BÁO CÁO TỔNG KẾT ĐỒ ÁN

**Môn học: Bảo mật web và ứng dụng**

**Tên chủ đề: PTHELPER: AN OPEN SOURCE TOOL TO SUPPORT THE PENETRATION TESTING PROCESS**

*Mã nhóm/Mã đề tài: C15*

**Lớp**: **NT213.P11.ANTT**

1. **THÔNG TIN THÀNH VIÊN NHÓM:**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ và tên** | **MSSV** | **Email** |
| 1 | Trần Anh Khôi | 22520701 | 20520701@gm.uit.edu.vn |
| 2 | Nguyễn Kim Khánh | 22520643 | 22520643@gm.uit.edu.vn |
| 3 | Nguyễn Phan Hữu Khánh | 22520645 | 22520645@gm.uit.edu.vn |
| 4 | Lăng Thị Cẩm Nhung | 22521057 | 22521057@gm.uit.edu.vn |

1. **TÓM TẮT NỘI DUNG THỰC HIỆN:**
   1. **Mã và tên đề tài:**

**C15\_ PTHELPER: AN OPEN SOURCE TOOL TO SUPPORT THE PENETRATION TESTING PROCESS**

* 1. **Bài báo tham khảo chính:**

|  |
| --- |
| [[1] *https://arxiv.org/abs/2406.08242*](https://arxiv.org/abs/2406.08242) |

* 1. **Tóm tắt nội dung chính:**

|  |
| --- |
| **Bài báo này giới thiệu một cách tiếp cận mới trong việc tự động hóa quá trình kiểm thử xâm nhập bằng cách phát triển một công cụ mã nguồn mở và hoạt động theo mô-đun có tên là PTHelper**. |

* 1. **Tóm tắt các kỹ thuật chính được mô tả sử dụng trong bài báo:**

|  |
| --- |
| PTHelper kết hợp bốn mô-đun với nhau, mỗi mô-đun chuyên biệt cho một hoặc nhiều giai đoạn trong quy trình kiểm tra xâm nhập:   * **Mô-đun Scanner** sử dụng công cụ *nmap* để xác định các port đang mở trên mục tiêu và lấy thông tin từ chúng với các kỹ thuật: * **Host Discovery** * **Service and Version Detection** * **Vulnerability Discovery** * **Mô-đun Exploiter** kết nối đến ExploitDB – 1 database chứa thông tin về các lỗ hổng để tìm và tải xuống các mã khai thác tương ứng * **Mô-đun NLPAgent**tích hợp nguồn *NLP* (*Natural Language Processing)* và thiết kế *Prompt Engineering* để chuẩn bị nội dung cho việc viết báo cáo cuối cùng * **Mô-đun Reporter** tạo báo cáo đánh giá kỹ thuật * **Sử dụng khuôn mẫu *Jinja2*** * **Tạo báo cáo đánh giá bảo mật theo chuẩn *SAFR*** * **Tự động điền thông tin từ các mô-đun khác** |

* 1. **Môi trường thực nghiệm của bài báo:**

|  |
| --- |
| 1. Black box infrastructure  * Gồm các mạng cục bộ gồm 3 máy Metasploit được kết nối với nhau như hình dưới, trong đó pentester chỉ có thể nhìn thấy host đầu tiên là Metasploit Window.      1. HackTheBox machine  * Môi trường là một máy chủ trên Internet, cụ thể là một box trên HackTheBox có tên là Blue |

* 1. **Kết quả thực nghiệm của bài báo:**

|  |
| --- |
| 1. **Black box infrastructure**  * PThelper đã có thể phát hiện ra lỗ hổng RCE (Remote Code Execution) trên host đầu tiêu. Sau khi khai thác nó, một kỹ thuật chuyển tiếp port đã được sử dụng để chuyển tiếp tất cả lưu lượng từ PTHelper đến các máy chủ không nằm trong phạm vi mạng trực tiếp (Host B và Host C). PTHelper đã phát hiện ra các lỗ hổng trong hai máy chủ trên và đưa ra danh sách các khai thác, một số trong đó được sử dụng để đạt được DoS (từ chối dịch vụ) và RCE để xâm phạm hoàn toàn cơ sở hạ tầng  1. HackTheBox machine  * Công cụ đã phát hiện ra lỗ hổng CVE-2017-0144 tồn tại trên máy chủ này và trả về một số khai thác để tận dụng RCE |

* 1. **Công việc/tính năng/kỹ thuật mà nhóm thực hiện lập trình và triển khai cho demo:**

|  |
| --- |
| * Cài đặt công cụ, lấy api key từ NVD và OpenAI * Cấu hình công cụ và máy ảo * Chỉnh sửa kịch bản cho phù hợp với trang thiết bị mà nhóm hiện có * Chỉnh sửa mã nguồn (không đáng kể, chỉ thêm options cho *nmap*) * Tìm ra các hạn chế của công cụ * Tìm hiểu kỹ thuật pivot host |

* 1. **Các khó khăn, thách thức hiện tại khi thực hiện:**

|  |
| --- |
| Nhóm không gặp khó khăn nào đáng kể |

1. **TỰ ĐÁNH GIÁ MỨC ĐỘ HOÀN THÀNH SO VỚI KẾ HOẠCH THỰC HIỆN:**

|  |
| --- |
| 90% |

1. **NHẬT KÝ PHÂN CÔNG NHIỆM VỤ:**

|  |  |  |
| --- | --- | --- |
| **STT** | **Công việc** | **Phân công nhiệm vụ** |
| 1 | Tìm hiểu về mô-đun Scanner | Nguyễn Kim Khánh |
| 2 | Tìm hiểu về mô-đun Exploiter | Nguyễn Phan Hữu Khánh |
| 3 | Tìm hiểu về mô-đun NLPAgent | Lăng Thị Cẩm Nhung |
| 4 | Tìm hiểu về mô-đun Repoter | Trần Anh Khôi |
| 5 | Triển khai kịch bản **Black box infrastructure** | Trần Anh Khôi, Nguyễn Phan Hữu Khánh |
| 6 | Triển khai kịch bản machine | Lăng Thị Cẩm Nhung, Nguyễn Kim Khánh |
| 7 | Tổng hợp số liệu và viết báo cáo | Cả nhóm |

Nội dung

[I. Bối cảnh 6](#_Toc392611678)

[1.1 Kiểm tra bảo mật và xâm nhập tấn công 6](#_Toc60293266)

[1.2. Vòng đời kiểm tra xâm nhập 6](#_Toc1000899445)

[II. Tổng quan 7](#_Toc726762374)

[2.1. Phương pháp kiểm tra xâm nhập truyền thống 7](#_Toc766815046)

[2.2. Phương pháp kiểm tra xâm nhập được đề xuất 8](#_Toc242183218)

[2.3. Cách thức hoạt động 8](#_Toc120309498)

[III. Kiến trúc 9](#_Toc1621100608)

[3.1. Mô đun Scanner: 9](#_Toc1511885461)

[3.2. Mô đun Exploiter 10](#_Toc1566014556)

[3.3. Mô-đun NLPAgent 11](#_Toc1781649081)

[3.4. Mô-đun Reporter 13](#_Toc1031349227)

[IV. Phương pháp thực hiện 14](#_Toc1385598652)

[4.1. Cài đặt 14](#_Toc2133854036)

[4.2. Kịch bản 14](#_Toc222843933)

[4.3. Cấu hình: 15](#_Toc1759298210)

[4.4. Hiện thực 18](#_Toc1255396536)

[4.4.1. TryHackMe machine: 18](#_Toc372137783)

[4.4.2 Black box: 19](#_Toc1127865103)

[4.5 Kết quả 22](#_Toc1551125840)

[4.5.1. TryHackMe machine: 22](#_Toc1973504228)

[4.5.2. Black box: 23](#_Toc1085485573)

[V. Kết luận và hướng phát triển 23](#_Toc1335075937)

[5.1. Ưu điểm: 23](#_Toc1408689961)

[5.2. Nhược điểm: 24](#_Toc31850884)

[5.3. Hướng phát triển 24](#_Toc2112662977)

**BÁO CÁO TỔNG KẾT CHI TIẾT**

# I. Bối cảnh

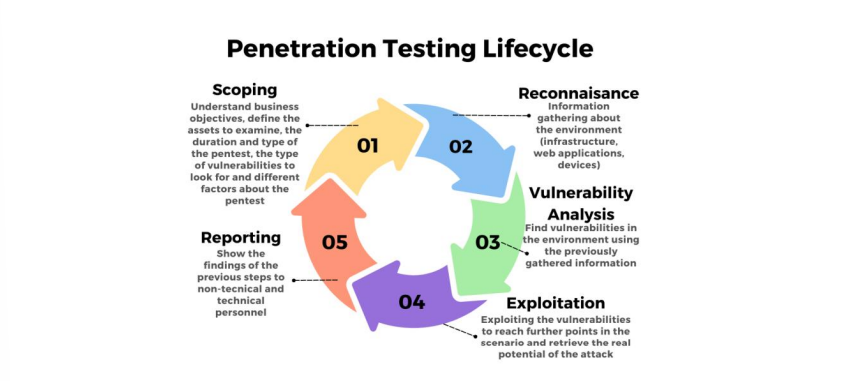
## 1.1 Kiểm tra bảo mật và xâm nhập tấn công

Ngày nay, việc chỉ áp dụng biện pháp bảo mật phòng thủ không thể đảm bảo an toàn cho các ứng dụng hay hệ thống hiện đại. Một phương pháp khác là bảo mật tấn công tập trung vào việc tìm hiểu cách hệ thống có thể bị xâm phạm, bao gồm nhiều kỹ thuật như đánh giá lỗ hổng, exploit development, social engineering...Cả hai phương pháp bảo mật trên có thể bổ sung cho nhau để đạt được mức độ bảo mật tối đa.

Trong quá trình kiểm thử, các lỗ hổng được phát hiện sẽ được pentester khai thác để xâm nhập vào các host khác trong cở sở hạ tầng của tổ chức nhằm phát hiện những lỗ hổng nằm sâu hơn. Pentester đóng vai trò như một kẻ tấn công ngoài đời thực nên họ cần trang bị những công cụ và phương pháp mới nhất để bắt chước hành động của chúng.

## 1.2. Vòng đời kiểm thử xâm nhập

Kiểm tra xâm nhập là một quá trình có cấu trúc bao gồm nhiều giai đoạn khác nhau thường cần được thực hiện trong một khoảng thời gian giới hạn. Phương pháp chung để phân chia các giai đoạn là như sau:

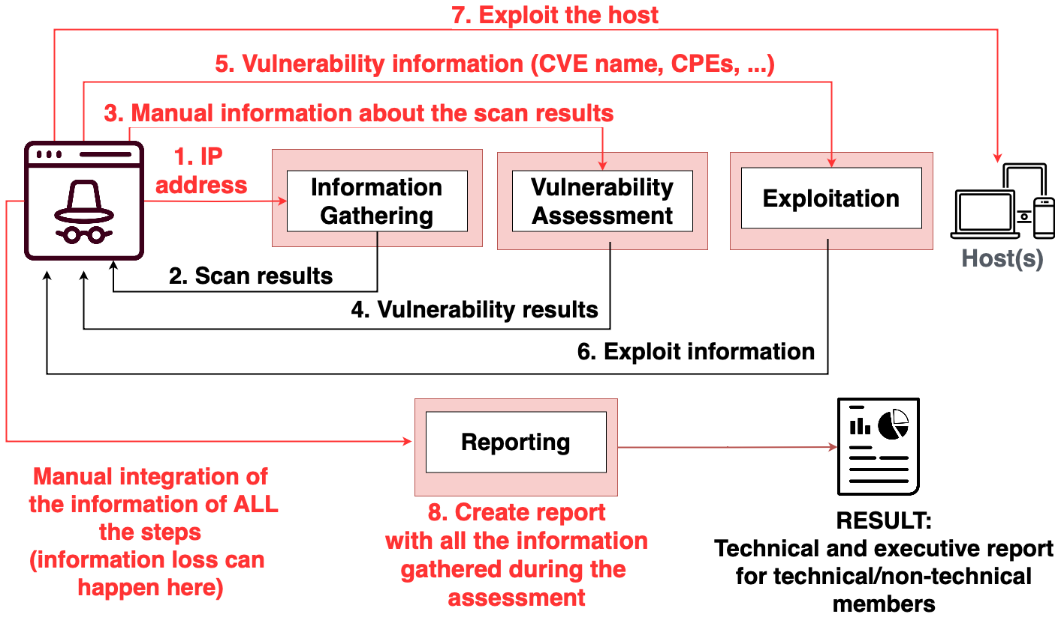


**Sơ đồ vòng đời quá trình kiểm thử xâm nhập**

* **Scoping:** pentester làm việc với khách hàng để xác định các tài sản trong hệ thống cần được kiểm tra cũng như các quy tắc liên quan đến đạo đức nghề nghiệp khi triển khai quá trình.
* **Reconaissance:** thu thập thông tin về môi trường mục tiêu
* **Vulnerable Analysis:** phân tích các lỗ hổng đã tìm được trong môi trường mục tiêu
* **Exploitation:** khai thác các lỗ hổng tìm được để tiến sâu hơn vào hệ thống, tìm ra các mối đe dọa thực sự
* **Báo cáo:** báo cáo tổng kết quá trình kiểm thử xâm nhập

# II. Tổng quan

## 2.1. Phương pháp kiểm tra xâm nhập truyền thống

  
*Hình 2.1.1 Phương pháp kiểm tra xâm nhập truyền thống*

Hình trên là mô hình tổng quát cho phương pháp kiểm tra xâm nhập truyền thống, đường màu đỏ hiển thị tương tác của người kiểm tra xâm nhập, còn đường màu đen biểu thị đầu ra của từng quy trình. Phương pháp này có những hạn chế:

* **Giới hạn về thời gian, nguồn lực:** các lỗ hổng phức tạp hay ẩn giấu tốt đôi khi bị giới hạn về thời gian, nhân lực và chi phí để kiểm thử nên không thể giải quyết
* **Phụ thuộc vào kỹ năng của người kiểm thử:** kết quả kiểm thử phụ thuộc lớn vào kỹ năng, kinh nghiệm của pentester
* **Công việc không đúng chuyên môn:** pentester phải tổng hợp và sắp xếp lại thông tin để viết báo cáo, điều này không nằm trong chuyên môn của họ. Thêm vào đó là vòng đời kiểm thử có nhiều giai đoạn nên thông tin thu thập được thường không đồng nhất, gây khó khăn cho việc lập báo cáo

## 2.2. Phương pháp kiểm tra xâm nhập được đề xuất

Hình 3 mô tả phương pháp tiếp cận của PTHelper liên quan đến việc tự động hóa pentest.


*Hình 2.2.1 Phương pháp kiểm thử xâm nhập tự động sử dụng PTHelper*

Đường màu đỏ biểu thị các tương tác của pentester, đường màu tím biểu thị hoạt động của công cụ PTHelper.

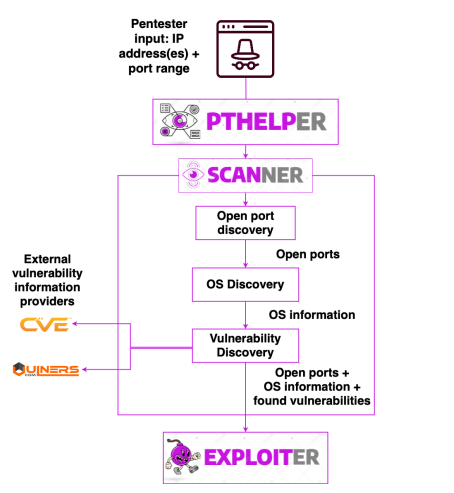
Khi sử dụng PTHelper, đầu vào duy nhất cần được cung cấp liên quan đến hệ thống mục tiêu là IP và port, các đầu vào còn lại là lựa chọn dịch vụ công cụ sẽ sử dụng. Sau khi khởi chạy công cụ thì quá trình diễn ra hoàn toàn tự động, phần tương tác của pentester nằm ở giai đoạn khai thác lỗ hổng bằng việc chạy mã thực thi và điền thông tin cá nhân cho báo cáo(name,organization,..). Điều này cho thấy công cụ sẽ giảm bớt gánh nặng thời gian cũng như công sức đi tìm kiếm lỗ hổng và báo cáo cho pentester.

## 2.3. Cách thức hoạt động

* **Scanner (Reconnaissance and Vulnerability Assessment)**: thực hiện quét và thu thập thông tin trên hệ thống mục tiêu, đánh giá các lỗ hổng có thể tồn tại trên hệ thống để làm cơ sở cho các bước tiếp theo.
* **Exploiter (Exploitation)**: khai thác các lỗ hổng đã được xác định trong bước trước.
* **NLP Agent (Reasoning and Report Fulfilling)**: sử dụng công nghệ xử lý ngôn ngữ tự nhiên (NLP) để phân tích kết quả từ các giai đoạn trước, hỗ trợ lập luận và tự động tạo nội dung cho báo cáo kiểm thử.
* **Reporter (Report Generating and Management)**: tổng hợp thông tin và tạo báo cáo kiểm thử xâm nhập chi tiết. Báo cáo có thể bao gồm các phát hiện, mức độ nghiêm trọng, và đề xuất giải pháp khắc phục.
* Mỗi mô-đun sẽ chỉ tương tác với mô-đun khác khi gửi dữ liệu đầu ra hoặc nhận dữ liệu đầu vào

# III. Kiến trúc

## 3.1. Mô đun Scanner:



*Hình 3.1.1 Hoạt động và tương tác của Mô-đun Scanner*

**Bước 1: Quét và thu thập thông tin**

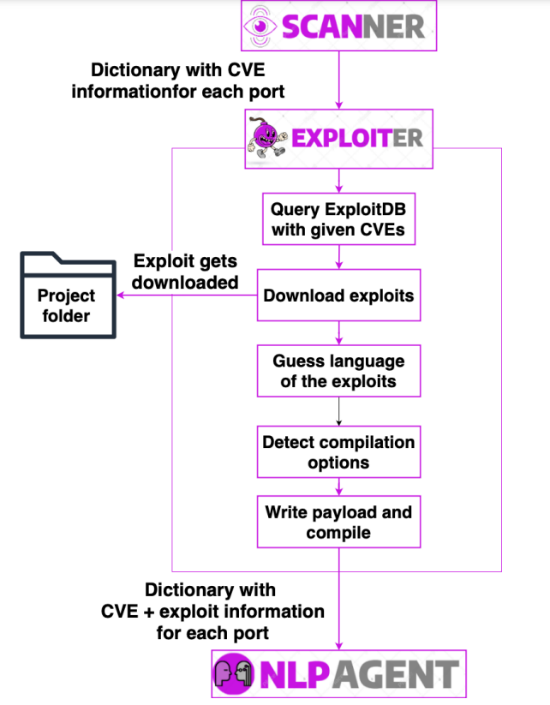
* Sử dụng *nmap* – một công cụ quét bảo mật mạng phổ biến.
* Host Discovery: kiểm tra xem các port đang mở trên mục tiêu
* Service and Version Detection: xác định dịch vụ nào đang chạy trên các port mở (ví dụ: HTTP, SSH) đi kèm với phiên bản của chúng

**Bước 2: Khám phá lỗ hổng ( Vulnerablity Discovery)**

* *Nmap Vulners Script*: tìm các lỗ hổng bảo mật trên những port mở
* *nvdlib*: thư viện python có chức năng truy vấn cơ sở dữ liệu lỗ hổng quốc gia (NVD) để thu thập mô tả và chỉ số CVSS của các CVE đã phát hiện.

**Bước 3: Truyền dữ liệu cho mô-đun Exploiter**

## 3.2. Mô đun Exploiter



*Hình 3.2.1 Hoạt động và tương tác của Mô-đun Exploiter*

**Bước 1: Nhận đầu vào từ mô-đun Scanner**

* Dữ liệu từ mô-đun Scanner bao gồm:
* IP mục tiêu
* Danh sách các port mở và những dịch vụ đang mở trên chúng
* Danh sách CVE được phát hiện qua quá trình quét và phân tích, mỗi CVE sẽ tương ứng với một hoặc nhiều port mở.

**Bước 2: Tìm kiếm mã khai thác và tải xuống**

* Dựa vào danh sách CVE do mô-đun Scanner cung cấp, mô-đun Exploiter tìm kiếm các mã khai thác công khai tương ứng có sẵn trên ExploitDB
* Tải mã khai thác về và bổ sung các thông tin như:
* Ngôn ngữ lập trình của mã
* Mã đã được cộng đồng trên ExploitDB kiểm tra hay chưa.
* Thư viện còn thiếu (nếu có)
* Đường dẫn đến tệp mã
* Ngoài ra mô-đun cũng nhận biết các tùy chọn biên dịch trong tệp mã và thực hiện biên dịch để pentester có thể sử dụng

**Bước 3: Cung cấp thông tin cho pentester**

* Hiển thị thông tin về CVE có trong mỗi port:
* Mô tả ngắn gọn của từng CVE, điểm CVSS
* Mã khai thác tương ứng với mỗi CVE (nếu có)
* Thông tin bổ sung (đã đề cập ở trên)
* Việc chạy các mã khai thác này hoàn toàn phụ thuộc vào pentester

**Bước 4: Truyền dữ liệu cho mô-đun NLPAgent**

* Đầu ra của mô-đun này bao gồm thông tin nhận được từ mô-đun Scanner cùng với thông tin khai thác bổ sung. Đây cũng là input của mô-đun NLPAgent

## 3.3. Mô-đun NLPAgent



*Hình 3.3.1 Hoạt động và tương tác của Mô-đun NLPAgent*

**Bước 1: Nhận đầu vào từ Exploiter**

* Quá trình khai thác lỗ hổng của pentester trong quá trình kiểm thử sẽ không bị ảnh hưởng bởi mô-đun NLPAgent vì nó nằm ngoài phạm vi của công cụ

**Bước 2: Tương tác với NLPAgent**

* Sử dụng Prompt Engineering để tạo các câu lệnh yêu cầu GPT cung cấp thông tin cần thiết cho từng phần của báo cáo
* Mỗi phần của báo cáo yêu cầu mức độ chi tiết khác nhau và các *prompt* này sẽ được thiết kế sao cho phù hợp với mỗi mục đích đó:
* Executive Summary: tạo phần tóm tắt, cung cấp một cái nhìn tổng quan về các lỗ hổng, hệ thống, và các biện pháp khắc phục -> Dành cho người không chuyên.
* Findings: cung cấp báo cáo chi tiết cho từng lỗ hổng, giải thích mức độ nghiêm trọng và đề xuất các biện pháp khắc phục-> Dành cho người có chuyên môn
* Dựa trên các prompt đã được tạo ra, mô-đun sẽ tương tác với NLP Agent (chẳng hạn như ChatGPT) để tạo ra các phần của báo cáo, từ thông tin tổng quan dễ hiểu cho người không chuyên đến chi tiết kỹ thuật cho người chuyên môn.

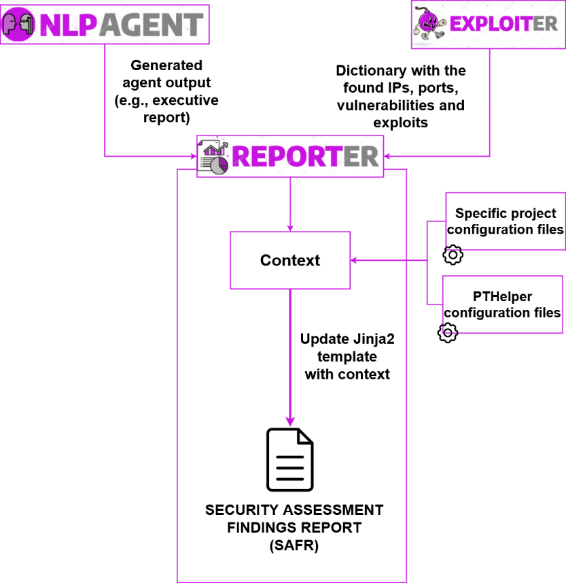
**Bước 3: Tối Ưu Hóa Việc Sử Dụng Token gọi API và tạo báo cáo**

* Tối ưu hóa việc sử dụng Token giúp tiết kiệm tài nguyên và chi phí trong khi vẫn giữ được độ chính xác và chi tiết của các báo cáo
* Sau khi hoàn tất các phần của báo cáo (Executive Summary, Findings), những thông tin này sẽ được kết hợp với dữ liệu input để tạo thành một dữ liệu hoàn chỉnh cho báo cáo.

**Bước 4: Chuyển thông tin đến mô-đun Reporter.**

* Có thể cấu hình tham số (trong mã nguồn):
* ***Temperature:*** quyết định mức độ sáng tạo hay tính logic trong đầu ra của NLPAgent (nếu giá trị càng cao, đầu ra càng sáng tạo, nhưng cũng có thể kém chính xác).
* ***API version:*** cho phép thay đổi phiên bản API của ChatGPT (ví dụ: phiên bản 3.5 hoặc 4).

## 3.4. Mô-đun Reporter



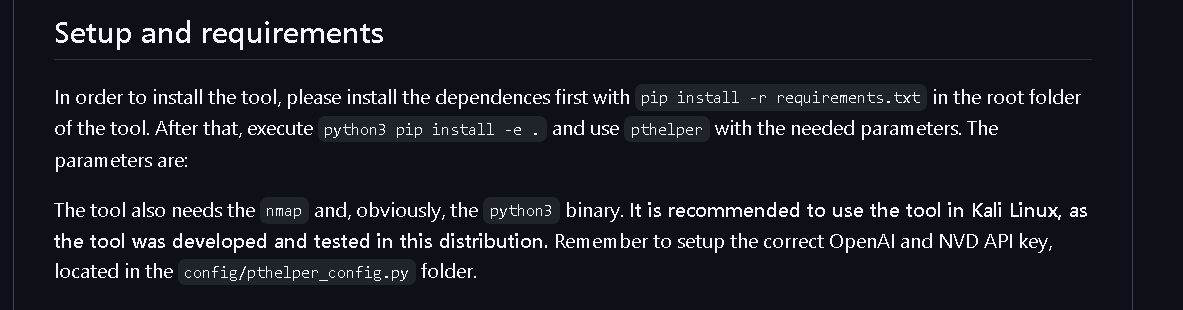
*Hình 3.4.1 Hoạt động và tương tác của Mô-đun Reporter*

* Mô-đun nhận thông tin tổng hợp được từ các mô-đun Exploiter và NLPAgent. Nội dung báo cáo bao gồm:
* **Bản tóm tắt điều hành (executive summary):** tóm tắt các phát hiện bảo mật quan trọng.
* **Bảng tổng quan về tổ chức:** bảng tóm tắt chứa tổng quan về kiến trúc với IP đã quét, port mở và các lỗ hổng được tìm thấy ở định dạng CVE.
* **Phần phát hiện chi tiết (findings section):** phần chứa phát hiện chi tiết về lỗ hổng, mức độ nghiêm trọng của lỗ hổng và tác động của lỗ hổng trong tổ chức, danh sách các khai thác công khai cho lỗ hổng đã cung cấp và danh sách các bước chi tiết để khắc phục
* Khung mẫu được sử dụng là *Jinja2* cho các tài liệu *docx*.
* Mặc dù công cụ mặc định tạo báo cáo ở định dạng *docx*, nó vẫn cho phép pentester báo cáo ở các định dạng khác như Markdown hoặc Latex, cũng như bổ sung các framework templating khác để phù hợp với nhu cầu của khách hàng
* Mẫu báo cáo đi kèm đã bao gồm tất cả các phần mà một báo cáo chuyên nghiệp cần có. Tuy nhiên, pentester có thể thêm thông tin bổ sung vào các phần này để hoàn thiện báo cáo sao cho dễ hiểu đối với cả các thành viên có chuyên môn và không trong tổ chức.

# IV. Phương pháp thực hiện

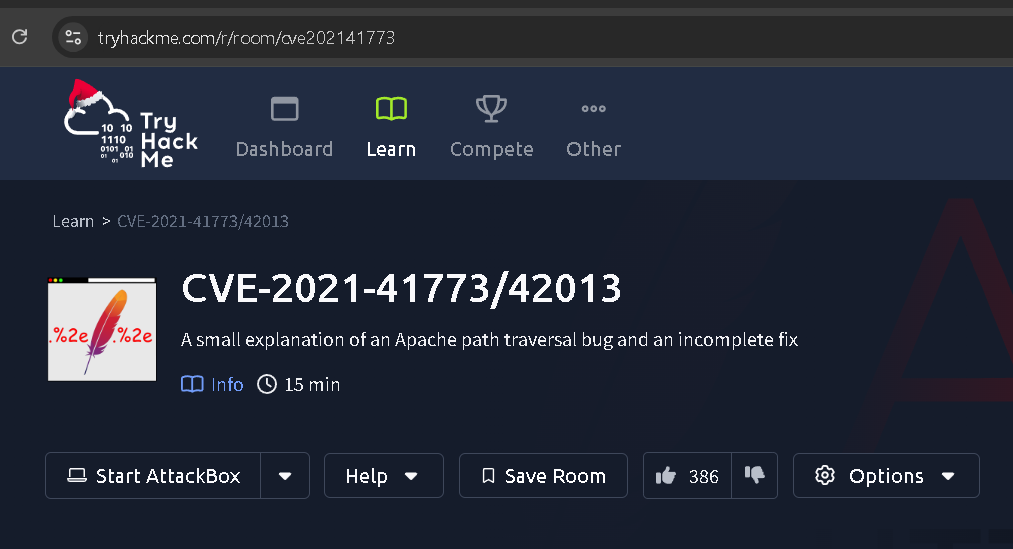
## 4.1. Cài đặt

* Môi trường cài đặt công cụ: Kali Linux.
* Do sự thay đổi từ Python 3.11, việc cài đặt các package trong *requirements.txt* cần được thực hiện trong *môi trường ảo*

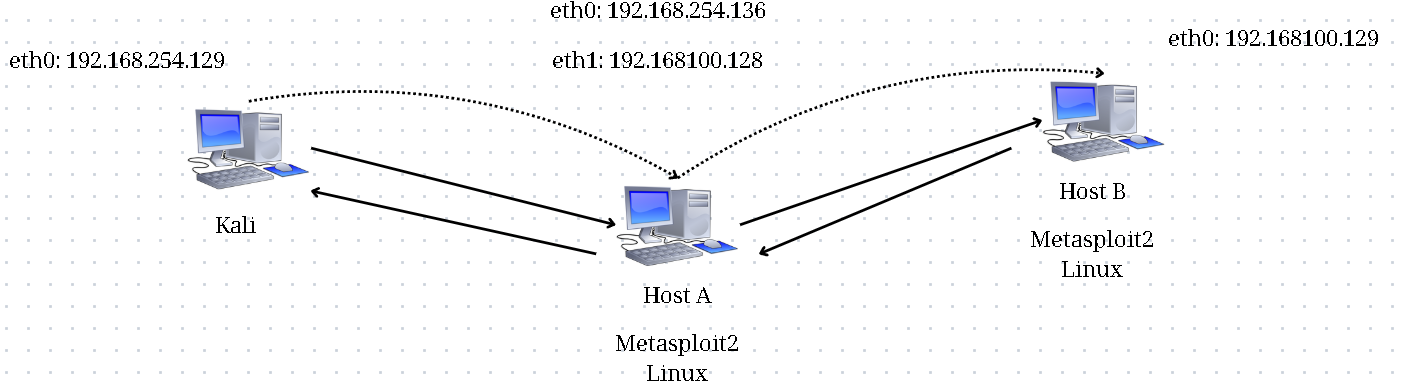


## 4.2. Kịch bản

* Với kịch bản HTB machine:
* Nhóm thay bằng một kịch bản trên *TryHackMe* vì *Blue* – box mục tiêu trong kịch bản của bài báo là một Window machine, lỗ hổng tồn tại và việc khai thác không liên quan đến nội dung môn học
* Máy chủ mục tiêu trong kịch bản mới có tên là *CVE-2021-41773/42013,* chứa lỗ hổng tồn tại trong web server có thể dẫn đến RCE

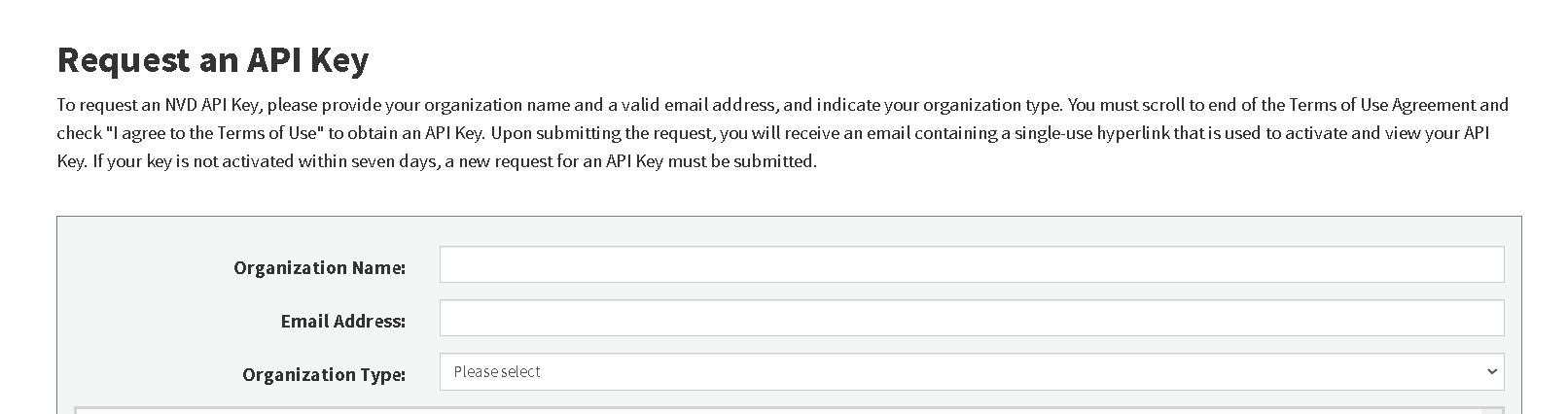


* Với kịch bản Black box:
* Vì yêu cầu phần cứng của máy *Window Metasploit3* cao hơn so với trang thiết bị mà nhóm hiện có nên không thể đáp ứng được.
* Kịch bản được sửa đổi thành chỉ có 2 máy *Linux Metasploit* với cấu hình như cũ là công cụ chỉ có thể nhận diện 1 host.

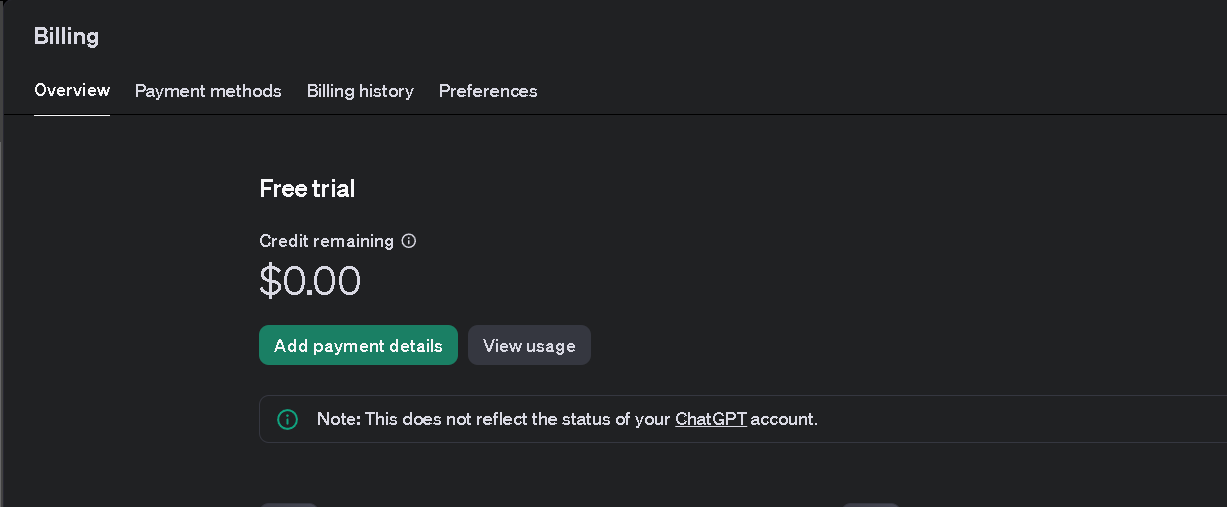


## 4.3. Cấu hình:

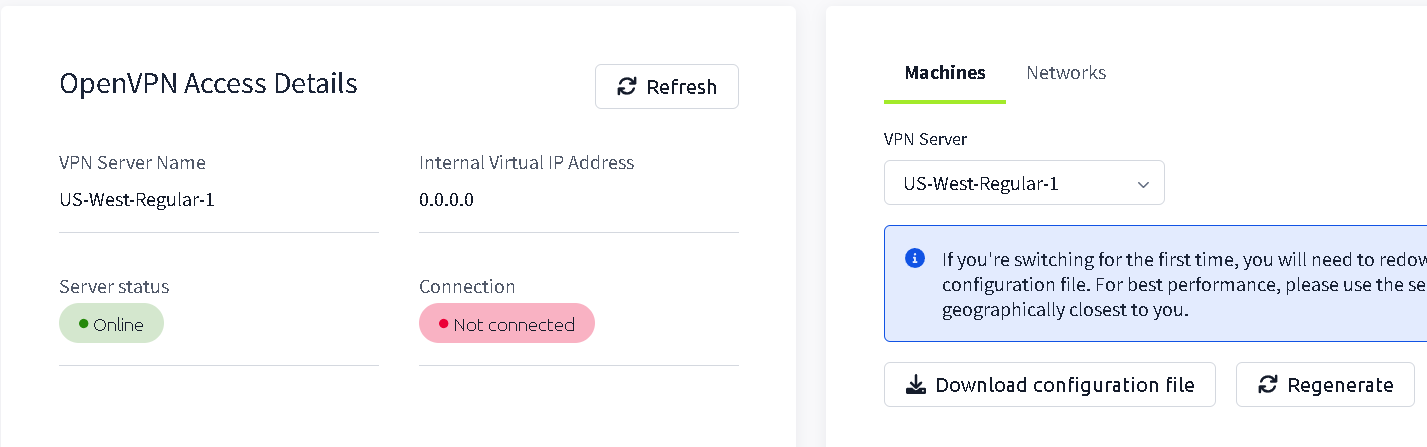
* Công cụ PTHelper:
* Request NVD API key:



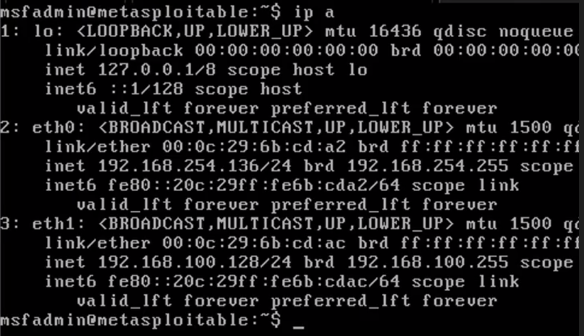
* Tạo OpenAI API key: cần thêm payment method để có thể tạo và sử dụng API key



* TryHackMe machine:
* Để kết nối đến máy chủ cần file khởi tạo kết nối vpn



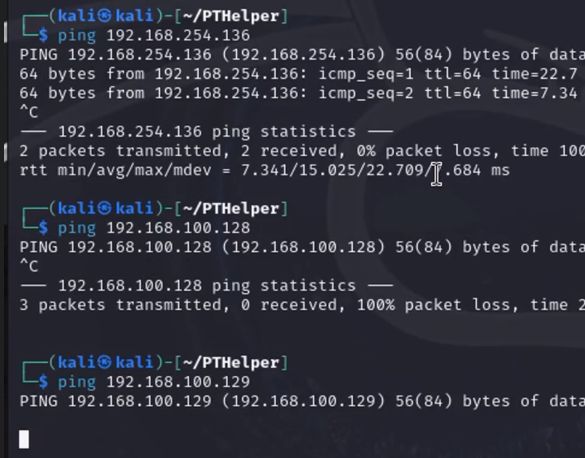
* Black box infrastructure:
* 2 máy Metasploit2: chia thành Host A và Host B
* Host A có 2 interface NAT và Host-Only



* Host B có 1 interface Host-Only



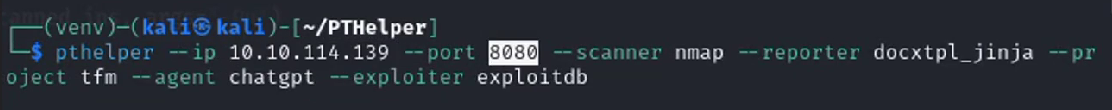
* Kali chỉ có thể nhận diện Host A qua NAT



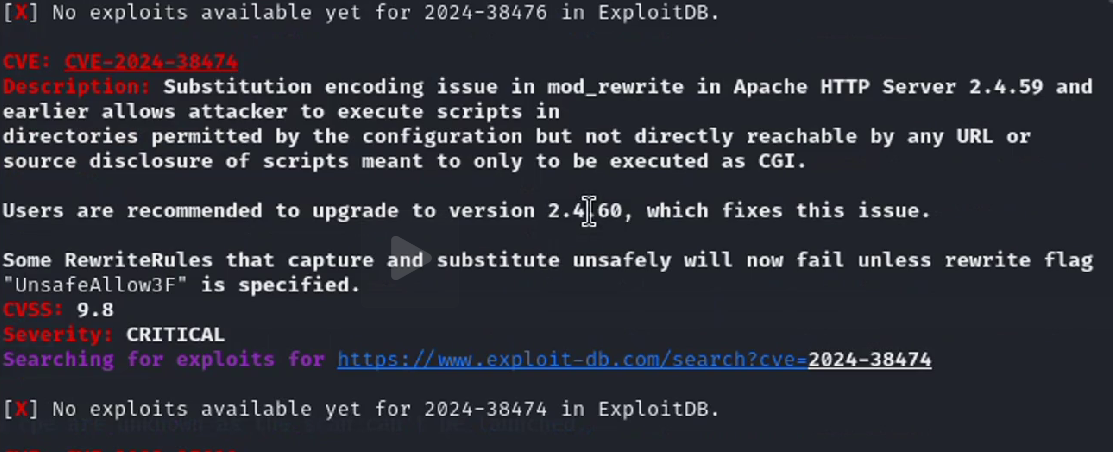
## 4.4. Hiện thực

### 4.4.1. TryHackMe machine:

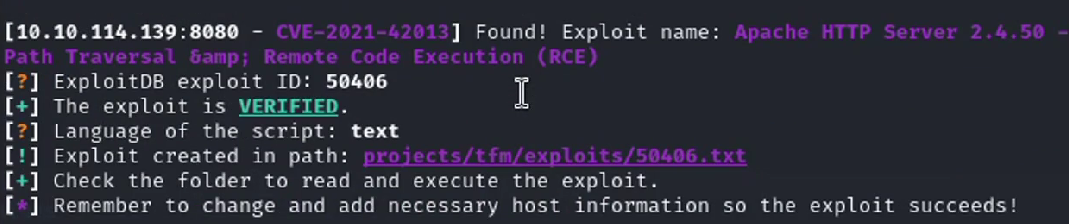
* Thực thi lệnh scan: tham khảo link github trong phần cài đặt và thay đổi giá trị của 2 argument *ip* và *port* để chỉ định mục tiêu quét



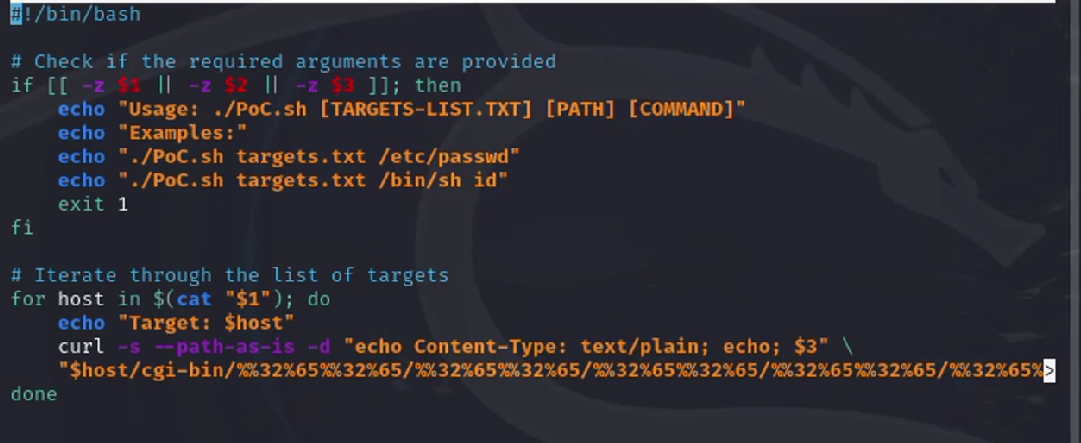
* Scanner:



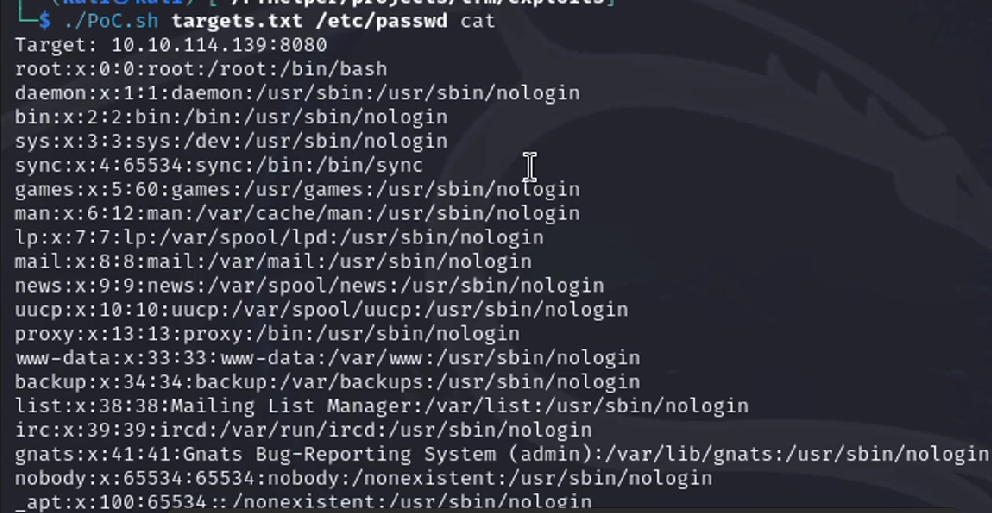
* Exploiter:
* Chọn CVE liên quan tới RCE để khai thác, sử dụng mã mà công cụ đã cung cấp, nên chọn những mã khai thác đã được *verified*



* Các mã khai thác được cung cấp thường có lỗi syntax hoặc thiếu thư viện cần thiết nên cần phải đáp ứng các yêu cầu trên. Dưới dây là mã đã được sửa syntax

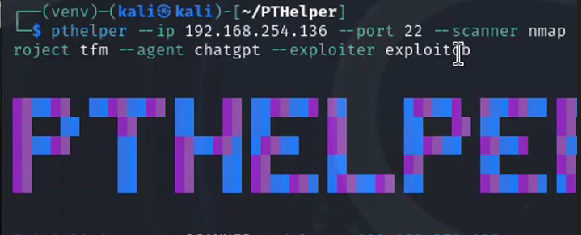


* Exploit thành công

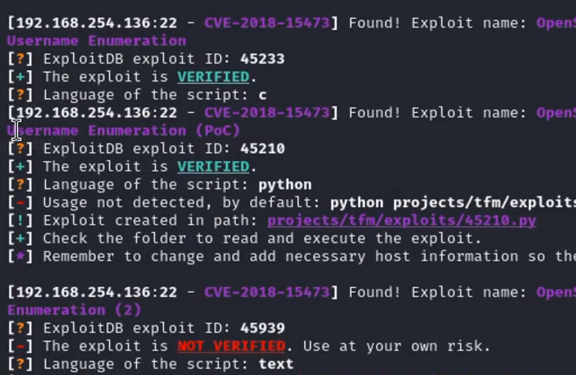


### 4.4.2 Black box:

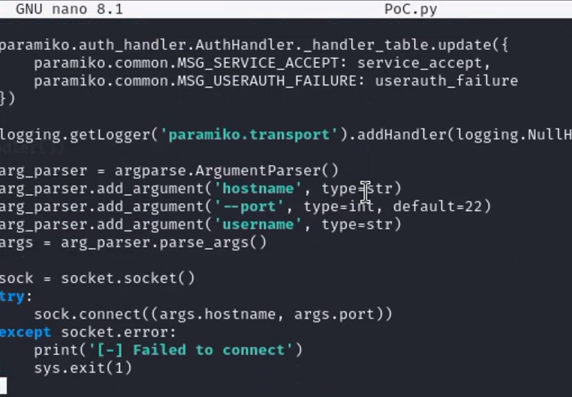
* Thực thi lệnh quét:



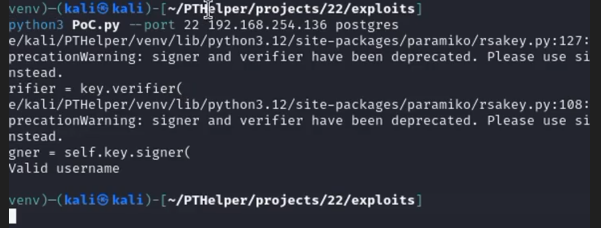
* Scanner:



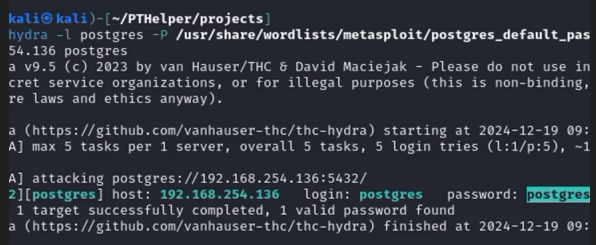
* Exploiter:
* Exploit lỗ hổng Username Enumeration đã được cung cấp mã khai thác ở trên, tương tự kịch bản đầu tiên khi phải thực hiện sửa syntax và cài đặt thư viện



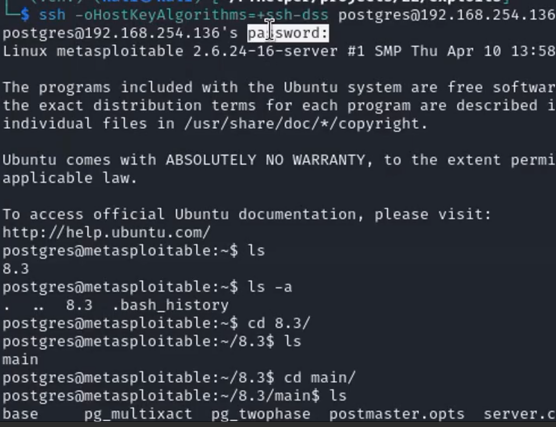
* Tìm được các valid username, trong đó có “postgres”



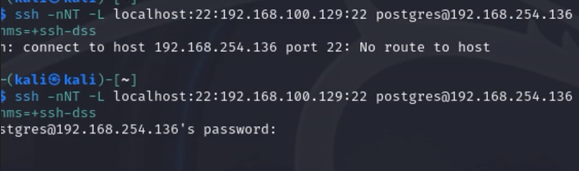
* Dùng *hydra* để brute force mật khẩu , tìm được mật khẩu của user trên là “postgres”



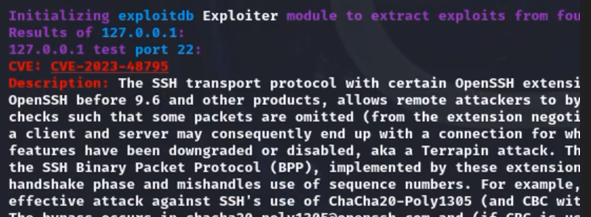
* Thành công *ssh* vào host A khi sử dụng credential đã tìm được



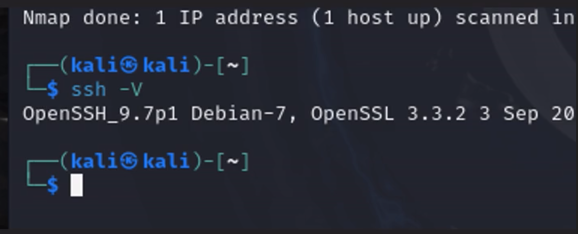
* Thực hiện *pivot host* từ port 22 trên localhost sang port 22 của host B với máy trung gian là compromised host A



* Scan port 22 trên localhost nhưng thực chất là đang scan port 22 trên host B



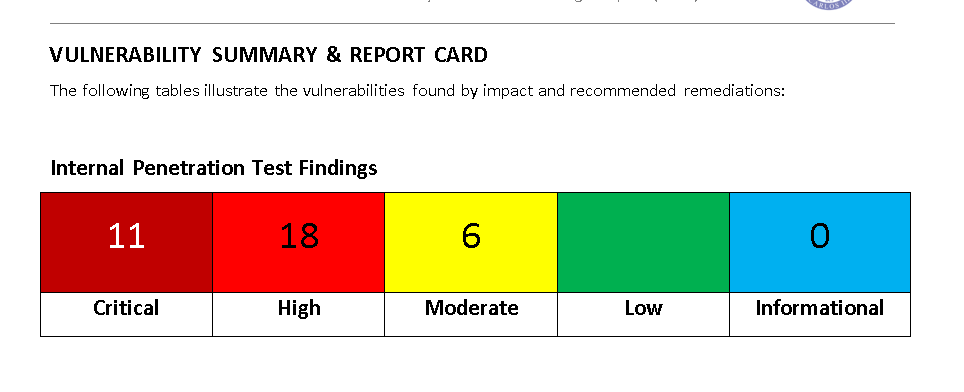
* Kali sử dụng phiên bản OpenSSH phiên bản mới hơn nhiều nên việc scan mục tiêu sử dụng pivot host đã thành công



## 4.5 Kết quả

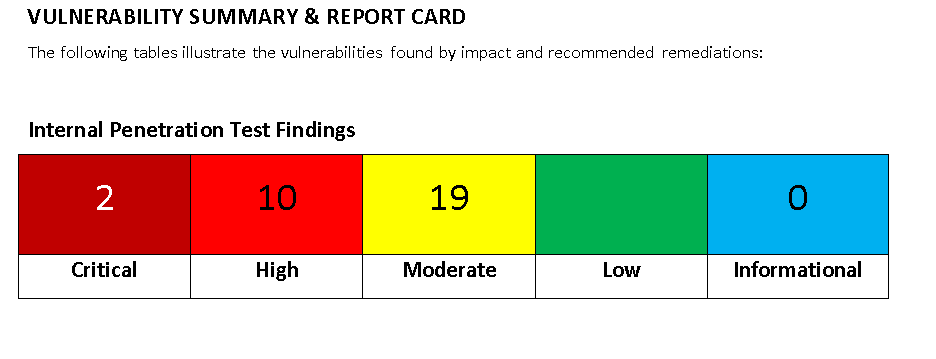
### 4.5.1. TryHackMe machine:

* Port: 8080 (Apache 2.4.49)
* Số lượng lỗ hổng phát hiện: 35 CVE
* Thời gian chạy khoảng 16 phút
* Report:

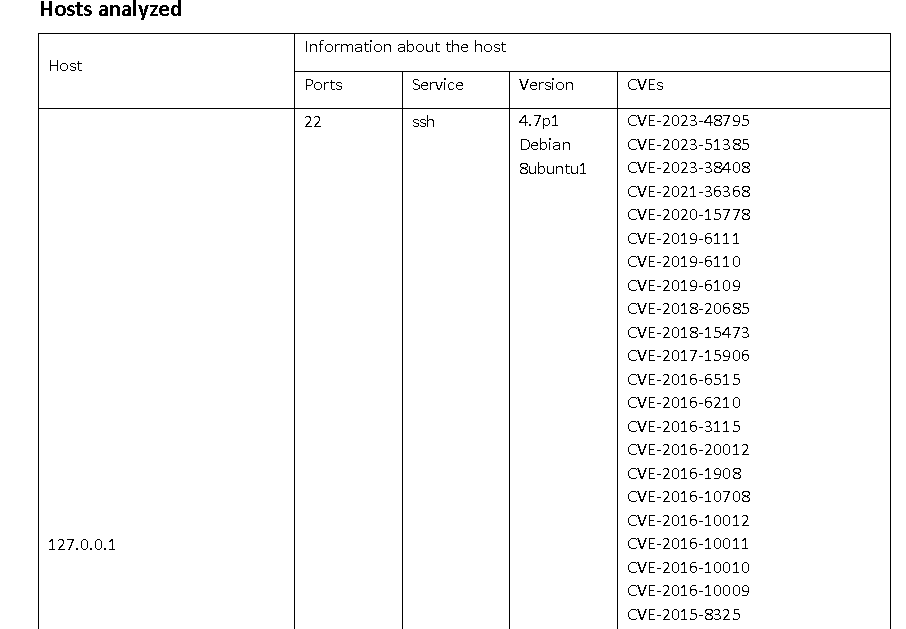


### 4.5.2. Black box:

* Port: 22 (OpenSSH 4.7)
* Số lượng lỗ hổng phát hiện: 31 CVE
* Host A và Host B đều chạy khoảng 14 phút
* Report:
* Host A:



* Host B:



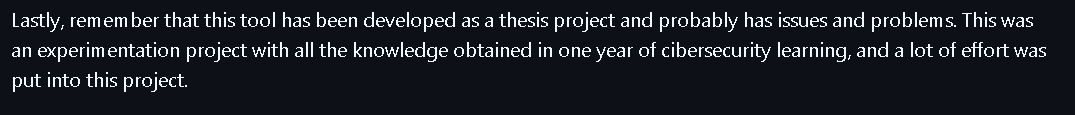
# V. Kết luận và hướng phát triển

## 5.1. Ưu điểm:

* Tự động hóa quá trình kiểm thử mà quyền khai thác vẫn nằm ở pentester
* Dễ sử dụng
* Cung cấp mã khai thác và các thông tin hữu ích liên quan
* Tạo báo cáo tự động, tiết kiệm thời gian cho pentester
* Kiến trúc module giúp việc tích hợp các tính năng hoặc công cụ mới trở nên dễ dàng

## 5.2. Nhược điểm:

* Nhóm tác giả cũng đã đề cập đến khả năng có vấn đề của công cụ.



* Vẫn còn tồn tại vấn đề liên quan đến chức năng:
* Trong kịch bản black box, mục tiêu là Metasploit2 - một môi trường có rất nhiều lỗ hổng. Sau khi tham khảo nguồn trên Internet, ở những port như 21 – vsftpd, 23 – telnet,...đều tồn tại lỗ hổng nhưng công cụ vẫn không phát hiện được. Với các port phát hiện được thì lại không thể quét ra các lỗ hổng đã được công khai trên các write up về Metasploit2.
* Trong phần scan có thông báo: *“PTHelper not running as root. OS scan will not work...”* nên có lẽ 1 phần chức năng đã bị bỏ qua. Nhưng vì các packages cài đặt trong môi trường ảo nên không thể chạy root được. Nếu thực hiện loại bỏ cơ chế này của python thì lại có thể dẫn đến những vấn đề như xung đột gói thư viện...Ngay cả trong video demo bài báo cung cấp cũng có dòng thông báo này
* Vấn đề trong các mã khai thác:
* Thường chứa lỗi syntax.
* Với các lỗ hổng cần sự tác động của nạn nhân để khai thác như XSS, CSRF, clickjacking,...thì các file chỉ đóng vai trò hướng dẫn chứ không thể chạy như một file thực thi

## 5.3 So sánh với các công cụ kiểm thử tự động khác:

* Với các công cụ trước đây, pentester phải làm việc với *dữ liệu thô* mà họ thu được sau khi quét
* Với PTHelper thì dữ liệu sẽ được làm giàu bằng AI trước khi hiển thị cho pentester. Ví dụ như thông tin về các CVE trong mô-đun Exploiter hay nội dung báo cáo sau khi kết thúc quá trình kiểm thử tự động
* Trong mô-đun NLPAgent có thực hiện việc xử lý ngôn ngữ tự nhiên để tạo thành nội dung cho mỗi phần trong báo cáo

## 5.4. Hướng phát triển

Không thể phủ nhận những tiện lợi mà công cụ mang lại trong quá trình kiểm thử. Về mặt chức năng thì nó vẫn có thể tiếp tục cải thiện trong tương lai vì đó mới chỉ là sản phẩm mà nhóm tác giả thực hiện sau 1 năm học về cyber security. Công cụ có thể mở rộng hơn về mục tiêu kiểm thử như Web hay ứng dụng di động. Bên cạnh đó là tích hợp nhiều sự lựa chọn hơn trong các mô-đun ví dụ như với NLPAgent thay vì ChatGPT thì sẽ có thêm Bard, LLama,...với Exploiter thì có thể chọn nhiều database hơn để tăng tỷ lệ tìm thấy thông tin lỗ hổng.

**HẾT**