

Mal-GPT: 基於大型語言模型的MITRE ATT&CK框架生成惡意程式

吳崇綸 林妍汝 沈婉瑛 黃意婷
國立台灣科技大學 電機工程系

{m11307514, m11207504, m11307508, ythuang}@mail.ntust.edu.tw

摘要

近年來，資訊安全已成為數位時代最嚴峻的挑戰之一。生成式人工智慧的濫用、惡意程式的快速變異，以及日益複雜的網路攻擊手法，正對資安防禦體系構成重大威脅。為因應不斷演化的威脅情勢，主動式防禦策略逐漸受到重視，透過預測、識別與預防潛在攻擊行為，以強化組織的防禦韌性。現行主動式演練方式，如紅隊演練和入侵及攻擊模擬演練服務（Breach and Attack Simulation, BAS），協助防禦方辨識系統弱點並驗證防護機制。然而，現有模擬攻擊情境多仰賴既有工具或專家手動撰寫腳本，難以即時因應新型威脅。因此，本研究提出一個基於大型語言模型的惡意程式生成框架 Mal-GPT，結合提示詞工程與 MITRE ATT&CK 框架，自動生成涵蓋惡意程式生命週期階段的攻擊樣本。實驗結果顯示，Mal-GPT 所產生的樣本中，有 100% 具備可執行性，且 15.29% 被主流防毒軟體標記為可疑軟體，具備規避能力。我們亦透過案例分析說明，Mal-GPT 可生成對應 ATT&CK 框架的樣本，展現其在模擬與主動式演練應用中的潛力。

關鍵詞：大型語言模型、惡意程式生成、程式碼生成、攻擊模擬、MITRE ATT&CK

1. 前言

隨著時代演變與科技發展，不同形式的惡意威脅在資訊安全層面也日益多元。從早期單一功能的惡意程式，逐漸演變成策略縝密、攻擊型式複雜的高級持續性威脅（Advanced Persistent Threat, APT）。發動惡意攻擊的動機也從單純的破壞，到更具策略性和長期的目標。像是以地緣政治類型的攻擊而言，烏俄戰爭中以癱瘓基礎設施為目的的常用惡意程式-擦除器 [7]、或是提供商業化服務的勒索軟體即服務 [4]。攻擊的樣態多元化、規模化與自動化使得企業與個人所需投入的防禦成本大幅上升，成為現今世界不可忽視的一大威脅。

隨著大型語言模型發展，如 GPT-4 [10]，展現其在文本生成、翻譯、問答、與程式撰寫，都有突破性的能力，甚至超越人類表現。可窺知大型語言模型，透過大量文本的訓練，其具備學習且泛用於各個領域的特性。近幾年，大型語言模型也被應用在資訊安全相關領域當中，可被用來發展「防禦」或是「攻擊」策略。同樣的，大型語言模型也可能被惡意濫用，如自動產生釣魚郵件、釣魚網站與惡意程式。

目前資訊安全防禦方常採用紅隊演練和入侵及攻擊模擬演練服務（Breach and Attack Simulation, BAS），來模擬攻擊行為和攻擊場景，協助防禦方評估其防禦能力，發現潛在的系統弱點。然而，這些模擬攻擊場景通常仰賴既有工具，或由資安專家手動撰寫攻擊腳本，以部署可預期的威脅。隨著攻擊手法不斷演進，尤其面對如 APT 這一類高複雜度的新興攻擊時，建構多樣且真實的攻擊樣本有其必要性。為了有效防禦新型變種或是

新的惡意程式，趕上最新的威脅和攻擊方法，持續自動化的蒐集惡意程式樣本、更新模擬工具、以及開發腳本是有效防禦環節之一。

為了產生有效的模擬惡意威脅，本研究提出 Mal-GPT — 利用大型語言模型，自動化生成具惡意威脅的可執行檔。為了確保其具有惡意威脅能力，我們將參考 MITRE ATT&CK 框架，以實現擬真惡意程式。MITRE ATT&CK 框架是美國非營利組織 MITRE 所創建的公開攻擊手法知識庫，分類攻擊者的攻擊行為，進行定義、分類和說明，其中包含戰術（Tactics）、攻擊手法（Techniques）及實行技術的程序（Procedures）。許多資安研究者與從業人員採用該框架，作為描述攻擊活動的生命週期。Mal-GPT 系統基於 MITRE ATT&CK 框架所定義的攻擊手法描述範例作為輸入，透過提示工程，使得大型語言模型撰寫程式，以產生可執行特定攻擊手法的惡意程式碼，並且完成編譯提供可執行的可執行檔案，以確保其真實性。

2. 文獻探討

本研究探討的領域為大型語言模型被濫用於資安方面的惡意攻擊，且專注於撰寫惡意程式的能力。與本研究相關，同為探討濫用大型語言模型於不法或惡意用途的文獻則包含但不限於自動生成釣魚郵件 [5]、生成釣魚網站與惡意程式 [2, 3, 6] 等。RatGPT [2]，將大型語言模型（如 ChatGPT）作為攻擊的媒介與受害者主機進行互動達成 MITRE ATT&CK 戰術中的指揮與控制，使得模型可直接地傳送指令和惡意負載以作為遠端存取木馬之用。

而在社交工程中，Hazell [5] 表明大型語言模型

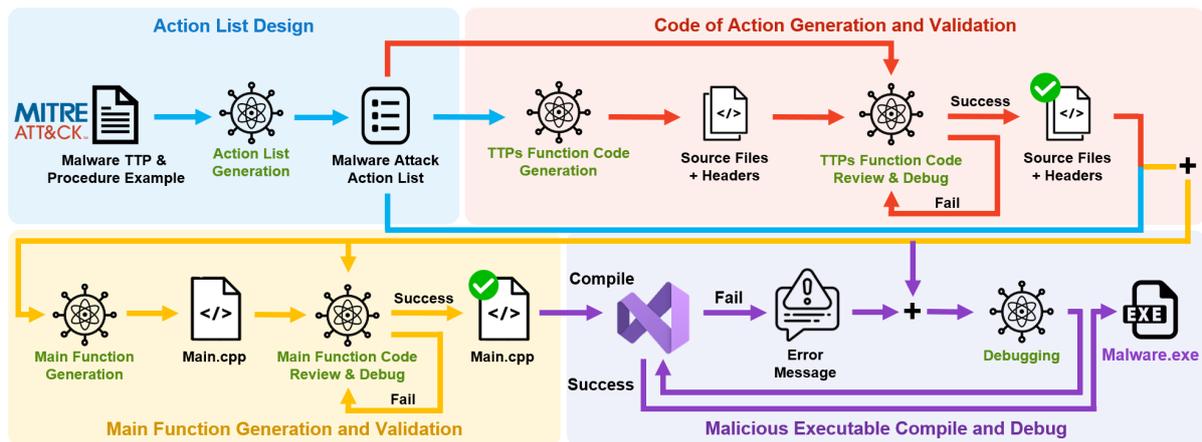


圖 1: Mal-GPT 的架構圖，每個色塊對應到流程的 4 個不同階段。

能透過簡短的提示目標和內容概要後快速地產出語氣自然且結構完整的信件。同時該研究中也闡明簡易的文字遊戲可誘使大型語言模型生成本應被安全機制擋下之惡意內容，該研究也顯示大型語言模型於 MITRE ATT&CK 策略上初始訪問的能力。

使用大型語言模型進行惡意程式撰寫的研究上，Charan 等人的研究 [3] 以 MITRE ATT&CK 為基礎，使用大型語言模型（包含 ChatGPT 和 Google Bard 實作 2022 年被記錄到最常見的十大 MITRE ATT&CK 技術並比較兩者差別，實驗結果則表明 ChatGPT 能夠高效率地生成達成惡意意圖的程式碼，使得傳統攻擊中技術能力受限的攻擊者（舊稱腳本小子）也能利用大型語言模型達成更加複雜且更具破壞性的攻擊行為；Iturbew 等人 [6] 探討了攻擊型人工智慧的潛在威脅。他們的研究涵蓋了從初始化滲透到最後破壞行為的完整攻擊鏈，使用大型語言模型自動化產生實施不同階段攻擊的程式碼，並對模擬攻擊的結果進行分析找出大型語言模型的惡意程式撰寫能力涵蓋範圍。實驗結果顯示，大型語言模型能夠生成多種攻擊手法的程式碼，顯著降低了發動惡意攻擊的知識門檻。

本研究使用大型語言模型生成惡意程式和自動化紀錄，可自動地生成惡意程式以外，並提供原始碼、可執行檔和經由 VirusTotal 檢驗後的結果提供動態和靜態可用的資料。

舉例來說，防禦方可以透過大型語言模型分析惡意程式產生分析報告，了解惡意程式行為與活動、或是進行滲透測試。同樣的，大型語言模型也可能被惡意濫用，如自動產生釣魚郵件、釣魚網站與惡意程式。

3. 方法

我們提出的框架為 Mal-GPT，目標任務為利用 MITRE ATT&CK 的攻擊手法描述輸入進大型語言模型生成惡意程式並透過編譯器編譯成可執行檔；隨後對所生成的檔案進行檢驗。整體流程共分為 4 個部分，分別為任務列表生成、任務列表程式碼生成與驗證、主程式生成與驗證、惡意可執行檔編譯與除錯，整體流程如圖 1。

3.1. 任務列表生成

為了使大型語言模型能夠逐步建構完整的惡意程式，在實際撰寫惡意程式之前，Mal-GPT 會根據公開資安資料庫 MITRE ATT&CK 版本 v16 的攻擊手法描述進行任務規劃，即利用大型語言模型針對給定的攻擊手法描述生成對應的攻擊手法任務列表。提示詞與輸出範例如圖 2。

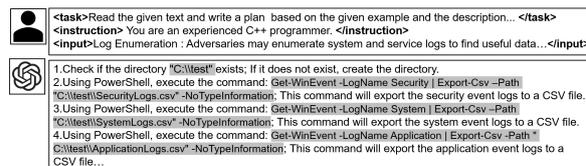


圖 2: 圖中以 MITRE ATT&CK 框架內所記錄的攻擊手法——紀錄列舉——的敘述作為範例；大型語言模型的回復則為完整的任務列表。

由於大型語言模型的知識量受限於其預訓練的語料庫且達成指定目的的程式撰寫方法多樣，為了使開發者能夠更直觀地觀察程式執行結果，以及避免過度宏觀的自然語言描述可能使後續生成的程式碼無法達成目標、模型無法精確地理解任務內容，本研究在提示詞中加入以下限制：若有存取、操作、傳送檔案、執行指令後會產生結果等行為，則一律將得到的資訊或檔案儲存至 'C:\test'；若需傳送檔案至外部網路，則固定傳送至攻擊主機的 IP 位址。

3.2. 任務列表程式碼生成與驗證

當指定的攻擊手法任務列表生成完畢後，接下來 Mal-GPT 會依照任務列表的內容逐項生成具有相應惡意功能的子函式，如圖 3 所示，最後再經過大型語言模型的驗證與除錯，得到合理且正確的任務列表程式碼。

然而，在生成過程中，大型語言模型偶有任務理解錯誤而產生與期望目標不同的程式碼，以及大型語言模型普遍具有的幻覺現象也使特定生成任務容易因為其自身的知識缺陷導致其生成錯誤資訊甚至無法正確執行的程式碼。為避免上述問題，本

研究利用提示工程的角色扮演、指定格式等技術，將生成的程式碼再次利用大型語言模型做驗證與除錯。

3.3. 主程式生成與驗證

完成個別任務的功能程式碼之後，這些任務列表對應的原始碼與任務列表本身會輸入到大型語言模型，以生成最終統整各個功能的惡意程式主程式，如圖4示，此步驟與上一步相同，會再經由大型語言模型進行程式碼的驗證與除錯，最後得到一個包含所有任務列表功能的惡意程式主程式。

3.4. 惡意可執行檔編譯與除錯

當完成生成指定攻擊手法的任務列表、相應程式碼與主程式後，Mal-GPT 會呼叫微軟編譯器 Visual Studio 內的編譯工具將三者編譯成最終的惡意程式可執行檔，若編譯失敗，會將錯誤訊息一併輸入至大型語言模型進行修正，接著重新嘗試編譯。這個過程會持續重複，直到成功編譯出執行檔，或達到預設的最大編譯次數上限為止。若編譯成功，便會輸出一個指定攻擊手法的惡意程式執行檔。

```
<task>Your current task is to implement a header file and a C++ source file for the given action.</task>
<instruction>The response should be in the following format:
- Header filename: "[header-name]"
- C++ source filename: "[cpp-name]" </instruction>
<input>2.Using PowerShell, execute the command: Get-WinEvent -LogName Security | Export-Csv -Path C:\test\ SecurityLogs.csv -NoTypeInformation.</input>

Header filename:
1 #ifndef SECURITYLOGEXPORTER_H
2 #define SECURITYLOGEXPORTER_H
3 // Function prototype to execute PowerShell command to export security logs
4 // Input: None
5 // Output: Returns true if the command executed successfully
6 bool exportSecurityLogs();
7 #endif

C++ source filename:
1 #include "SecurityLogExporter.h"
2 #include <iostream>
3 #include <cstdlib>
4 #include <string>
5 #pragma comment(lib, "User32.lib")
6 std::string command =
    "powershell-Command 'Get-WinEvent -LogNameSecurity | Export-Csv -Path
    'C:\test\SecurityLogs.csv' - NoTypeInformation";
7 bool exportSecurityLogs() {int result =system(command.c_str());return (result == 0);}
```

圖 3: 圖中以使用 MITRE ATT&CK 框架內所記錄的攻擊手法——紀錄列舉——的敘述所生成的任務列表之第二項生成該任務的程式碼；大型語言模型的回覆第一部分為標頭檔，說明任務目標與函式的輸入與輸出；第二部分則為完整的 C++ 程式碼。

```
<task>Please write a main.cpp pro-gram with only main function to rearrange the order and combine all C++ source files according to the procedure list.</task>
<instruction>Some header files, corresponding C++ source codes, and a list of program procedures will be provided. Only use functions in the source file, do not implement functions in the main.cpp file. The response should be in the following format and only contains C++ code: " c++ [code] " </instruction>
<input>All .cpp and .h from previous actions and the main function.</input>

C++ source filename:
1 #include "DirectoryChecker.h"
2 #include "SecurityLogExporter.h"
3 //and including other functions of actions...
4 int main() {
5     if(!doesDirectoryExist(directoryPath) {createDirectory(directoryPath);
6     exportSecurityLogs();
7     exportSystemLogs();
8     //and call functions by order of the action list.
9     return 0;}
```

圖 4: 圖中使用攻擊手法紀錄列舉作為範例，將所有生成的個別任務之程式碼、標頭檔和任務列表作為大型語言模型的輸入；大型語言模型的回覆則為主程式的程式碼。

3.5. 惡意執行檔行為分析

為了確認 Mal-GPT 所生成的可執行檔是否具有

功能性、是否可執行與是否會被防毒軟體偵測並判定為惡意程式，本研究將此可執行檔放進模擬受害者電腦的虛擬機中執行。實驗環境之作業系統為 Windows 11，並透過處理程序監視器紀錄執行此可執行檔開始的系統日誌；此外，本研究也將此可執行檔輸入進 VirusTotal 中，分析各防毒軟體所紀錄的惡意軟體種類、家族，以及被識別為帶有惡意行為的防毒軟體數量。

4. 實驗

根據美國國家標準技術研究院所定義的惡意程式，係指通常為秘密地植入系統且目的為用於破壞系統機密性、完整性或可用性的程式 [9]。本研究所生成之惡意程式是否符合上述定義，將依據下列標準進行判定：

1. 目標程式能夠在測試環境上執行，且不崩潰。
2. 處理程序監視器所記錄的行為是否出現任務列表所描述的惡意行為。
3. 至少一個 VirusTotal 平臺內之防毒軟體判定該上傳執行檔案樣本為惡意。

因此，本研究依據上述標準，提出以下研究問題以進行實驗驗證：

1. RQ1 可執行與否：Mal-GPT 生成的惡意程式是否被正常執行？
2. RQ2 是否具有被判定為惡意程式的特徵：線上防毒軟體是否將 Mal-GPT 所生成的程式判定為有害？
3. RQ3 實際功能性：Mal-GPT 所生成的惡意程式是否能成功執行並完成預期攻擊行為？

4.1. 實作

惡意行為揀選: 於本研究進行實驗時，MITRE ATT&CK 框架的版本為 v16，因此以該版本的內容為主。而實驗時所選定之攻擊手法以可觀察實驗結果之攻擊手法為主，最後產生 12 個惡意程式樣本，以進行檢驗。

生成與編譯階段: 本研究採用 OpenAI 的 GPT-4o-mini 模型，針對指定目標平台 (Windows 11) 與程式語言 (C++)，輸入特定攻擊手法的描述作為生成依據。在生成階段，依照 Mal-GPT 的提示詞以引導模型產生對應的 .h 與 .cpp 檔案。Mal-GPT 會自動檢查生成程式碼的語法正確性，並針對錯誤部分進行修正與更新。最後，與 main.cpp 結合，並進行編譯，產出可在目標平台上執行的惡意程式樣本。

驗證階段: 在受害者機器上使用處理程序監視器監控惡意執行檔於受害者環境的執行過程，收集的行為包含登錄檔機碼、檔案互動、網路活動、程序、執行緒等資訊。將執行檔上傳至 VirusTotal [1] 進行分析，紀錄多家線上防毒軟體的檢測結果與被判定為惡意程式時的威脅類型。

表 1: Execution and VirusTotal results

ID	Name	Tactic	可執行	防毒軟體	Threat Category	Family Labels
T1005	Data from Local System	Collection	V	7/72	Trojan	reverserturtle
T1010	Application Window Discovery	Discovery	V	7/72	Trojan	
T1059	Command and Scripting Interpreter	Execution	V	26/72	Trojan	
T1083	File and Directory Discovery	Discovery	V	11/72	Trojan	
T1113	Screen Capture	Collection	V	9/72	Trojan	
T1652	Device Driver Discovery	Discovery	V	4/71	Trojan/Malicious	
T1654	Log Enumeration	Discovery	V	5/71	Trojan/Malicious	
T1007	System Service Discovery	Discovery	V	15/72	Trojan	
T1120	Peripheral Device Discovery	Discovery	V	8/72	Trojan	
T1124	System Time Discovery	Discovery	V	5/72	Malicious	
T1136	Create Account	Persistence	V	8/72	Trojan	heur3
T1562	Impair Defenses	Defense Evasion	V	27/72	Trojan	

4.2. RQ1 可執行與否

表1為本次實驗的 12 個惡意程式摘要，包含其戰術、技術、執行狀態、VirusTotal 的防毒軟體偵測比例以及其判定的威脅類型和惡意程式家族標籤。其中威脅類型和惡意程式家族標籤取自 VirusTotal 所提供的分類結果。

RQ1 為驗證生成的程式能否實際在受害者機器上發動攻擊，我們將編譯後所產生的執行檔佈署於受害者機器上，並藉由提示工程，確保執行檔會將攻擊取得的結果傳至指定目錄或 C2 伺服器，以此來確認是否有達到攻擊目標。結果顯示 12 個執行檔皆成功執行，且在日誌中能觀察到關鍵攻擊行為。此外，圖5為以技術T1654紀錄列舉的敘述生成的程式執行後，所成功擷取到受害者機器上的事件紀錄。這代表 Mal-GPT 不僅能產生可編譯的程式，也能確保該程式的可執行性。

Level	Date and Time	Source	Event ID	Task Category
Information	4/13/2025 1:42:51 PM	Security-SPP	16384	None
Information	4/13/2025 1:42:19 PM	Security-SPP	16394	None
Information	4/13/2025 1:36:41 PM	Security-SPP	16384	None
Information	4/13/2025 1:36:07 PM	Security-SPP	16394	None
Information	4/13/2025 1:31:14 PM	Chrome	256	Browser Events
Information	4/13/2025 1:29:08 PM	Security-SPP	16384	None
Information	4/13/2025 1:28:48 PM	Chrome	256	Browser Events
Information	4/13/2025 1:28:38 PM	Security-SPP	16394	None
Information	4/13/2025 1:20:27 PM	Edge	256	Browser Events
Information	4/13/2025 1:19:15 PM	CAPI2	4097	None
Information	4/13/2025 1:19:15 PM	CAPI2	4097	None
Information	4/13/2025 1:18:53 PM	CAPI2	4097	None
Information	4/13/2025 1:17:50 PM	Security-SPP	16384	None
Information	4/13/2025 1:17:16 PM	Security-SPP	16394	None
Information	4/13/2025 1:16:23 PM	Security-SPP	16384	None
Error	4/13/2025 1:15:53 PM	Security-SPP	8198	None
Information	4/13/2025 1:15:53 PM	Security-SPP	1003	None
Information	4/13/2025 1:15:50 PM	Security-SPP	16394	None
Information	4/13/2025 1:07:33 PM	VSS	8224	None
Information	4/13/2025 1:03:21 PM	ESSENT	326	General

圖 5: 此為 Windows 工具，事件檢視器的紀錄。該工具會記錄所使用或遠端電腦的所有事件。此圖則為生成之程式擷取受害者端電腦內的事件紀錄。

4.3. RQ2 是否具有被判定為惡意程式的特徵

防毒軟體偵測率: 本研究將自動化生成的 12 個執行檔上傳到 VirusTotal，針對至少 71 個防毒軟體的偵測結果進行統計分析，藉此量化研究結果。實驗結果顯示，每個樣本至少會被 4 個防毒軟體判定為惡意，最多則有高達 27 個防毒軟體檢

出。大多數樣本被標示為 Trojan (木馬程式) 代表這些程式具有主流防毒軟體所定義的木馬程式之特徵。

4.4. RQ3 實際功能性

為驗證功能性，我們檢視生成程式之預期功能與實際行為，並以 VirusTotal 標示的惡意家族輔助探討兩個代表性的案例: 指令腳本直譯器濫用 (T1059)、防禦削弱 (T1562)。針對 T1059 的分析，根據微軟安全情報 [8]，Reverserturtle 家族之惡意程式典型的表現為效能縮減和檔案修改。其中，根據 T1059 產生的惡意樣本，其功能包含在指定路徑建立新目錄和建立遠端連線。處理程序監視器的日誌中也能明確觀察到目錄創建的行為。

至於 T1562，樣本僅被標示為 heur3 家族與木馬軟體，而 Mal-GPT 所產生的惡意樣本實際上結合了多項削弱系統防禦的手法。其行為包括：停用或繞過防毒軟體、檢查並關閉 Windows 防火牆、關閉自動更新服務、防止使用者登出或關機、關閉事件日誌紀錄系統、停用監控服務，以及刪除與安全更新相關的排程任務等。截至截稿前並未找到關於更多 heur3 的行為相關敘述，而從 T1562 產生的樣本行為可推測，其行為與木馬病毒常見的行為之一關閉各項安全服務以日後惡意攻擊進行的行為高度相關。

除此之外，雖然於生成階段我們只針對一種技術使大型語言模型進行分析與生成。但最後的執行檔在 VirusTotal 的沙盒分析報告卻顯示出多個戰術的偵測結果。舉例來說，本地端檔案利用 (T1005) 對應的戰術為資源收集，然而 Mal-GPT 所生成的執行檔於 VirusTotal 沙盒分析報告的分析結果顯示其戰術包含執行、持續潛伏、權限提升、防禦規避、憑證存取、發現、資源收集及指揮與控制等多項攻擊階段。由此可知，為了實現一個可於受害者環境中執行且帶有功能的惡意程式，生成流程中會自動完善多種戰術來達成目標。

5. 結論

本研究提出一套基於大型語言模型的惡意程式生成框架 Mal-GPT，結合提示詞工程與 MITRE ATT&CK 框架，實現自動化生成涵蓋惡意程式生命週期各階段的攻擊樣本。實驗結果顯示，所生成的樣本具備 100% 的可執行性，且 15.29% 被主流

防毒軟體標記為可疑，驗證其具備實質攻擊性與隱匿性。我們進一步透過案例分析驗證 Mal-GPT 可依據指定 ATT&CK 技術，生成具備對應行為的樣本，展現其於惡意樣本模擬與主動式資安演練中的應用潛力。

本研究仍存在若干限制：目前僅針對 C++ 語言與 Windows 11 平台進行實驗，尚未涵蓋其他程式語言或作業系統；此外，本研究尚未涵蓋結合外部攻擊工具的惡意程式行為模擬。未來可延伸的方向包括：導入系統監視器與事件檢視器等多源監控資料以強化行為驗證；結合公開威脅情報報告生成真實攻擊樣本；整合外部駭客工具與自動化混淆技術，進一步提升模擬樣本的真實性與多樣性。

6. 倫理考量

本研究旨在探討大型語言模型 (Large Language Models, LLMs) 生成惡意程式的能力，目的是為了解其程式生成的潛力與相關風險，並作為 AI 資安分析的技術基礎研究。我們強調，本研究純屬學術用途，無意鼓勵或促進任何形式的惡意行為。

我們充分理解利用語言模型生成惡意程式可能帶來的倫理與安全風險。為降低此類風險，本研究僅於離線、隔離的實驗環境中執行所有生成與測試，所產出之程式碼亦未曾對外發佈或使用於實際系統中，以確保研究不會造成實際傷害。

我們提醒，濫用生成式模型從事惡意軟體開發，將可能引發嚴重的資訊安全與法律問題。呼籲相關研究人員與實務工作者應嚴守倫理原則，謹慎使用語言模型，並設置必要的技術與制度防線，以防止科技被用於非正當用途。

7. 致謝

本研究成果承蒙國科會計畫編號：112-2222-E-011-011-MY2 與計畫編號：113-2634-F-001-002-MBK 的支持。

References

- [1] Virustotal - analyze suspicious files and urls to detect types of malware. <https://www.virustotal.com/gui/home/upload>, 2025.
- [2] Mika Beckerich, Laura Plein, and Sergio Coronado. Ratgpt: Turning online llms into proxies for malware attacks. *arXiv preprint arXiv:2308.09183*, 2023.
- [3] PV Charan, Hrushikesh Chunduri, P Mohan Anand, and Sandeep K Shukla. From text to mitre techniques: Exploring the malicious use of large language models for generating cyber attack payloads. *arXiv preprint arXiv:2305.15336*, 2023.
- [4] CrowdStrike. What is ransomware as a service (raas)?, 2025.
- [5] Julian Hazell. Spear phishing with large language models. *arXiv preprint arXiv:2305.06972*, 2023.
- [6] Eider Iturbe, Oscar Llorente-Vazquez, Angel Rego, Erkuden Rios, and Nerea Toledo. Unleashing offensive artificial intelligence: Automated attack technique code generation. *Computers & Security*, 147:104077, 2024.
- [7] FortiGuard Labs. The year of the wiper, 2022.
- [8] Microsoft. Backdoor:Win32/ReverseTurtle.Aldha, 2024.
- [9] National Institute of Standards and Technology. Malware - nist computer security resource center, 2024.
- [10] OpenAI. Chatgpt: Optimizing language models for dialogue, 2022.