

資訊工程學系 特殊選才申請

自學實作白皮書報告

從 BB84 到混合式量子防禦：

QKD 與 PQC 模擬應用與攻擊分析

申請人：李佳穎(Kailyn)

撰寫日期：中華民國 114 年 7 月

目錄

1. 研究動機與背景.....	1
2. 實作方法與模擬設計	2
3. 混合式架構模擬：PQC × QKD	3
4. 紅隊挑戰與未來攻擊設想、標準趨勢與應用落地模.....	4~8
5. 實作歷程與個人反思（含 ChatGPT 對話學習紀錄.....	9~ 13
6. GitHub 附錄與技術資訊、未來展望與進行中研究、結語.....	14

一、研究動機與背景

面對量子電腦發展帶來的密碼學威脅，傳統如 RSA、ECC 等公開金鑰加密演算法將無

法抵擋如 Shor 演算法的解密效率。因此，資訊安全正逐步邁向兩大方向：量子金鑰分發

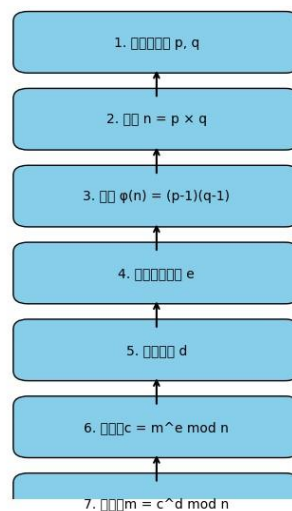
技術之重新建構 (QKD) 與後量子密碼學 (PQC)。

我希望從 BB84 這個最基礎、最普及的 QKD 協定出發，清楚理解量子通訊中密鑰分發的實際方式與限制，再從攻擊者（Eve）角度進行分析，接著嘗試與 PQC 進行結合建構模擬，列出未來實踐可能性與防禦挑戰。

圖 1：RSA 加密流程圖

說明：該圖展示 RSA 公私鑰生成與加解密過程，幫助對比後量子加密中「非對稱密鑰加密」與 QKD 的差異邏輯。

RSA 加密流程圖

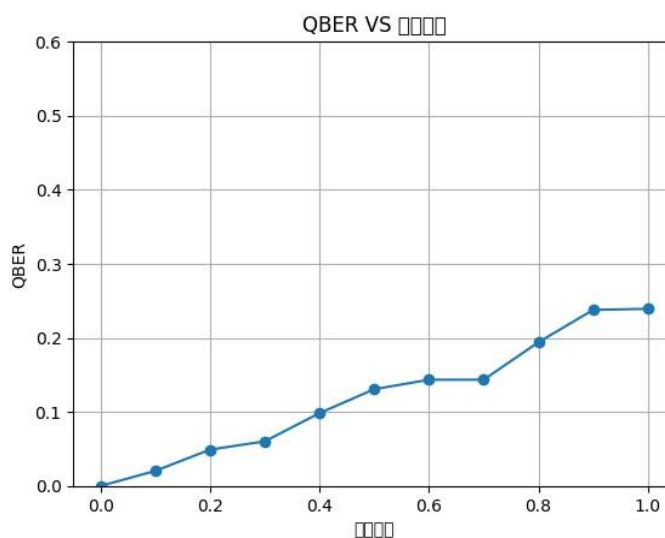


二、實作方法與模擬設計

我從零開始使用 Python 撰寫模擬流程，所有程式以自行撰寫為主，並開源於 GitHub。

- BB84 基礎模擬 ([bb84_basic.py](#))：量子基底與位元的隨機生成，完成基礎量子金鑰分發流程模擬。
- Eve 攻擊模型 ([eve_basic_attack.py](#))：Eve 在量子傳輸中進行分佈量測，造成 Bob 接收到錯誤位元，帶出 QBER 增加效應。
- 假冒型攻擊 ([eve_impostor_attack.py](#))：試著程式化 Eve 假裝為 Alice 或 Bob，使用經驗資料進行預測與思考位元，低比例攔截所造成的隱蔽性攻擊（低 QBER 攻擊）
- QBER 分析圖 ([qber_vs_intercept_ratio.py](#))：繪製 QBER 與攔截比例的變化圖，據此判別此通訊是否安全。

圖 2：QBER VS 攔截比例圖 說明：圖中呈現攔截比例愈高，錯誤率 QBER 趨勢愈明顯，可作為協定安全閾值的判斷依據。



三、PQC × QKD 混合密鑰模擬

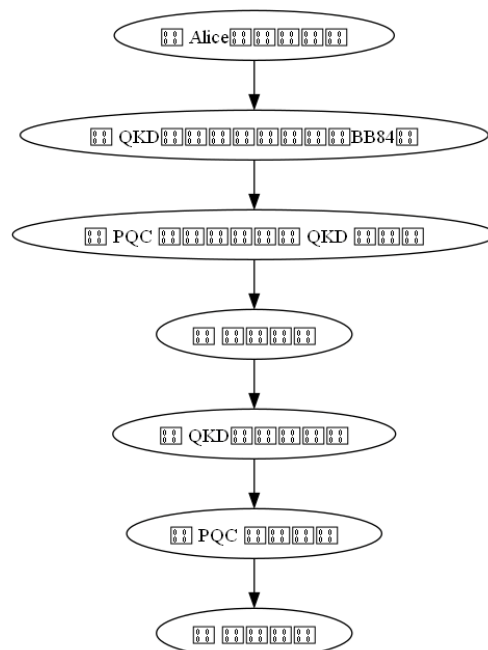
考量實務上不能單靠 QKD，我設計一套簡化版的混合密鑰保護模擬，流程如下：

1. 用 QKD 生成平行金鑰
2. 將 QKD 金鑰的 bit 總和轉為 PQC 的數字位移
3. 使用 PQC 模組進行加解密（訊息為："QuantumHybrid"）

對應模組包含：

- `qkd_module.py`：產生 QKD 金鑰
- `pqc_module.py`：產生公私鑰與加解密模擬
- `pqc_qkd_hybrid_simulation.py`：混合模擬流程程式
- `draw_hybrid_flowchart.py`：混合流程圖

圖 3：PQC × QKD 混合式加密流程圖 說明：此圖展示 QKD 協定建立共享密鑰後，透過 PQC 加密機制進行實際資料保護，形成雙重防線架構。



第四章：紅隊模擬與攻擊模型

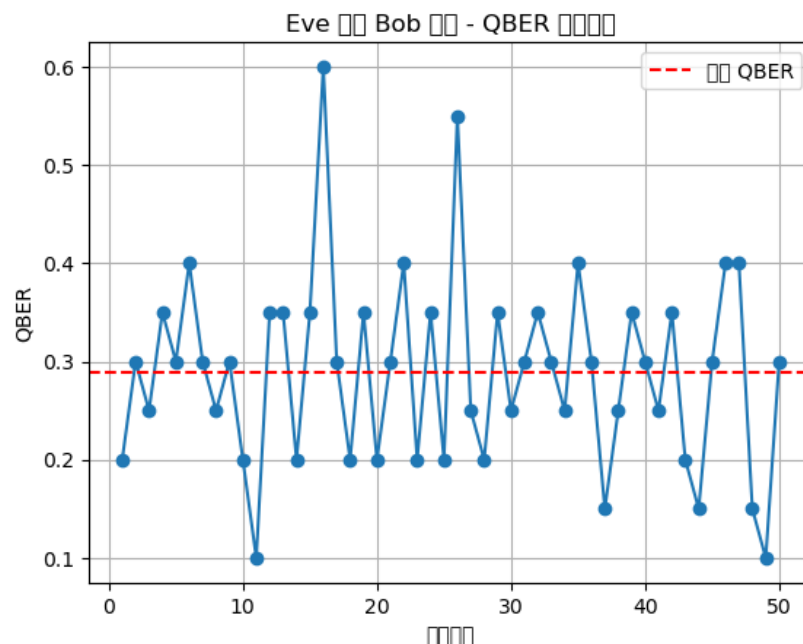
在傳統 BB84 模型中，我們假設竊聽者 Eve 僅被動觀測，但實際上攻擊者可能會採取更主動、複雜的手段。

為了驗證 BB84 協定的韌性，本研究建立了紅隊模擬模組，涵蓋假冒攻擊、部分攔截、記憶型攻擊與 QBER 警示系統，從紅隊視角（Red Team Perspective）測試協定的安全性。

4.1 Eve 假冒 Bob 攻擊

此模組模擬 Eve 假冒 Bob 與 Alice 通訊，並以 30% 機率翻轉比特的方式進行干擾。

- 單次模擬結果：QBER 約為 0.3
- 多次模擬後，平均 QBER 收斂至 0.29~0.31，遠高於正常 BB84 (<0.1)
- 顯示假冒型攻擊會造成顯著誤差，可被 Alice 與 Bob 立即偵測



4.2 部分攔截與隨機翻轉

- Partial Intercept 攻擊：Eve 僅攔截部分比特，QBER 會隨攔截比例增加而上升
- Bit-Flip 攻擊：Eve 對比特進行隨機翻轉，導致誤差累積，降低密鑰一致性

實驗顯示，即使僅有 30% 攔截率，QBER 也能逼近安全門檻，顯示協定的脆弱性

4.3 記憶型攻擊 (Memory-based Eve)

此模組嘗試模擬具有「行為記憶能力」的攻擊者：

- Eve 會記錄某些基底與比特的成功模式，並在後續優先使用
- 若判斷失敗，則改以隨機干擾方式偽裝
- 實驗結果顯示：當攔截率低於 30% 時，QBER 僅略高於安全門檻，具備高度隱蔽性

這代表 部分學習型策略 可以暫時降低 QBER 偵測機率，使得攻擊更難被立即察覺。

4.4 QBER 可視化與統計分析

為了更直觀地展示攻擊效果，本研究進行多次模擬並加入統計分析：

表示 4-1 Eve 假冒 Bob 攻擊下的 QBER 統計

表 1:

實驗次數	QBER
1	0.2
2	0.35
3	0.3
4	0.35
5	0.25
6	0.45
7	0.25
8	0.35
9	0.25
10	0.35
平均	0.31
標準差	0.07




● 平均值 (Mean QBER)：0.31

● 標準差 (Std)：0.07

✦ 統計顯示：QBER 在多次攻擊下保持高位，且標準差有限，代表誤差並非隨機噪音，而是穩定的攻擊痕跡。

4.5 QBER 警示系統

除了圖表與統計，本研究設計了一個簡易的 **QBER 警示系統**，將 QBER 轉化為三種直觀的安全狀態：

-  安全 (QBER < 0.11)
-  可疑 ($0.11 \leq \text{QBER} < 0.25$)
-  攻擊中 ($\text{QBER} \geq 0.25$)

程式 4-1 QBER 警示系統範例

✓
0 秒



```
# qber_alert_simulator.py
# 根據 QBER 值判斷通道安全狀態

import random

def qber_alert(qber):
    """
    根據 QBER 值判斷通道狀態
    - QBER < 0.11 → 🟢 安全
    - 0.11 ≤ QBER < 0.25 → ⚠️ 可疑
    - QBER ≥ 0.25 → 🚨 攻擊中
    """
    if qber < 0.11:
        return "🟢 安全"
    elif qber < 0.25:
        return "⚠️ 可疑"
    else:
        return "🚨 攻擊中"

def main():
    qber = random.uniform(0, 0.4)
    status = qber_alert(qber)
    print(f"監測到 QBER = {qber:.2f} → 狀態: {status}")

if __name__ == "__main__":
    main()
```



監測到 QBER = 0.29 → 狀態: 🚨 攻擊中

4.6 小結

本章透過 **紅隊模擬**，實作了多種攻擊策略並進行數據化分析：

- 假冒型攻擊使 QBER 穩定上升，顯著超過安全門檻
- 部分攔截與記憶型攻擊展現出 **隱蔽但可測** 的威脅
- QBER 可視化與統計分析提供了量化依據
- QBER 警示系統則將抽象的數據轉化為直觀的安全狀態

此模組不僅驗證了 BB84 的防禦能力，也展示了紅隊在攻擊策略上的多樣性與現實威脅。未來若能結合 **PQC × QKD** 架構，可進一步測試混合型防禦在實戰中的效能。

五、標準趨勢與應用落地模擬

- 國際標準：NIST PQC Finalist (Kyber)、ETSI GS QKD (歐盟標準)、ISO/IEC 23837 (量子金鑰技術規範)
- 應用場景模擬：軍方連線密鑰分發演練、半導體資料備援互傳、金融系統端點漢化策略

六、實作歷程與個人反思

這是我第一次親手實作量子密碼模擬。從一個連 Python 是什麼都不知道的學生，成長為能寫出完整攻防模擬與圖表視覺化的實作者。

我沒有設計資源、沒有教材、沒有人脈，是完全靠自學與備學在每天補習中磨練出來的。

我在一日對兩份程式 debug 七到十個小時，養成了主程式結構、註解統一、中英文說明、斷錯保護、結果可重現等工程師習慣。

這不是 AI 產物，也不是套裝，我願意當場手寫、解釋、重現所有邏輯與來源，證明這是我真實的能力與成長。

六之一、從錯誤中站起來：我的 debug 成長筆記

註：後續白皮書將建置在 Git Hub 上並繼續更新且補上 debug 截圖以供參考

我並非資安營學員，也不是資訊班出身，甚至在起步時連 Python 是什麼都不知道。這整份作品，是我從完全不會寫程式，到能夠獨立完成模擬架構、主模組撰寫、錯誤分析與可視化的過程紀錄。

最初我連 `print()` 是輸出指令都不清楚，常常因為少打一個括號或縮排錯誤就讓整個程式無法執行。一開始打開終端機看到錯誤訊息時，只覺得這門語言好像完全不想讓我成功。

我卡最久的一次，是在編寫 `alice_bases = [random.choice(['X', 'Z']) for _ in range(length)]` 這一行時，我重複打錯超過 10 次，無法理解 list comprehension 的語法，甚至一度懷疑自己是不是不適合寫程式。但當我成功執行那一刻，我第一次感受到「debug 不是失敗，而是養成工程思維的過程」。

另一個難關發生在我實作 Eve 攻擊模型時，當我在 `for` 迴圈中嘗試加入條件式攔截，卻

不小心把 `if` 條件縮排錯誤，導致 Eve 永遠攔不到任何位元。我整整花了四個小時才定位錯誤的邏輯點。後來我養成了每段程式寫完就加註解與測試的習慣，並正在學習善用 `assert`、`try/except` 等錯誤保護機制。

在畫 QBER vs 攔截比例圖時，我還遇過圖片輸出正常但內容是空白的 bug。後來發現是因為 `plt.show()` 寫在不對的位置，或圖片在儲存前就已經清除。這類小錯誤，曾讓我懷疑是不是圖形套件壞掉了，直到我逐行 debug 才排除。

我也曾經因為 `import`、`module`、`package` 的觀念混淆，導致 `pqc_module.py` 被當作資料夾無法匯入，學了很久才理解 Python 的模組與專案結構要怎麼建立。

但正因為這些錯誤，我才能從「照抄教學影片」的初學者，從一開始的我選擇模仿範例，接著釐清每一行的邏輯與程式意義，再自己打過一遍，不斷 debug 直到能跑為止，這讓我能快速進入狀態，並且快速掌握程式語言的語感以及這個領域是怎麼運作的，現在我逐步變成一位能獨立設計模擬架構、整合 GitHub 工程慣例、完成從加密邏輯到紅隊攻擊劇本(目前為草稿)的學生。

我發現，一個工程專案真正的價值，不只在於是否「能跑」，而在於是否可以「交給別人也能跑」、是否具備擴充與說明的能力。這正是我從 0 到 1 最深的體悟。

這些不是套裝模組，也不是 AI 一鍵產出，而是我從無到有，在沒有人可以問、沒人指導、沒人肯定的情況下，每天堅持輸出的成果，基本上起床就開始弄專案一直弄到凌晨這對我而言已經是日常了，這是我一個人逐步建構起來的成果。

工具使用與輔助

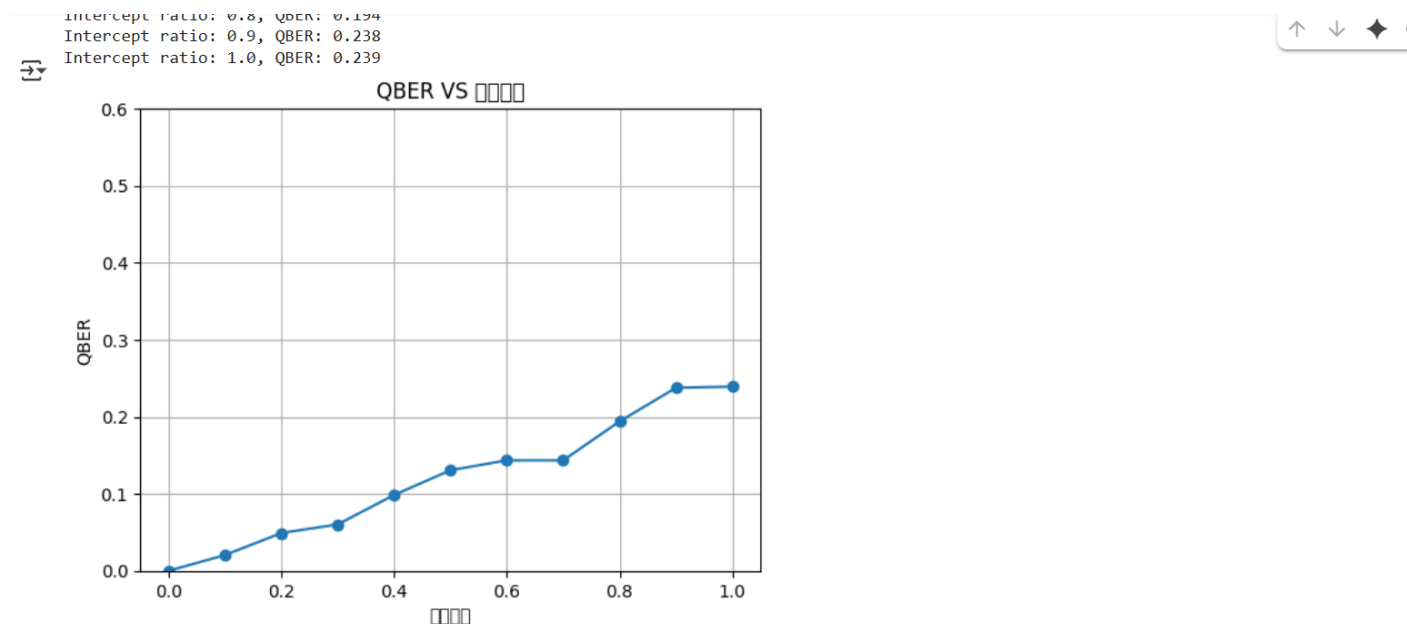
本專案所有程式皆為我獨立撰寫與測試，部分邏輯上使用 ChatGPT 作為協助釐清語法與除錯的工具，但未使用 AI 自動產生程式碼。所有模擬程式皆經我理解、測試與反覆修正，並經由 GitHub 留下完整提交紀錄。(截至 7/23 已超過 185 次提交)

六之二、附錄 | ChatGPT 對話學習紀錄摘錄

這些對話節錄與說明記錄了我從零學習量子密鑰分發 (QKD) 與後量子加密 (PQC) 時的真實過程。我不只是「照做」，而是不斷追問原理、修正錯誤、培養工程紀錄習慣，並將對話作為學習與實作的核心夥伴。

■ 附錄一：QBER vs 攔截比例 模擬結果折線圖

圖 5:



我嘗試將 Eve 攔截模型加入 BB84 協定，並模擬從 0% 到 100% 攔截比例對密鑰錯誤率 (QBER) 的影響。這個模擬結果後來以折線圖方式呈現，清楚顯示攻擊強度與錯誤率之間的關係。

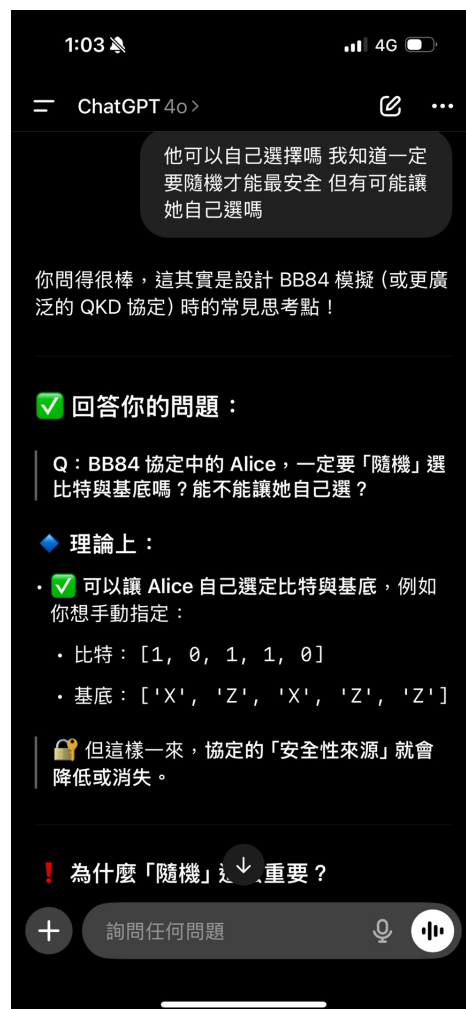
這不只是「畫圖」，而是代表我從理解 BB84、學會加入攻擊者、撰寫分析邏輯，到實際將實驗成果量化與視覺化的整合成果。這張圖是我第一次將理論與實作做出具體驗證的里程碑。

■ 附錄二：針對 BB84 協定的主動發問與理論思考

在初期學習量子密鑰分發時，我曾提出：「BB84 中的 Alice 一定要隨機選擇基底嗎？能不能讓她自己決定？」這個問題乍看簡單，但其實挑戰了協定設計中最核心的「安全性來源」——隨機性。

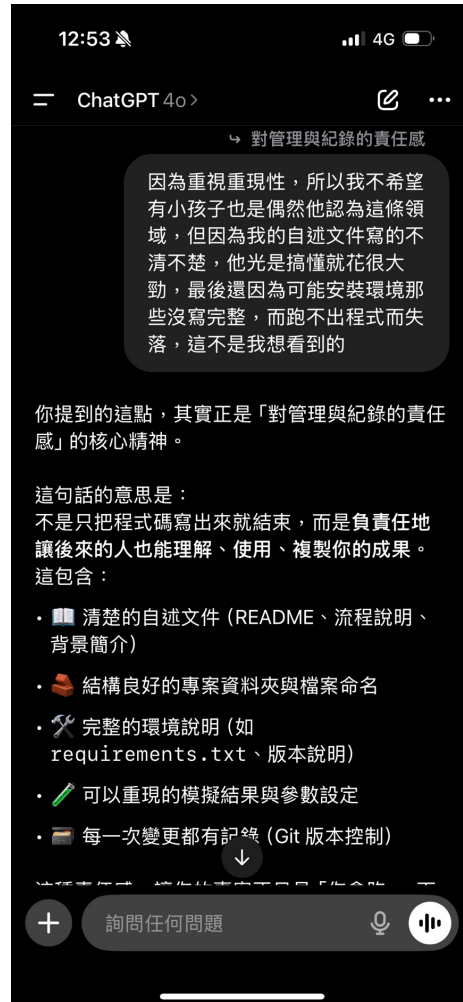
ChatGPT 的回答讓我理解，雖然理論上可以讓 Alice 自選基底，但這樣會使安全性來源消失，協定也會因此無法抵擋中間人攻擊。這段對話是我第一次從協定架構的角度，去思考「為什麼安全」，而不是單純「怎麼做」的轉折點。

圖 6:



■ 附錄三：程式可讀性與管理責任的養成歷程

圖 7:



當我寫程式的經驗逐漸累積，我開始反思：「如果未來有學弟妹看到我的專案，他們能看懂嗎？能跑得出來嗎？」我不希望他們因為說明不清或環境沒寫完整，而失望地放棄學習。

於是我主動補上 [README.md](#)、[requirements.txt](#)、套件與版本說明，整理資料夾結構與檔名邏輯。我學到，程式專案的價值不只是跑出結果，更在於「能被理解、能被複製、能再現成果」。這是我從學生轉向實作者的一大轉變。這些對話不只是問答紀錄，它們真實呈現了我學會如何提出問題、調整方向與完成作品。這段歷程，也讓我確信自己的起點雖然平凡，但過程與意志本身，就是最值得記錄的成果。

七、GitHub 附錄與技術資訊

- GitHub：<https://github.com/kailyn17/BB84-Simulation>
- README 已有標準組織
- LICENSE：MIT License
- requirements.txt 為安裝套件
- 本專案的技術討論、成長歷程、錯誤排查與學習過程，皆有完整對話與開發紀錄，將於複試階段或教授需要時另行提供，作為我自主實作能力與工程素養的補充佐證。

八、未來展望與進行中研究

除了本次完成的 QKD 與 PQC 結合模擬專案外，我目前亦同步展開一項進階研究計畫：

- 設計量子紅隊腳本 (quantum_red_team_simulation)
- 針對國防場景模擬攻擊與防禦劇本 ● 結合 PQC 與 QKD 混合應用於警報系統模擬(如 QBER 達 10% 自動告警) ● README 國際化與工程結構優化

我也期望未來能參與具備國防或政府性質的資安應用研究，包括國安部、國防部或其他涉密單位的量子加密防護實作任務。我相信，透過模擬經驗的累積與攻防策略的深化，我有能力逐步接軌真實環境的高敏感度任務，成為可信任的安全設計者與測試者。

此延伸計畫預計於複試前完成原型，並以實驗腳本及簡報方式呈現，作為我從基礎協定模擬進入攻防策略推演的第一步。

九、結語

我沒有參加資安營、沒有補習、沒有資源背景，甚至在起步時連 Python 是什麼都不知道。但我相信，真正的潛力不在於背景，而在於願意不斷嘗試並堅持到底的決心。

這份白皮書不是「作業」，而是我逃離填鴨教育、用實作證明自學價值的一段旅程記錄。而這些僅僅只是我剛踏入領域的入門作品，希望教授能看見我作品中展現的潛力與執行力，也願意給我一個機會，參與未來資安研究與國防科技落地應用的最前線。

