應用AP調查接觸史

tags: CNL

1. 發想與目的

受到政府推出的台灣社交距離 app 啟發，能夠在節省許多人力的情況下，提升疫調的效果，阻斷傳播鍊。
AP 在許多人群聚集的地方都有設置，例如學校、百貨公司等等，因此想使用 AP 來協助政府做接觸史的確認。

2. 設計考量

(1) 接觸者的定義

AP訊號可觸及的範圍大小受到許多因素的影響，但相對於政府在公布接觸史時總是以整棟建築物或附近地區做為接觸的範圍，使用 AP 作為接觸依據的範圍要小很多，但也比較精確。
假設有一新加入某 AP 的使用者，我們定義他的接觸者為以下兩者：

- 在此時間回推一小時內，連線過 AP 的使用者
- 往後此使用者連線期間，新加入的使用者

以小時為單位的界定方式是可以更改的，這裡只是作為假設的前提。

(2) 隱私安全

接觸史的製作一定程度上會侵犯到使用者的隱私權，因此必須設計一套方法，不只能避免要求使用者提供過多的使用者資訊，又可以兼顧接觸史的正確性。

- **用什麼來代表 client 的身份?**
  unique 或接近 unique 的 ID 是最適合讓電腦辨認身份的形式。
- **ID的產生方式和使用者資訊的選用**
  ID 如果要作為辨識身份的依據的話，其中一定要包含使用者的資訊。

但明文的傳輸安全性和隱私性都太低，所以最好的方法是將特定的使用者資訊通過 hash function，轉換成 ID。

- **結論**
  使用 key 配合時間經過 cryptographic hash function 產生 ID 是最安全的做法。
  client 每天取得 key 之後，儲存起來，藉此生成 ID 來表明自己的身份。
  cryptographic hash function 的性質確保了沒有人可以透過這個 ID 來反推出 client 持有的 key。
  這項設計額外的好處，是當 client 確診，必須提供自己的資訊時，不需要事先紀錄所有用過的 ID 再上傳，只需要上傳 key
  server 端就可以經由這些 key 和當天的時間來計算出確診者所有可能使用過的 ID。
  - $$k_i \xleftarrow{R} K$$
    $$ID_{ij} \leftarrow MAC_{k_i}(j)$$
    - K 代表一個 key pool，從中 random 選出一把 key k_i 作為 client 第 i 天的 key
    - MAC 是一個 cryptographic hash function，j 則代表現在時間的小時部分(24小時制)。將 k_i 和 j 經過 MAC 以後，即可計算出第 i 天第 j 個小時的 ID。
- **key 和 hash function**
  這裡我們使用了 python 的 `secrets.token_bytes` 來作為隨機選擇 key 的方式。
  而 cryptographic hash function 則是 python 的 `hamc class` 。

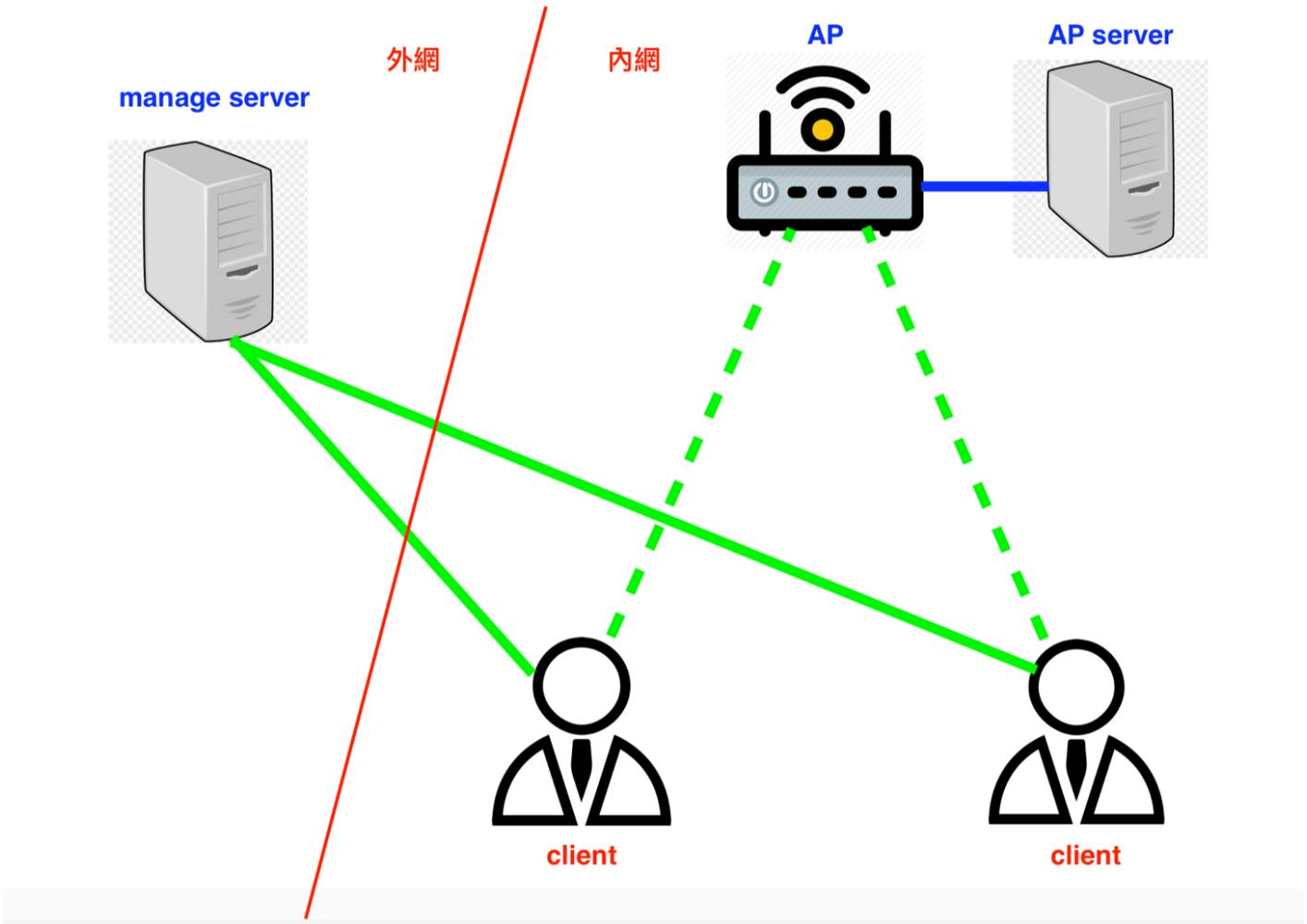
(3) 運算和儲存成本

- 運算成本
  - hash function 運算速度極快，成本低。適合在 client 端進行。
- 儲存空間成本
  - 裝在 AP 上的 server，應該盡量輕量化，不適合儲存大量的資料。因此應該將儲存的成本放在 client 端。
  - server 儲存 key 和 client 之間的 mapping 的話，成本過高。必須尋找更節省儲存空間的方式
  - 接觸史的特殊性：
    病毒潛伏期為 14 天是各國普遍相信的基準，也是疫調上追訴的期限。這提供了 server 需要將資料留存多久的參考依據，也可以讓 server 避免儲存過多不必要的資料。

3. 實作架構與流程

(1) 架構

- **AP server**
  連接於 AP 上的伺服器。
  負責確保符合上述接觸者定義的人之間，可以得知對方的資訊。
- **manage server**
  運作於外網的獨立伺服器。
  如果只有 AP server 的話，資料會過於分散，確診者的紀錄難以集中管理，因此需要一個運作於外網的 manage server ，來記錄並彙整確診者的資訊，以提供民眾查詢自己的接觸史，也負責處理確診者上傳足跡的服務。
- **client**
  手機或電腦等行動裝置(在這次實作裡只有電腦)。
  安裝應用程式，確保在出門前保持開啟，才能隨時與 AP server 互動，接收確診者的資訊。

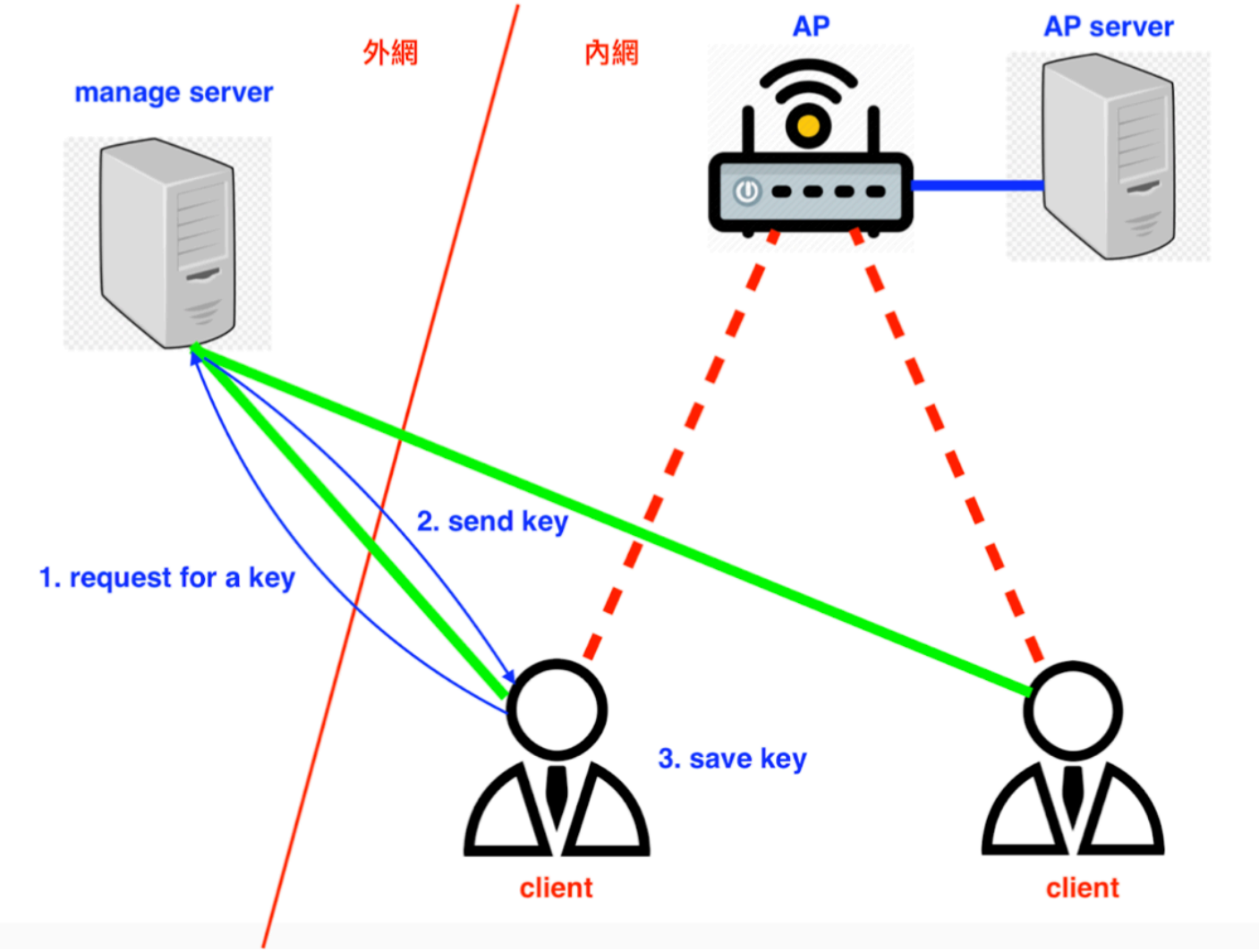


(2) 流程

(a) 取得 key

這一個步驟會發生在 client 出門或是連上 AP 之前。
manage server 在這一個步驟作為 key 的提供者。
client 每天向 manage server 索取 key ，為了確保 client 一天只會得到一把 key ，我們在 client 處做日期的檢查，來決定此次接收到的 key 是否需要丟棄。
在 client 處檢查的做法相比在 server 處檢查簡便許多，不會為 server 增加記錄 client 索取紀錄的成本。
key 最多將會被 client 保存 14 天，超過 14 天的 key 在獲取新的 key 時會被丟棄。

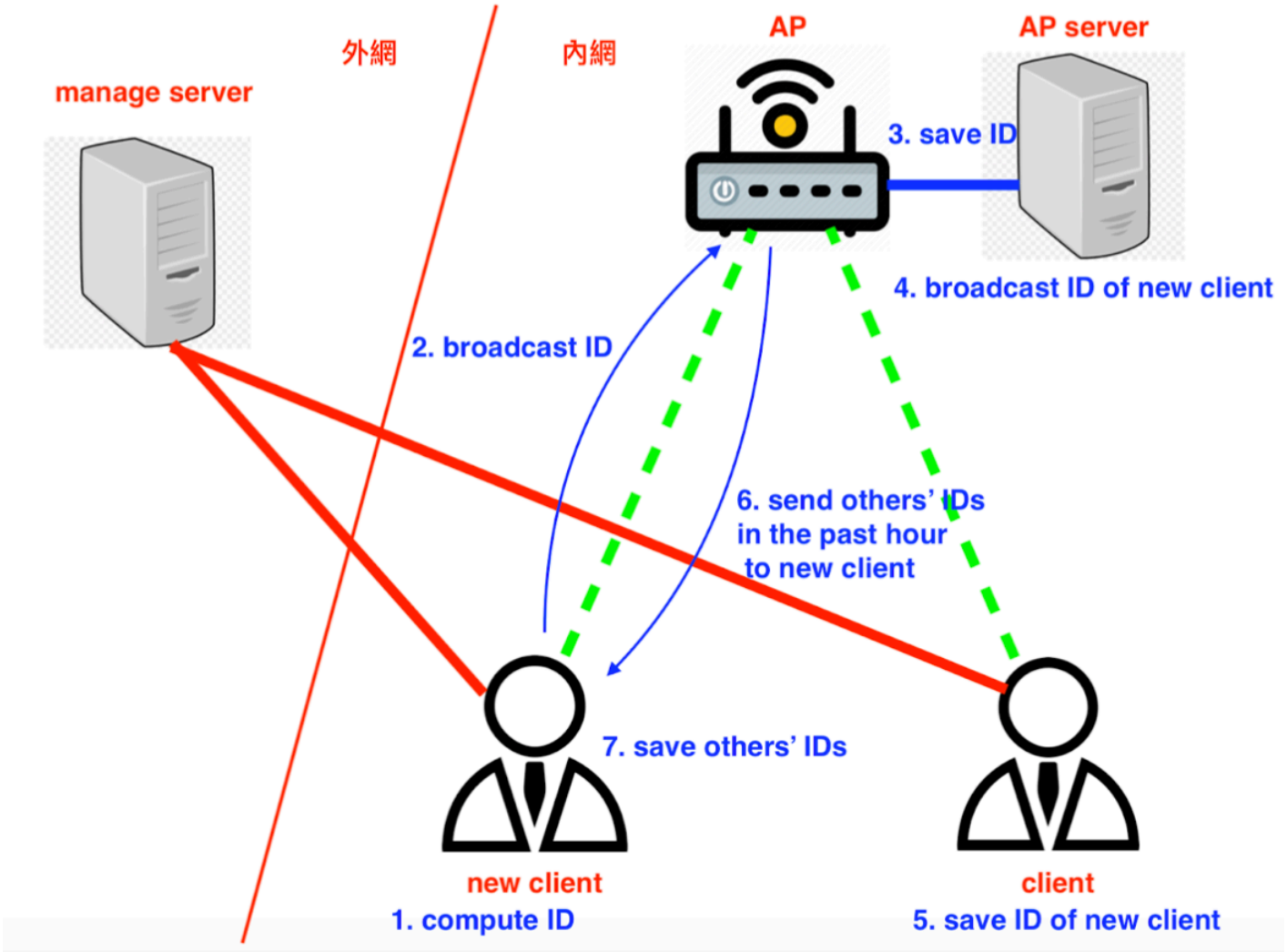
- 實作：
  這部分的實作是以 http 完成，為了提升傳送 key 安全性，實際上線時必須以 https 來傳輸。



(b) 連接 AP

client 在得知自己連上 AP 後，將會根據今天的 Key 以及現在的日期、時間來計算 ID。
AP 以 DHCP 分配 IP ，剛加入 AP 的使用者難以得知 server 的位置，因此 client 必須藉由 broadcast 的方式來告知 AP server 他的 ID。
AP server 在得知有新的使用者加入後，將此 ID 分發給其他連線中的使用者，作為一個接觸的紀錄，這項紀錄會被使用者保存 14 天。同時也把此 ID 記錄起來，在一小時後失效。
最後 AP server 將提供這個新的使用者過去一小時內連線過此 AP 的 ID (符合前述接觸者的定義)。

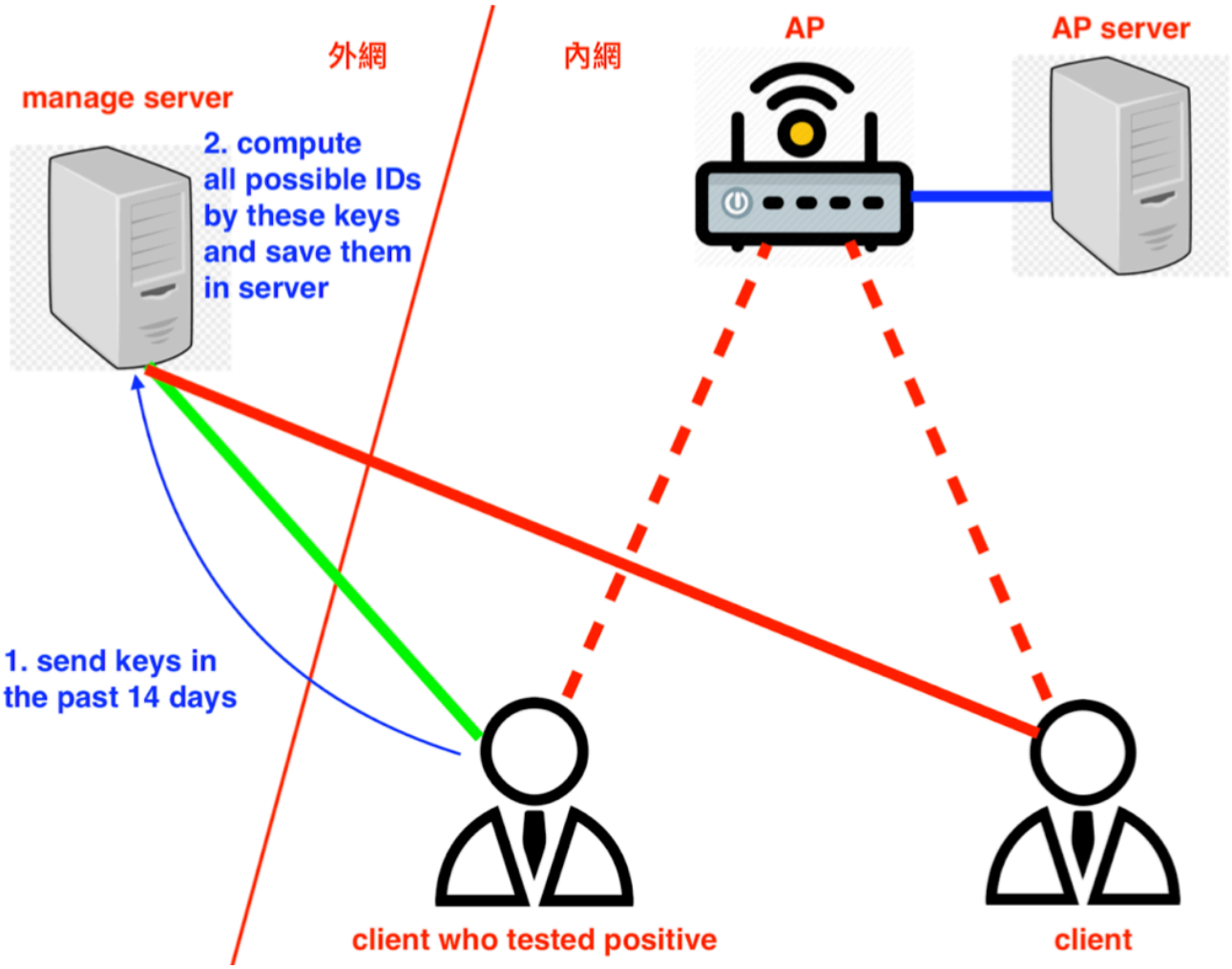
- 實作：
  這部分的實作是以 udp socket 完成， udp 的非連線導向提供了 broadcast 的功能，讓 client 可以使用 broadcast 的方式找到 AP server，也讓 AP server 用簡單的 broadcast 就能達到 ID 的轉傳。



(c) 確診者足跡上傳

確診者在上傳足跡時，只需要提供過去 14 天的 key 。
而 manage server 透過這些 key ，以及過去 14 天的日期和時間，通過和 client 處相同的 hash function 來計算出 client 所有可能送到 AP 裡的 ID。並將這些 ID 歸檔，以提供後續的接觸史查詢。
由於潛伏期為 14 天的緣故，這些檔案在 14 天後也可以自動失效。

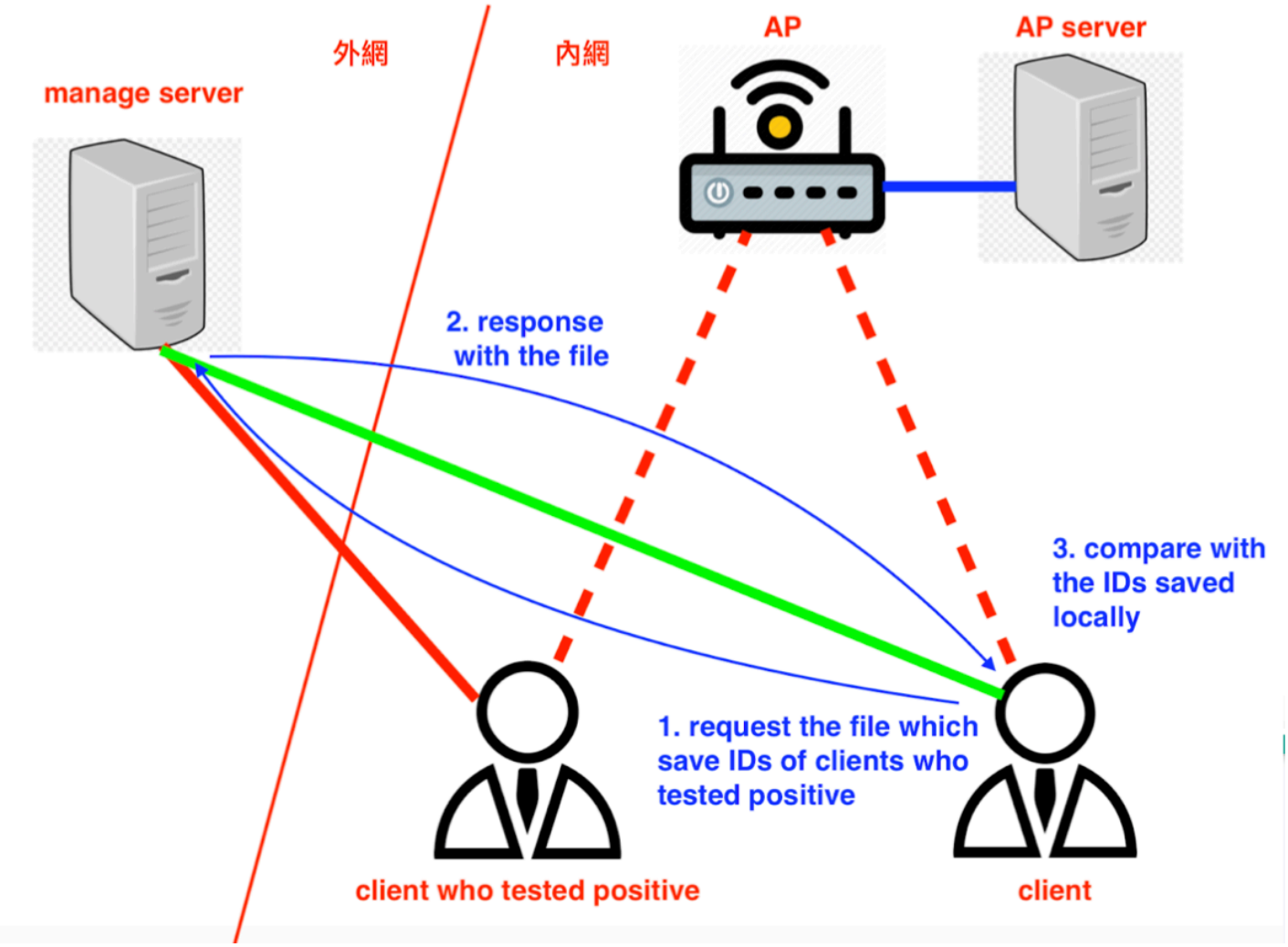
- 實作：
  這部分的實作是以 http 完成，為了提升傳送 key 安全性，實際上線時必須以 https 來傳輸。



(d) 接觸史的確認

當用戶想要確認接觸史時，會向 manage server 發起 request ，獲取存放確診者使用 ID 的檔案，並和本地因為連上 AP 所記錄到的 ID 進行比對。
倘若發現相同的 ID ，即表示與確診者在過去 14 天內有過接觸，應該立即反應。

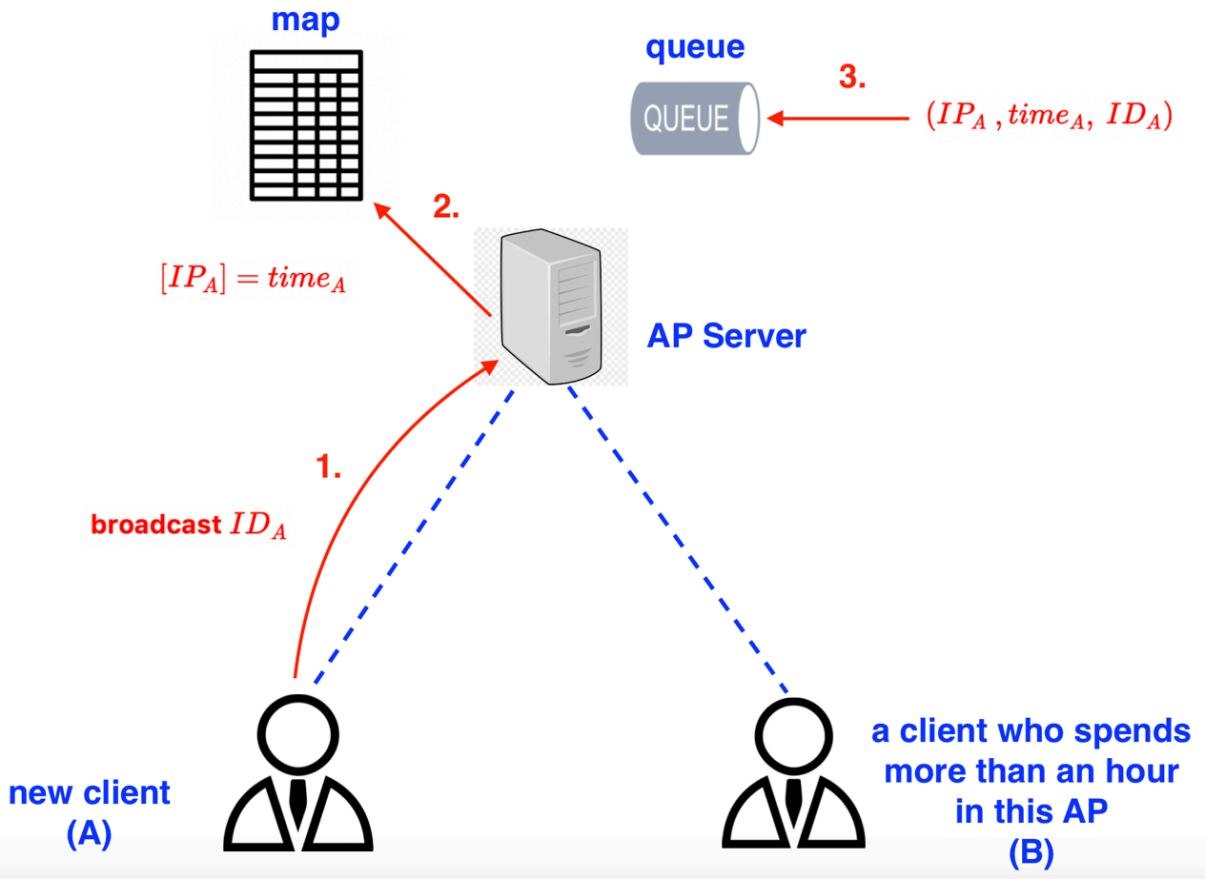
- 實作：
  這部分的實作是以 http 完成。因為有從 server 端獲取檔案的程序，類似 file server ，因此以 http 實作。



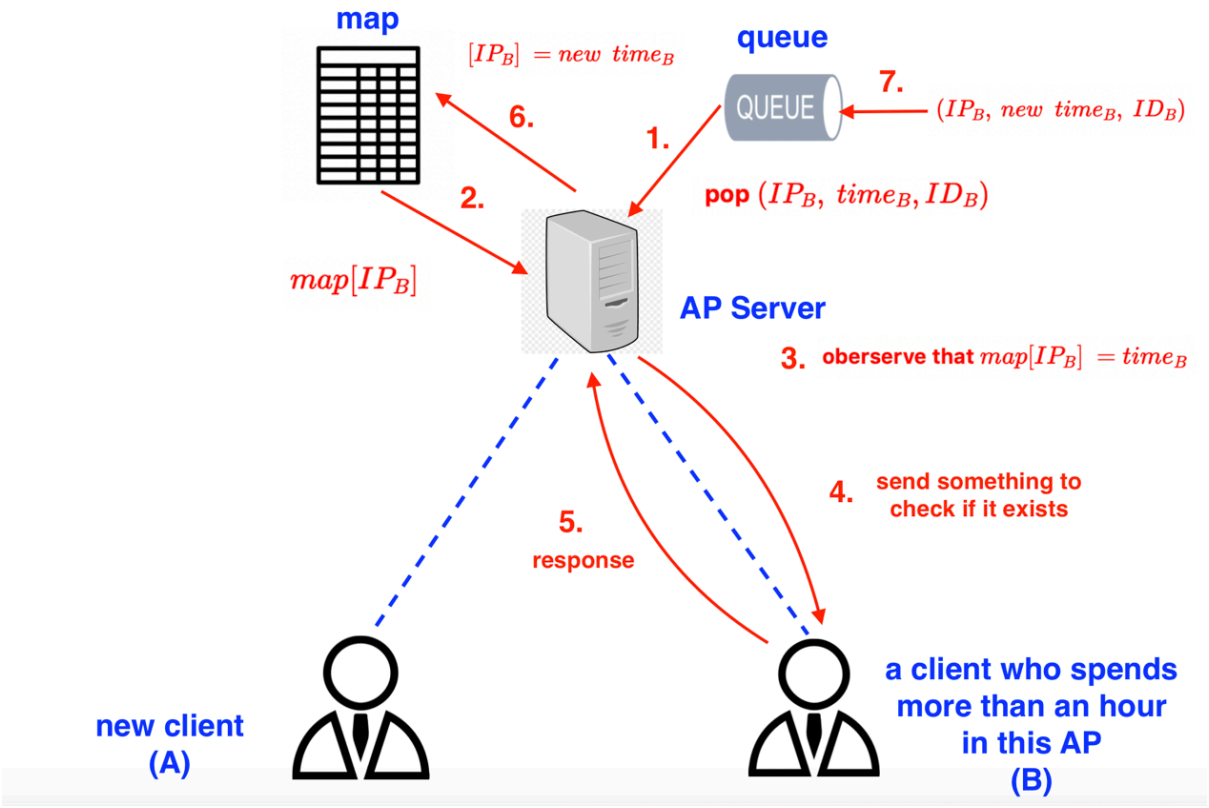
4. 實作上的困難與解決方法

- **AP server 的 IP 位址不固定**
  - 因為 AP 採用 dhcp 來分配 IP，所以當有新的連線者進入 AP 網域時，不能確定 AP server 的 IP 位置。
  - 解決辦法 :
    將 server listen 在特定的 port ，由新的 client 進入時，broadcast 自己的 IP 以及 ID，來找到 AP server。
- **TCP 無法 broadcast**
  - http(TCP)作為連線導向的協定，並不支持 broadcast 的操作。
  - 解決辦法 :
    使用 udp socket 作為替代方案。 只是 socket 較為底層，訊息收發的時機自行定義，確定好 client 和 AP server 之間的動作先後順序，才能確保系統正確運作。
- **出現連線超過一小時的使用者**

- 上面提到 AP server 必須提供新的使用者連線時回推一個小時內的所有 ID。所以 AP server 會丟棄被紀錄超過一小時的 ID 。
- 如果 AP server 紀錄 ID 的時間超過一小時，提供這個 ID 的使用者卻還連線在 AP 上的情況發生的話，這個 ID 不應該被丟棄，應該一併傳送給新的使用者。
- AP 採用的 dhcp 方式，可能讓不同的使用者拿到相同的 IP 。
- **解決辦法** :
  - client 在連線上 AP 時，必須提供 ID 以及 IP 位置。
  - AP servr 除了紀錄 ID 被記錄到的時間外，也需要額外紀錄這個 ID 的來源 IP 位置。
  - 並且使用一個 dictionary(map) 紀錄「來源 IP 」和「該 IP 最新一次提供的 ID 被記錄下來的時間」之間的對應。
  - server 在丟棄某個 ID 之前，先檢查「這個 ID 被記錄的時間」 以及 「提供這個 ID 的 IP 最新一次提供的 ID 被記錄下來的時間」是否相等，如果不同就表示提供該 ID 的使用者已經離開 AP，可以丟棄這個 ID。
  - 如果相等的話，表示提供該 ID 的使用者可能還在 AP 內。利用 socket 或是 `arp -a` 指令等等方式，確認這個 IP 是否存在於內網，如果不存在，則可以安心丟棄這個 ID。
- 流程
  - AP server 紀錄新連線的使用者



- AP Server 檢查是否有 ID 需要丟棄，且不該丟棄的情況



5. Future Works

- **Reliable UDP socket 或是 broadcast 的取代方法**
  - TCP 雖然不能做到 broadcast ，但是提供了 reliable 的傳輸，另外也有 congestion 的 control ，如果選用 UDP 的話就必須再上層自行實作這些功能。
  - 另外一個避開 udp 的方法，是 AP server 定期掃描網域內的 IP ，或是從 AP admin 處獲取訊息。這種方式需要 server 主動的掃描，成本較高，也可能造成內網的封包較多的情況。
- **防止惡意上傳 key**
  - 在這次的 project 中，並沒有對上傳 key 的動作進行限制，但惡意上傳 key 是這個系統的一大弱點。
  - 可以使用一次性的帳號密碼，來確保使用者上傳 key 的動作是受到許可的。
- **強制連線者上傳 ID**
  - 如果害怕普及率不足的話，也可以在 AP server 裡限制只有上傳 ID 的使用者才能和 key 進行連線。
  - 這個部分需要 AP admin 的權限才能操作。

6. 参考連結

- https://docs.python.org/3/library/secrets.html (https://docs.python.org/3/library/secrets.html)
- https://docs.python.org/3/library/hmac.html (https://docs.python.org/3/library/hmac.html)