**資訊工程學系**

**114學年度專題製作期初報告**

**題目：Wi-Fi 封包偵測與分析（被動式）**

指導老師： 教授

組長:高恩在  
組員:

**中華民國 114年 6 月 22 日**

目錄

一、 摘要

二、 研究背景與目的

三、

四、 研究方法

五、 流程圖

六、 預計完成之工作項目及具體成果

七、 工作分配與預定進度甘梯圖

八、 遭遇問題與解決方案

九、 儀器設備需求表

十、 參考資料

十一、 附錄

一、 摘要

本研究透過具備 Monitor 模式的無線網卡，以被動方式擷取 Wi-Fi 無線封包，分析周遭裝置與基地台之間的通訊行為。重點包括 SSID 廣播、裝置探測行為、MAC 位址活動與資料流量特徵，並結合 Wireshark、airodump-ng 等工具進行視覺化與統計分析，期能建立合法且具展示價值之資安監測實驗架構。

二、 研究背景與目的

隨著無線通訊技術普及，Wi-Fi 封包已成為觀察使用者行為與網路安全的重要資訊來源。透過被動式監聽，不僅可收集網路環境資訊，也能分析裝置間互動，對資安教育與攻防展示具有高度價值。

研究目的：

* 探討合法被動偵測 Wi-Fi 封包之技術與應用
* 建立基礎裝置與流量辨識分析流程
* 實作封包分析流程並產出圖表報告

三、 專題內容

| **項目** | **說明** |
| --- | --- |
| 封包類型 | Beacon, Probe Request/Response, Data, QoS |
| 分析指標 | SSID、BSSID、Client MAC、頻道、封包大小、流量趨勢 |
| 觀察主題 | 熱點活動、裝置識別、流量統計、封包時間圖 |
| 工具應用 | Wireshark、airodump-ng、tcpdump、Scapy、matplotlib |

四、 研究方法

使用支援 Monitor Mode 的 USB 無線網卡

透過 airmon-ng 啟用 monitor mode

以 airodump-ng/tcpdump 擷取封包至 .cap 檔案

利用 Wireshark 篩選 beacon、probe、data 封包進行觀察與記錄

製作表格與圖像化統計

五、 流程圖

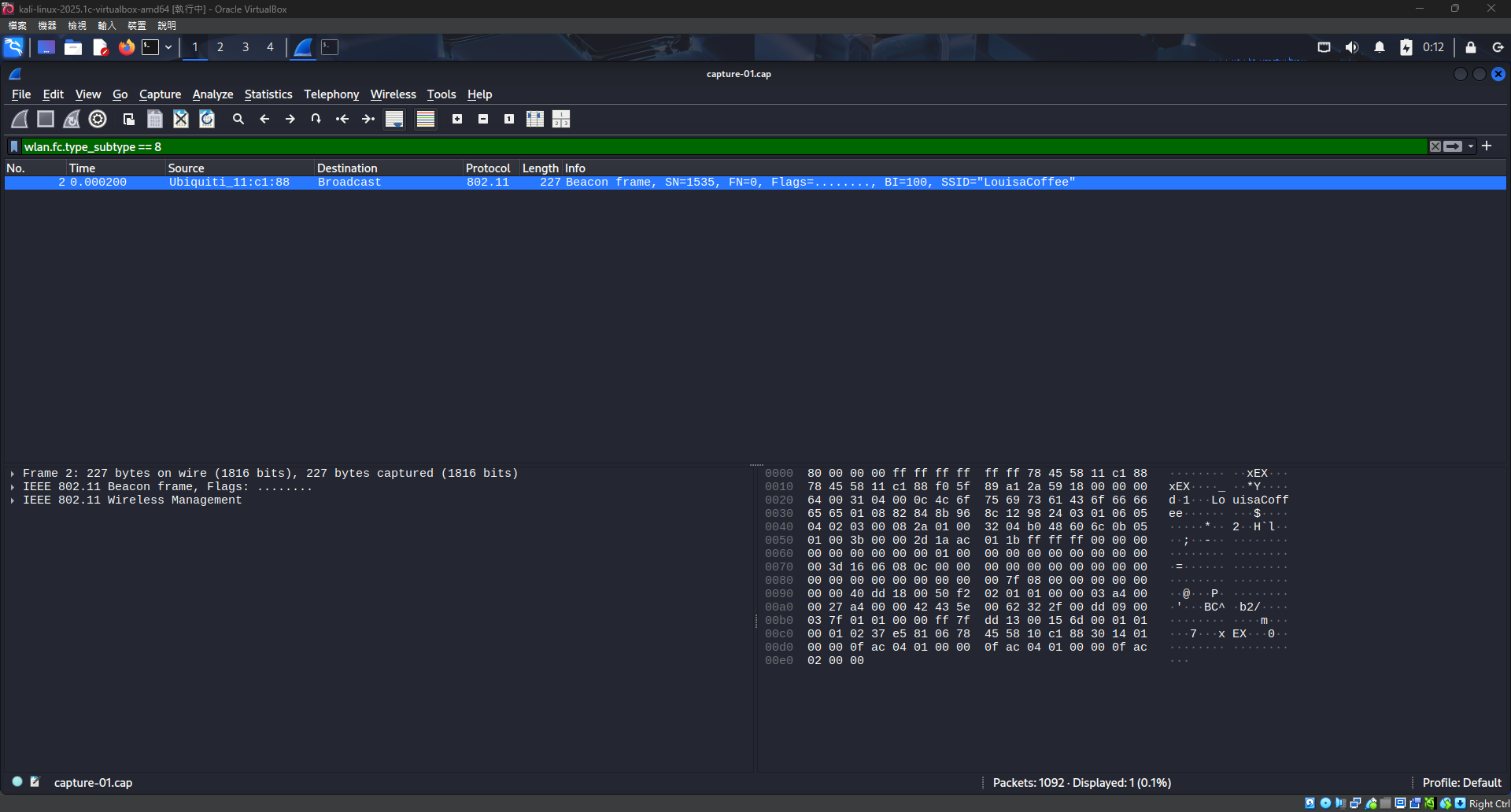


六、 預計完成之工作項目及具體成果

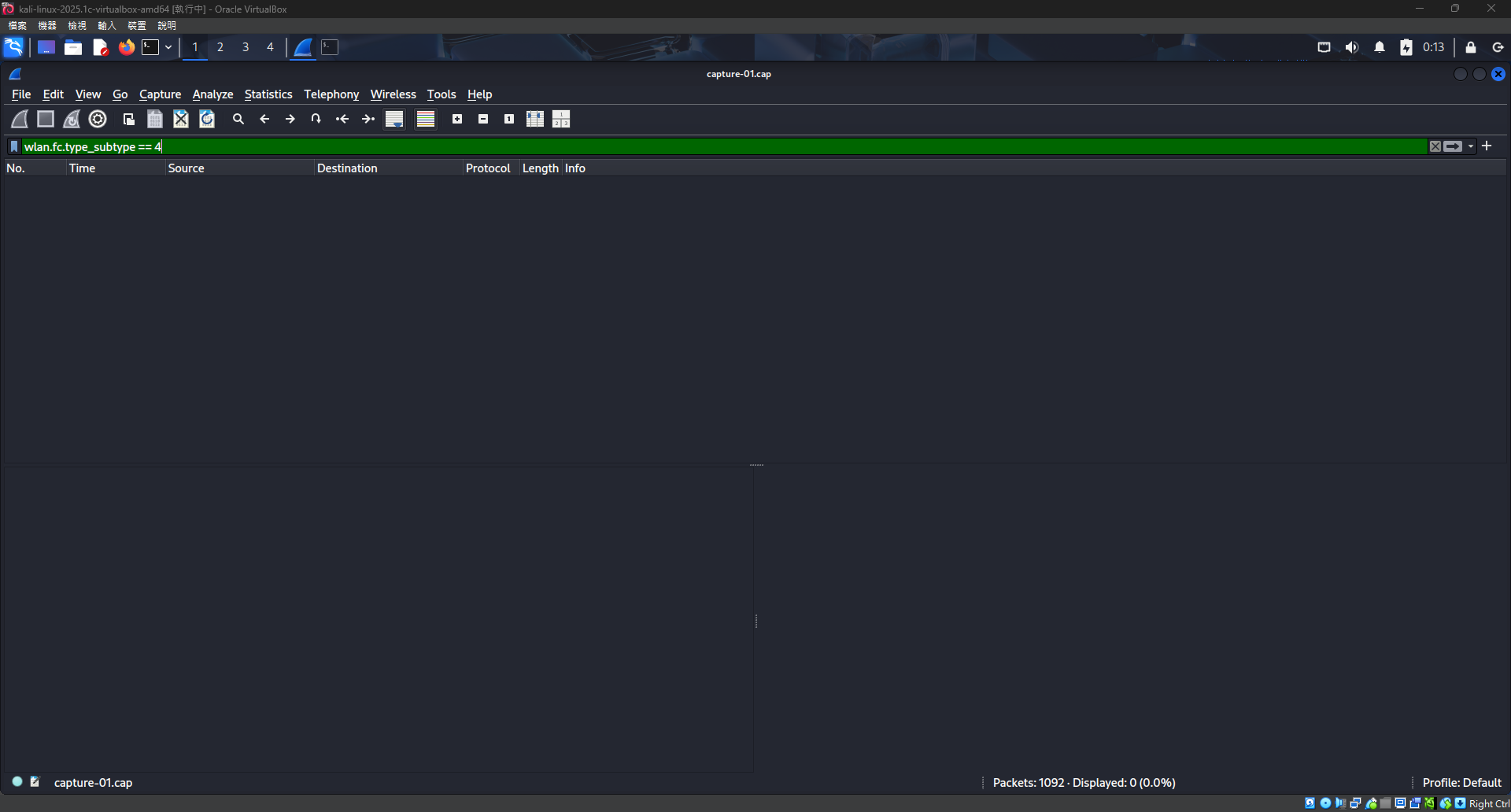
| **項目** | **內容** | **預期成果** |
| --- | --- | --- |
| 封包擷取 | 擷取 Wi-Fi 空中封包 | capture-01.cap 封包檔 |
| 分析紀錄 | 表列裝置與封包活動 | MAC 分析表格 / 流量圖 |
| 程式腳本 | 自動提取封包特徵 | Python 檔：mac\_stats.py |
| 報告彙整 | 實驗分析報告 | PPTX、PDF 報告 |

實驗圖片(以LouisaCoffee：78:45:58:12:C1:88為例)

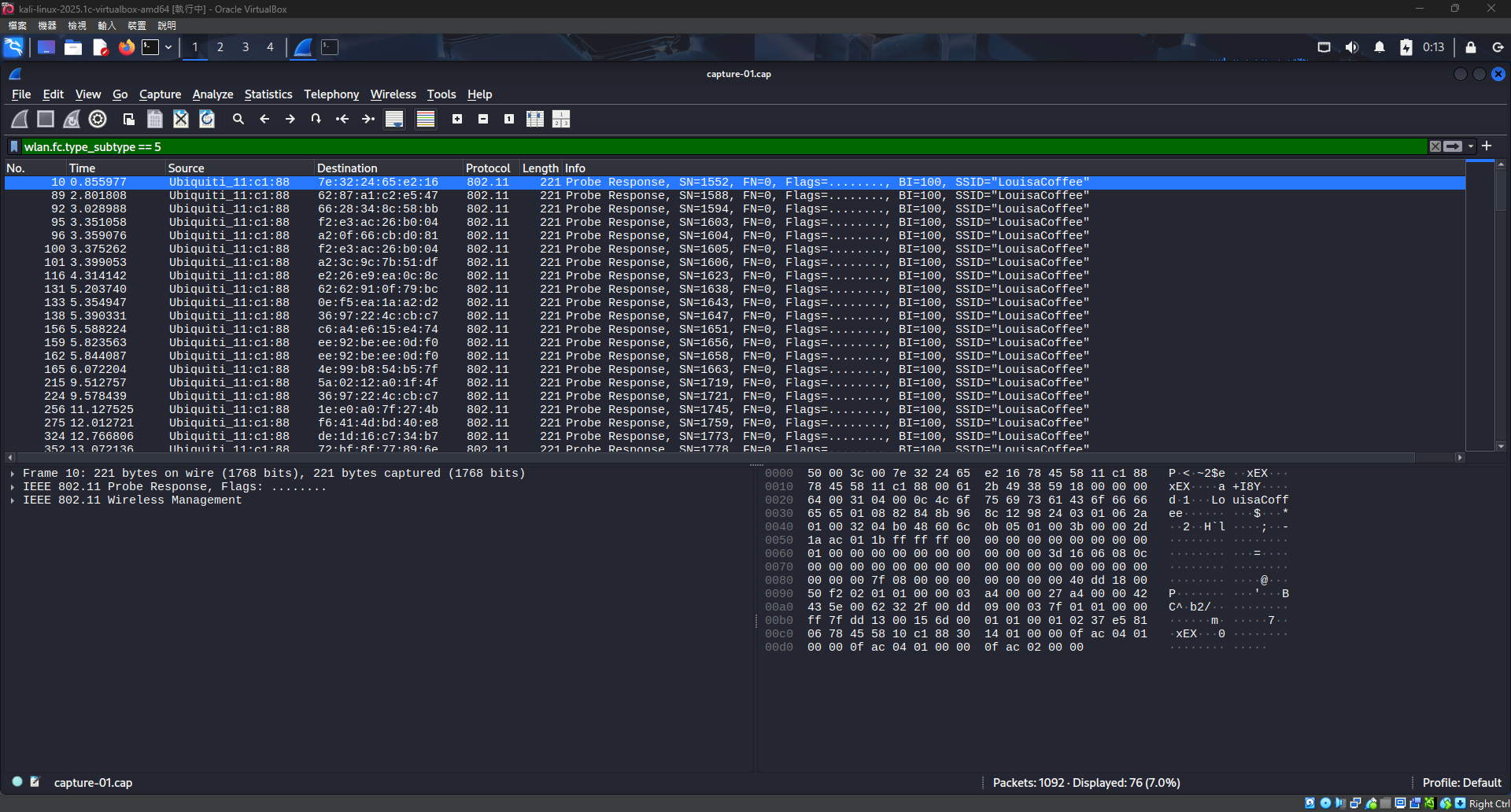
顯示 Beacon 封包



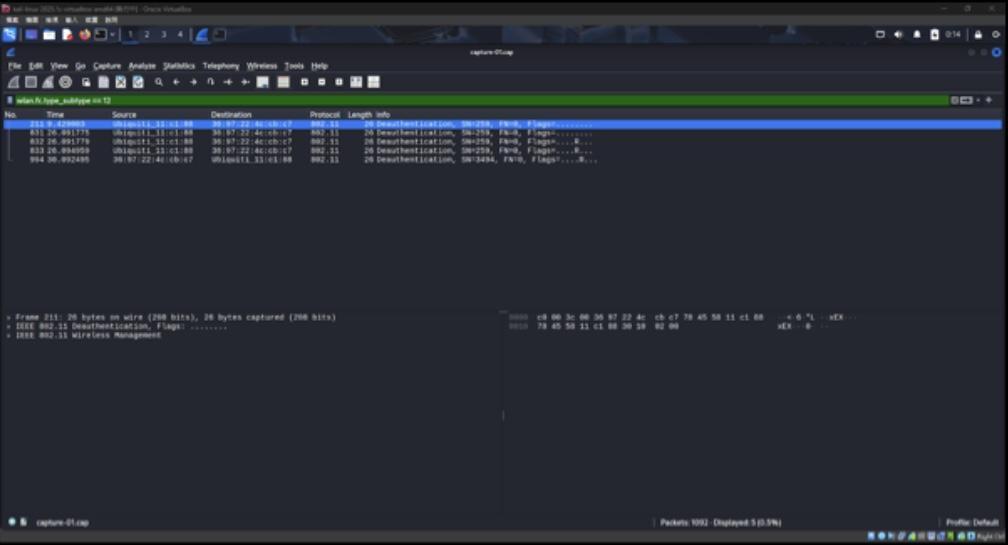
顯示 Probe Request



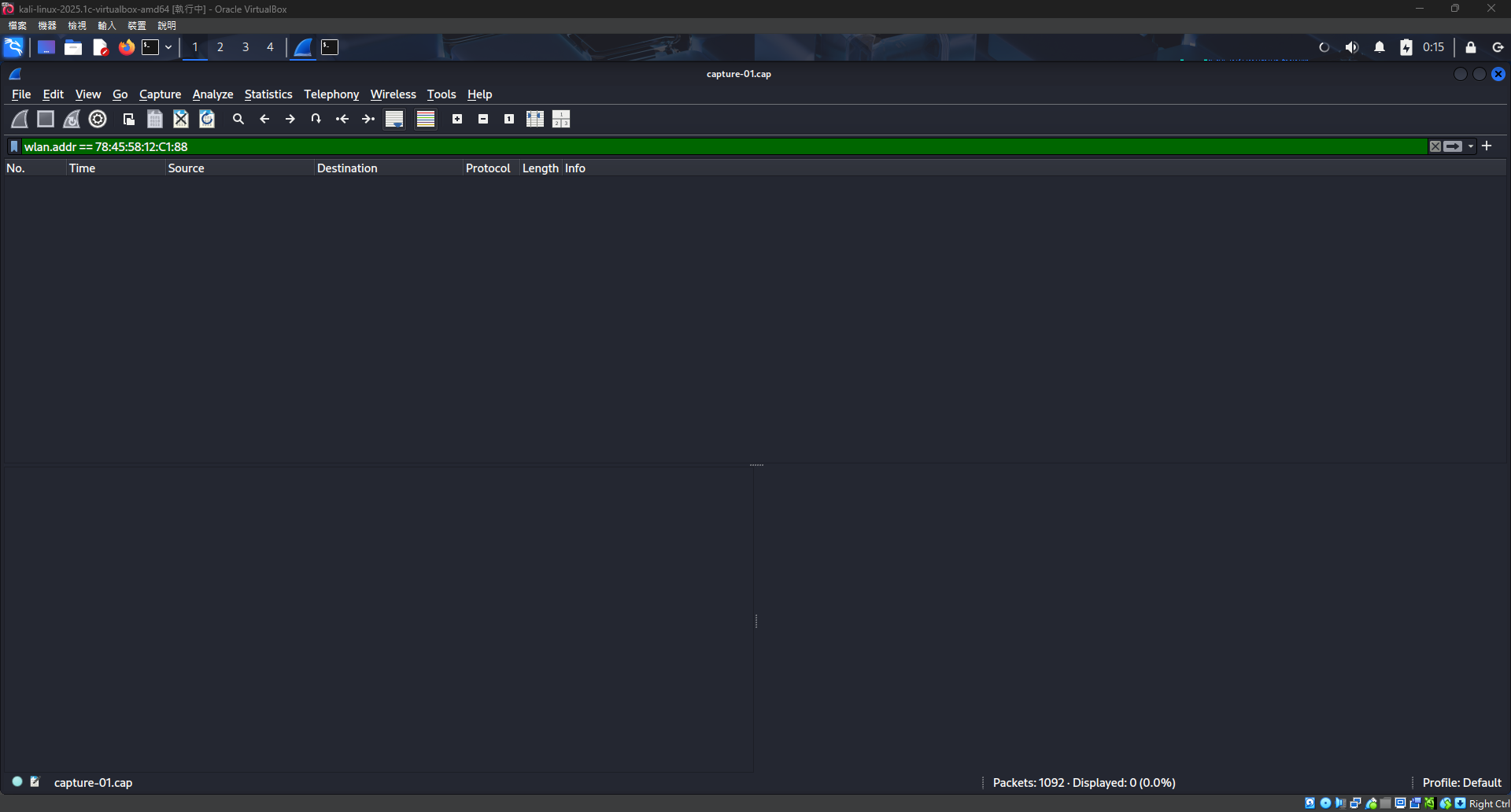
顯示 Probe Response



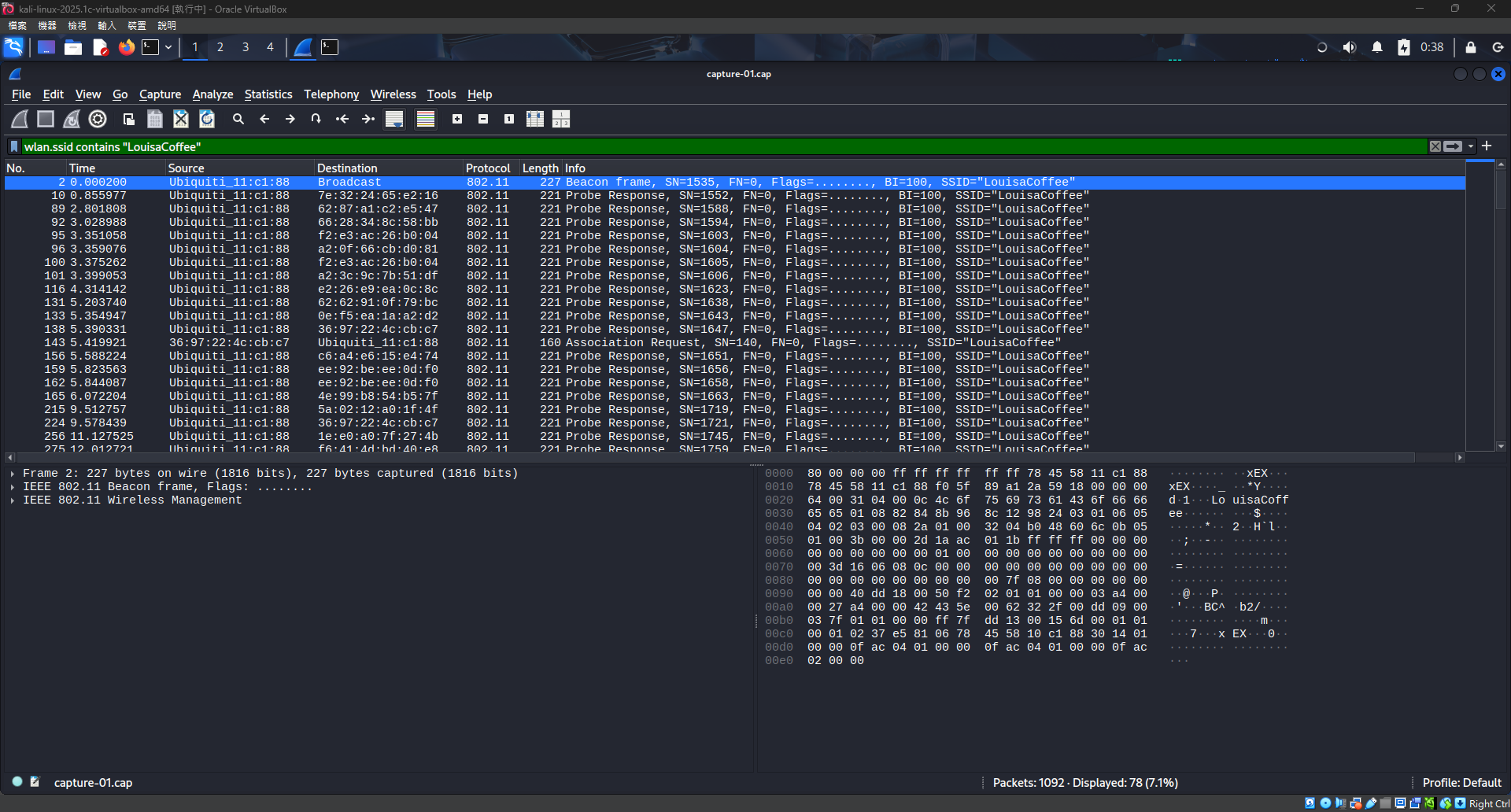
顯示 Deauth 封包



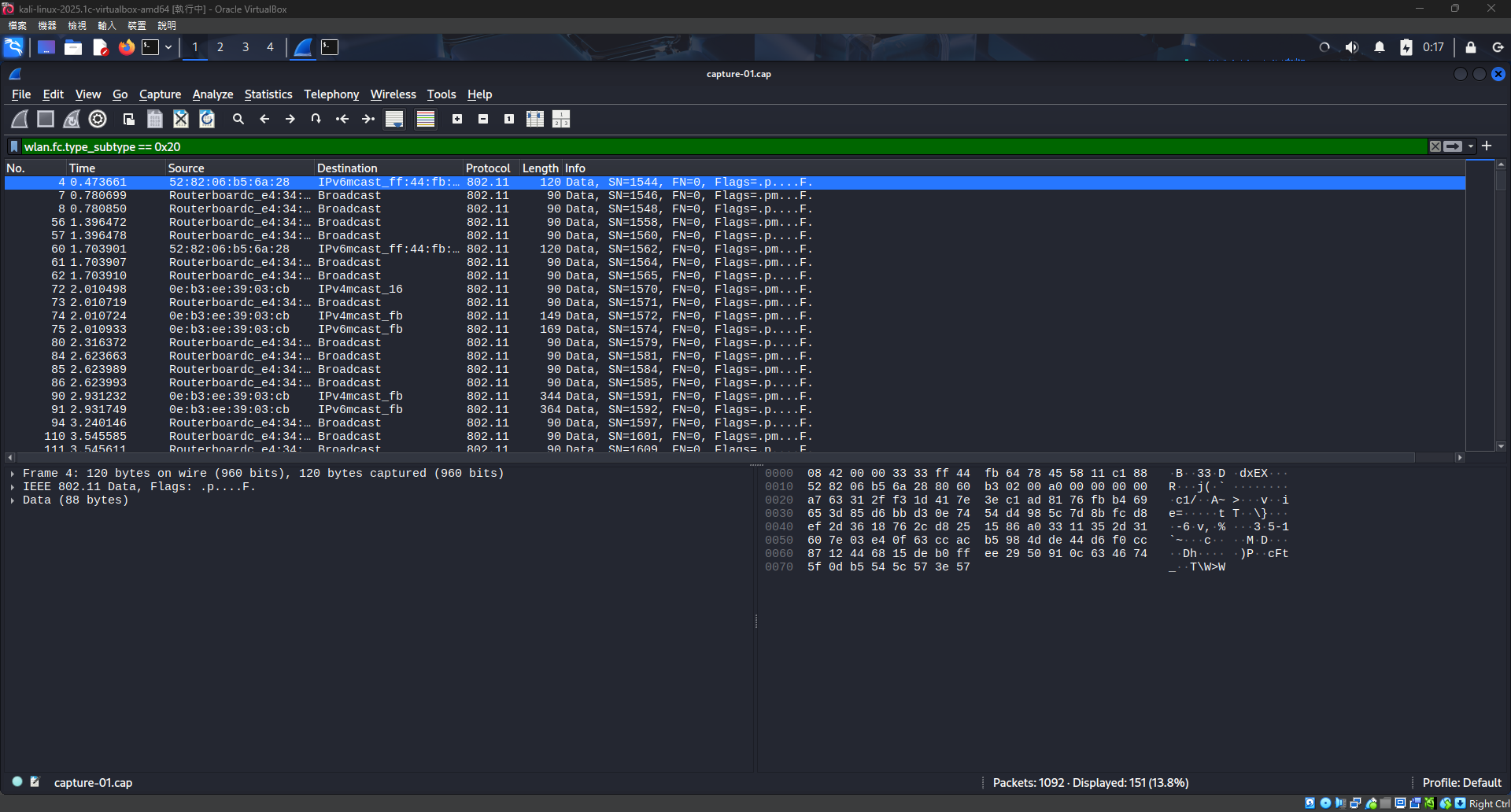
顯示某個 MAC 所有封包



顯示特定 SSID



顯示 Data 封包



已使用 Wireshark 篩選語法（Filter）

| **功能** | **Filter 語法** |
| --- | --- |
| 顯示 Beacon 封包 | wlan.fc.type\_subtype == 8 |
| 顯示 Probe Request | wlan.fc.type\_subtype == 4 |
| 顯示 Probe Response | wlan.fc.type\_subtype == 5 |
| 顯示 Deauth 封包 | wlan.fc.type\_subtype == 12 |
| 顯示某個 MAC 所有封包 | wlan.addr == aa:bb:cc:dd:ee:ff |
| 顯示特定 SSID | wlan.ssid contains "YourSSID" |
| 顯示 Data 封包 | wlan.fc.type\_subtype == 0x20 |

封包偵測與分析報告

### 測試 SSID：

### LouisaCoffee

### 測試環境：公共場所（被動封包擷取）

### 測試工具：Wireshark + 擷取介面卡（Monitor Mode）

## 合法用途聲明

本報告僅針對 Wi-Fi 廣播封包進行被動擷取與技術研究分析，不涉及任何通訊解密或未授權資料擷取。

所有資料均已匿名化處理，並符合《通訊保障及監察法》與《個人資料保護法》之規定。

本分析僅作為資訊安全教學、網路行為研究之用，不對外揭露個別裝置資訊或身份。

## 觀察目標

* SSID：LouisaCoffee
* BSSID（AP MAC）：78:45:58:12:C1:88（Ubiquiti 設備）
* 測試時間長度：數分鐘（被動擷取）

## 1.Beacon Frame 分析

* Beacon Frame 為 AP 周期性廣播自身存在的管理封包。
* 擷取結果顯示：  
  + 來源位址：Ubiquiti 11:C1:88:xx:xx:xx（OUI 顯示為 Ubiquiti 裝置）
  + 目的位址：Broadcast（ff:ff:ff:ff:ff:ff）
  + SSID 名稱：LouisaCoffee
  + Beacon Interval：100 TU（約 102 毫秒）
* 結論：AP 正常廣播 SSID，表示無線網路運作正常。

## 2.Probe Request 分析

* 未擷取到 wlan.fc.type\_subtype == 4 的封包（主動搜尋封包）。
* 可能原因推測：  
  + 現場裝置未啟動 Wi-Fi 掃描（螢幕關閉或靜置）
  + 擷取時間不夠長
* 建議後續測試：  
  + 延長擷取時間
  + 使用裝置手動刷新 Wi-Fi 清單以觸發掃描

## 3.Probe Response 分析

* 成功擷取多筆 wlan.fc.type\_subtype == 5 封包。
* 觀察結果：  
  + 來源：Ubiquiti 11:C1:88:xx:xx:xx
  + 目的：不同行動裝置（如 7E:32:24:xx:xx:xx 等，僅顯示前 3 bytes）
  + 對象均為 LouisaCoffee 的探測請求
* 推論：有多台裝置曾主動尋找此 SSID，AP 對其進行回應。

## 4.Deauthentication Frame 分析

* 擷取到 wlan.fc.type\_subtype == 12 封包（解除認證封包）
* 封包內容：  
  + 來源：Ubiquiti AP（11:C1:88:xx:xx:xx）
  + 目的：裝置 MAC 為 36:97:22:xx:xx:xx
  + Reason Code：0x0002（之前的認證無效）
* 推論：可能為裝置離線、AP 重啟、或 roaming 行為，非異常狀況。

5.Data Frame 分析（QoS Data）

* 使用過濾器 wlan.fc.type\_subtype == 0x20 擷取 QoS 資料封包
* 特徵如下：  
  + 來源裝置：RouterBoard（MikroTik）、其他匿名化裝置
  + 目的地：多為 Broadcast 或 IPv6 Multicast 位址
  + 封包大小：90～149 bytes
  + 內容特徵：疑似包含 mDNS、DHCPv6、IPv6 ND 等協定資料
  + 部分 raw payload 為非明文，但可辨識 QoS 結構
* 推論：顯示曾有背景網路活動發生，裝置與 AP 有資料互動。

**綜合結論**

1. SSID = "LouisaCoffee" 確實由一台 Ubiquiti 裝置廣播。
2. 存在多筆 probe response 與 association request，顯示周圍有多台行動裝置主動互動。
3. 部分裝置進行資料傳輸，包含 QoS Data frame，表示裝置與 AP 間曾有實際網路活動。
4. 擷取期間亦觀察到合法 Deauth 行為，可作為 Wi-Fi 行為結束的指標。

七、 工作分配與預定進度甘梯圖

| **天次** | **任務** | **負責人** | **備註** |
| --- | --- | --- | --- |
| 第1天 | 工具安裝與測試 | 本人 | 測試 usb 無線網卡 |
| 第2天 | 封包擷取與分類 | 本人 | 用 Wireshark 製圖 |
| 第3天 | Python 資料處理 | 本人 | 使用 Scapy 抽取特徵 |
| 第4天 | 撰寫分析報告 | 本人 | 統一格式與排版 |

八、 遭遇問題與解決方案

| **問題描述** | **解決方案** |
| --- | --- |
| wlan0mon 無法啟用 | 改用 iw dev 檢查介面名稱，或直接使用 wlan0 |
| 驅動不支援 monitor mode | 改用 Alfa AWUS036NHA 等具穩定支援之晶片 |
| 虛擬機未抓取 USB 網卡 | 手動連接至 VM 裝置設定 |
| 封包過多難分析 | 使用 Wireshark filter + 自動化腳本輔助篩選 |

九、 儀器設備需求表

| **名稱** | **數量** | **備註** |
| --- | --- | --- |
| 支援 monitor mode 無線網卡 | 1 | Realtek RTL88x2bu |
| Kali Linux 虛擬機 | 1 | 建議實體或能穩定掛載 USB |
| Wireshark | 1 | 封包圖形分析工具 |
| Python + Scapy | 1 | 可用於後續封包統計與視覺化 |

十、 參考資料

Kali Linux 官方文件：<https://www.kali.org/docs/>

Aircrack-ng 官方網站：<https://www.aircrack-ng.org/>

Wireshark Display Filter 手冊：<https://wiki.wireshark.org/DisplayFilters>

[YouTube] WiFi Packet Sniffing with Wireshark

實測封包來源：使用家用 AP 與手機裝置進行被動式分析

## 十一、 附錄（合法依據摘要)

附錄一、合法依據摘要

| **法規** | **條文摘要** | **說明** |
| --- | --- | --- |
| 《個人資料保護法》 | 個資定義包含可識別個人裝置資訊（如 MAC） | 報告已遮蔽個資，未進行身份推測 |
| 《通訊保障及監察法》第3條 | 禁止未經同意擷取通訊內容 | 本分析僅擷取廣播封包，不涉私密內容 |
| 《電信管理法》 | 禁止未授權使用頻譜或干擾他人 | 本測試屬被動擷取，不發送任何封包 |

結語：本報告所有數據與結果，皆在未干擾實體網路運作前提下完成，並經匿名處理。分析目的為促進無線網路行為理解與資安教育推廣，無任何未授權通訊行為。

附錄二、Python 腳本內容(補充文中提到的**mac\_stats.py**)

import pyshark

import csv

from collections import defaultdict

# 設定封包檔案路徑

capfile = 'capture-01.cap' # 你擷取的封包檔案

output\_csv = 'mac\_statistics.csv' # 輸出的統計結果

# 初始化 MAC 地址統計字典

mac\_stats = defaultdict(int)

# 讀取封包檔案

cap = pyshark.FileCapture(capfile, display\_filter="wlan.fc.type\_subtype == 4 or wlan.fc.type\_subtype == 8") # 只處理 Probe Request 和 Beacon

# 解析封包

for packet in cap:

try:

# 提取來源和目的地 MAC 地址

src\_mac = packet.wlan.sa

dst\_mac = packet.wlan.da

if src\_mac: # 若有來源 MAC

mac\_stats[src\_mac] += 1

if dst\_mac: # 若有目的 MAC

mac\_stats[dst\_mac] += 1

except AttributeError:

continue

# 輸出統計結果到 CSV 文件

with open(output\_csv, 'w', newline='') as csvfile:

fieldnames = ['MAC Address', 'Count']

writer = csv.DictWriter(csvfile, fieldnames=fieldnames)

writer.writeheader()

for mac, count in mac\_stats.items():

writer.writerow({'MAC Address': mac, 'Count': count})

print(f"MAC 地址統計已儲存到 {output\_csv}")

**主要步驟解釋：**

1. **讀取封包檔案**：  
   使用 pyshark.FileCapture 讀取 .cap 檔案，並設置過濾條件，只選取 Beacon 與 Probe Request 封包。
2. **提取 MAC 地址**：
   * 使用 packet.wlan.sa 提取來源 MAC 地址。
   * 使用 packet.wlan.da 提取目的地 MAC 地址。
3. **統計 MAC 地址**：
   * 用 defaultdict(int) 記錄每個 MAC 地址出現的次數。
   * 每次遇到封包時，來源或目的 MAC 地址的計數加 1。
4. **輸出結果**：
   * 統計結果會輸出為 mac\_statistics.csv 文件，其中包含每個 MAC 地址及其出現的次數。