、歷史計數與告警冷卻 (Cooldown) 狀態。
logs/	**日誌目錄**。存放 illumio_pce_events.log 與 illumio_pce_traffic.log (僅供除錯，不參與運算)。

---

2. 架構與設計模式 (Architecture & Design)

為了在無需資料庫的情況下運作，本工具採用 **「無狀態運算 + JSON 狀態持久化」** 的模式。

2.1 類別結構 (ApiMonitorEngine)

所有核心邏輯皆封裝於 ApiMonitorEngine 類別中，以避免全域變數汙染。

- **__init__**: 載入設定與狀態。
- **核心方法 (Public)**:
  - analyze(): Crontab 進入點，執行完整的 ETL (Extract-Transform-Load) 與告警流程。
  - run_debug_mode(): 開發者測試用，不寫入 State，僅印出 Console。
  - send_email(): 負責 HTML 報表渲染與 SMTP 發送。
- **內部邏輯 (Private)**:

- _execute_traffic_query(): 處理 Illumio 非同步 Traffic Query API。
- _get_matches(): 負責記憶體內的規則比對。
- _trigger_alert(): 負責冷卻時間檢查與告警物件生成。

2.2 關鍵演算法 (Key Algorithms)

A. 單次遍歷 (One-Pass Processing) - *效能關鍵*

- **問題**: 若有 50 條規則，傳統寫法會對 API 回傳的 20 萬筆流量進行 50 次迴圈，導致 $O(N \times M)$ 的複雜度與記憶體爆炸。
- **解決方案**:
  1. 找出所有規則中最大的時間窗口 (Max Window)。
  2. 對 API 發送一次大範圍查詢。
  3. 使用單次迴圈遍歷流量，針對每一筆流量同時比對所有規則。
  4. **記憶體優化**: 不儲存完整的匹配清單，僅維護 Top 10 樣本與 Sum/Max 統計值。

B. 混合計算模式 (Hybrid Calculation) - *準確度關鍵*

Illumio Traffic Logs 對於長連線 (Long-lived connections) 的行為特殊，程式實作了自動判斷邏輯：

Python

```
def calculate_mbps_debug(self, flow):
    # 優先嘗試計算區間流量 (精準反映當下頻寬)
    if delta_bytes > 0: return delta_bytes / duration ...
    # 若區間流量為 0 (常見於長連線)，則讀取生命週期總量作為備援
    else: return total_bytes / lifetime_duration ...
```

C. 動態滑動視窗 (Dynamic Sliding Window)

- API 雖然拉取了 (例如) 60 分鐘的資料，但針對設定為 "10 分鐘窗口" 的規則，程式會在記憶體中過濾 timestamp，剔除舊資料，確保告警的精準度。

---

3. 資料結構說明 (Data Schema)

3.1 規則定義 (config['rules'])

Antigravity 接手後若需新增監控類型，請參考此結構：

JSON

```
{
    "id": 1708001234,                    // Unique Timestamp ID
    "type": "traffic",                   // event | traffic | bandwidth | volume
    "name": "Blocked SSH",
    "threshold_type": "count",           // immediate | count
    "threshold_count": 10,               // 觸發閾值
    "threshold_window": 10,              // 時間窗口 (分鐘)
    "cooldown_minutes": 30,              // 冷卻時間
    "pd": 2,                             // Policy Decision (2: Blocked)
    "port": 22,                          // 過濾條件
    "src_label": "role=Web"              // 過濾條件
}
```

3.2 狀態追蹤 (state['alert_history'])

用於防止重複告警：

JSON

```
{
    "1708001234": "2026-02-15T10:00:00Z" // 記錄該規則最後一次告警時間
}
```

---

4. 已知限制與邊界情況 (Known Limitations)

請 Antigravity 團隊注意以下 Illumio API 特性與程式限制：

1. **API 查詢上限**: 程式碼中硬限制 MAX_TRAFFIC_RESULTS = 200000 筆，以防止 512MB/1GB RAM 的 VM 發生 OOM (Out Of Memory)。若部署在高規機器，可調大此值。
2. **長連線誤判風險**: 對於持續數天的連線，Illumio 可能只在結束時寫入 Log，或者間歇性更新 Total Bytes。V70.0 已引入 First Seen 與 Last Seen 欄位協助判讀，但在極端情況下，Volume 告警可能會包含 "過去數天" 的累積量。

3. **Python 縮排敏感度**: 由於 ApiMonitorEngine 類別龐大，維護時請務必小心縮排，避免發生 V60-V67 版本中 AttributeError 的情況（方法脫離類別）。

---

5. 建議開發路線圖 (Roadmap for Antigravity)

為了進一步強化此工具，建議 Antigravity 可往以下方向開發：

- **[P1] 非同步優化 (AsyncIO)**: 目前 requests 為同步阻斷式。若需同時監控多個 PCE Cluster，建議改寫為 aiohttp。
- **[P2] Webhook 整合**: 目前僅支援 Email。可新增 Slack/Teams/Line 的 Webhook 支援 (requests.post)。
- **[P3] Syslog 輸出**: 將聚合後的告警轉發至 SIEM (Splunk/ArcSight)，而非僅依賴 Email。
- **[P4] 設定檔加密**: 目前 API Secret 為明文儲存於 JSON。建議整合 keyring 或簡單的 Base64/AES 混淆。