**Navigate CMS Unauthenticated Remote Code Execution**

*CVE-2018-17552 (Unauthenticated SQL Injection) + CVE-2018-17553 (Path traversal by upload file)*

# Phần cấu hình

## Cấu hình Navigate web

/etc/var/html

/etc/apache2/sites-available/yourdomain.conf

/etc/hosts

a2ensite yourdomain.conf

a2dissite

sudo systemctl restart apache2

**Remote debug tới máy ảo Ubuntu để quan sát rõ hơn quá trình exploit của CVE-2018-17552**

## Cài đặt xdebug trên máy ảo

sudo apt update

sudo apt install php-xdebug

## Thiết lập xdebug.ini trên máy ảo

sudo nano /etc/php/<your\_php\_version>/mods-available/xdebug.ini

xdebug.remote\_enable=1

xdebug.remote\_autostart=1

xdebug.remote\_port=9000

Mở cổng 9000 để debug

## Cài đặt OpenSSH Server

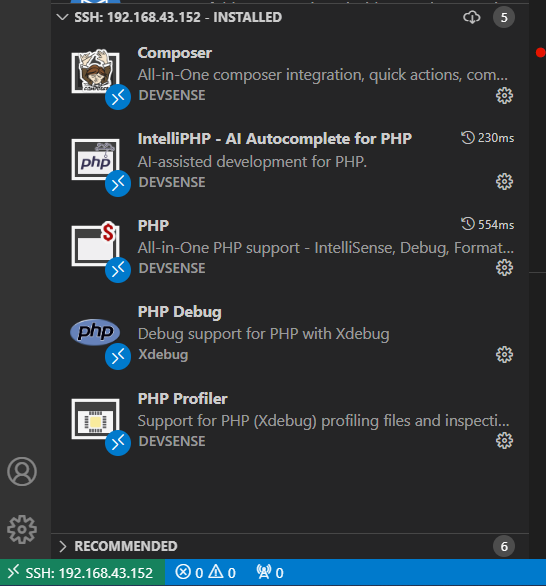
sudo apt-get install openssh-server

sudo systemctl enable ssh

sudo systemctl start ssh

## SSH từ vscode tới máy Ubuntu

## Cài đặt extention trên VScode (trên máy ubuntu)

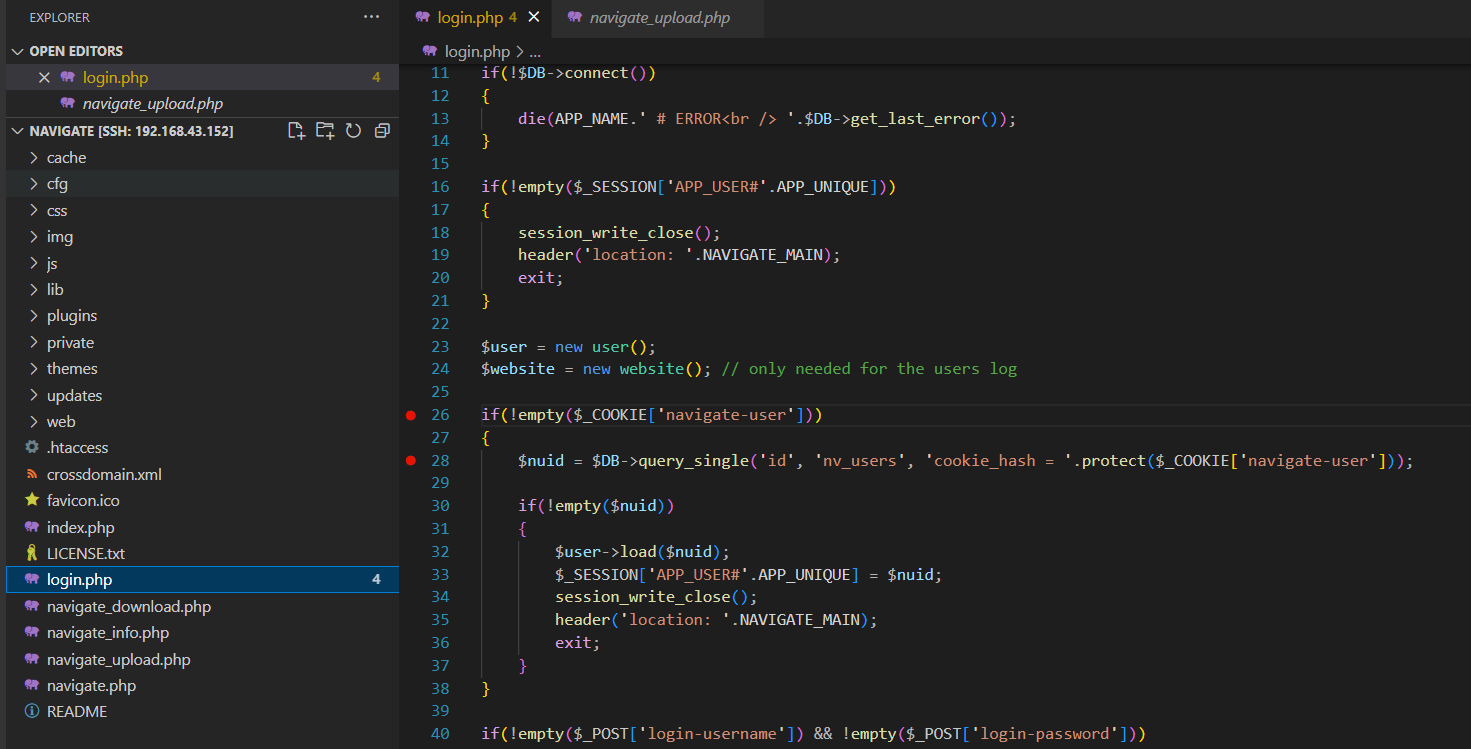


# Phân tích CVE 2018-17552(3)

## SQL Injection bypass Authentication

Ta đặt break tại dòng 26 trong file login.php

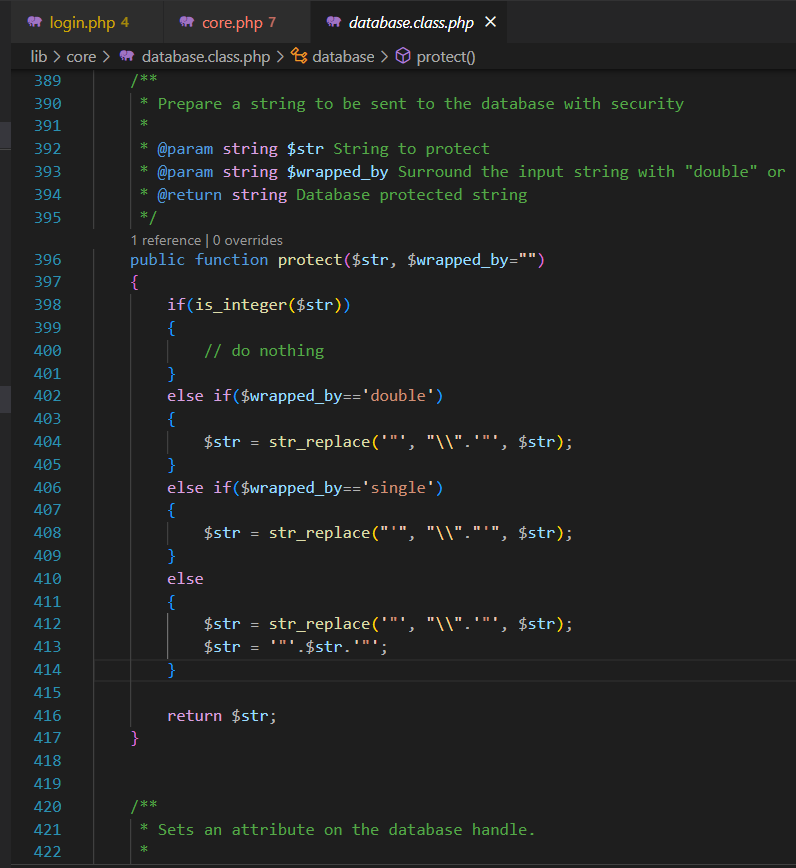
Tại đây navigate xử lý **$\_COOKIE['navigate-user']**

****

Và đặt bp tại hàm gây lỗi sqli

$DB->query\_single

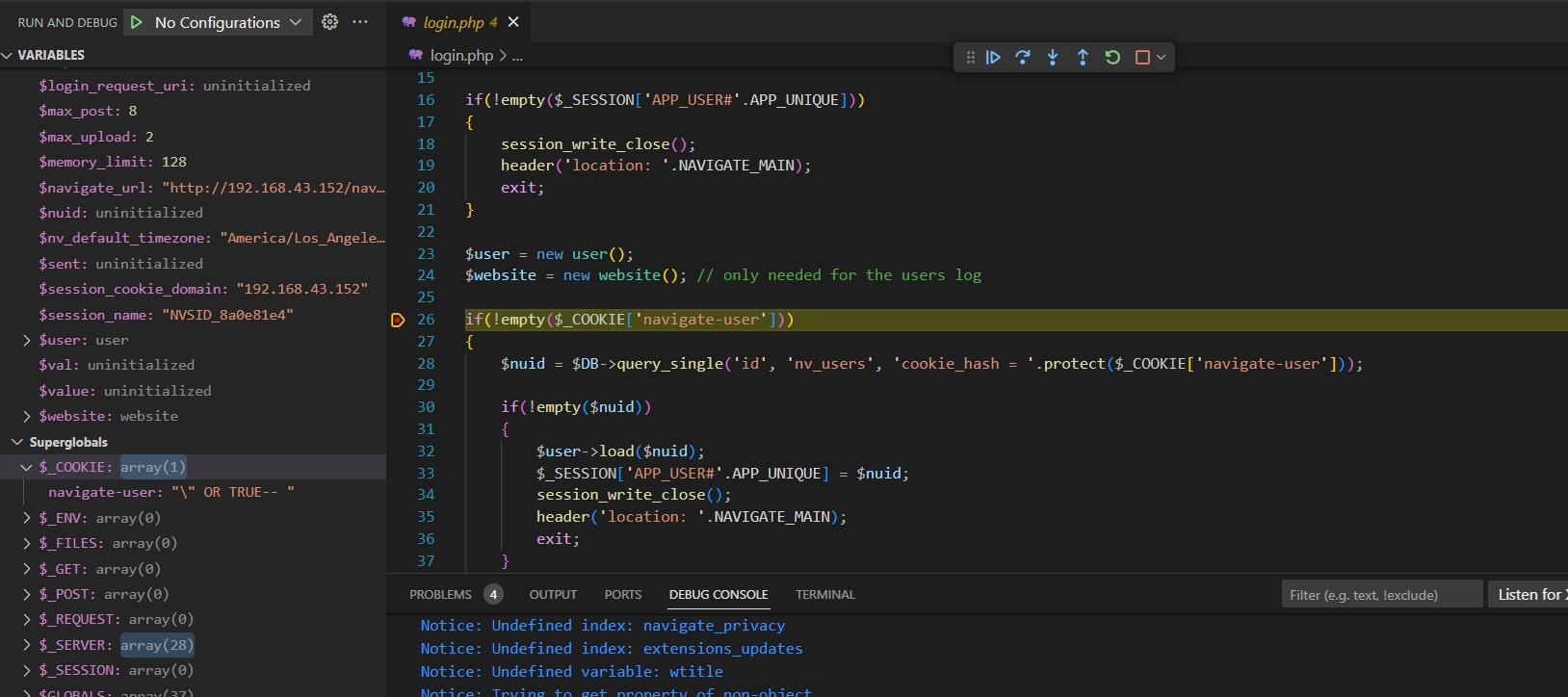
Theo mình kiểm tra thì hàm gây ra lỗi là hàm protect đã chưa filter được hết trường hợp gây lỗi sqli

’

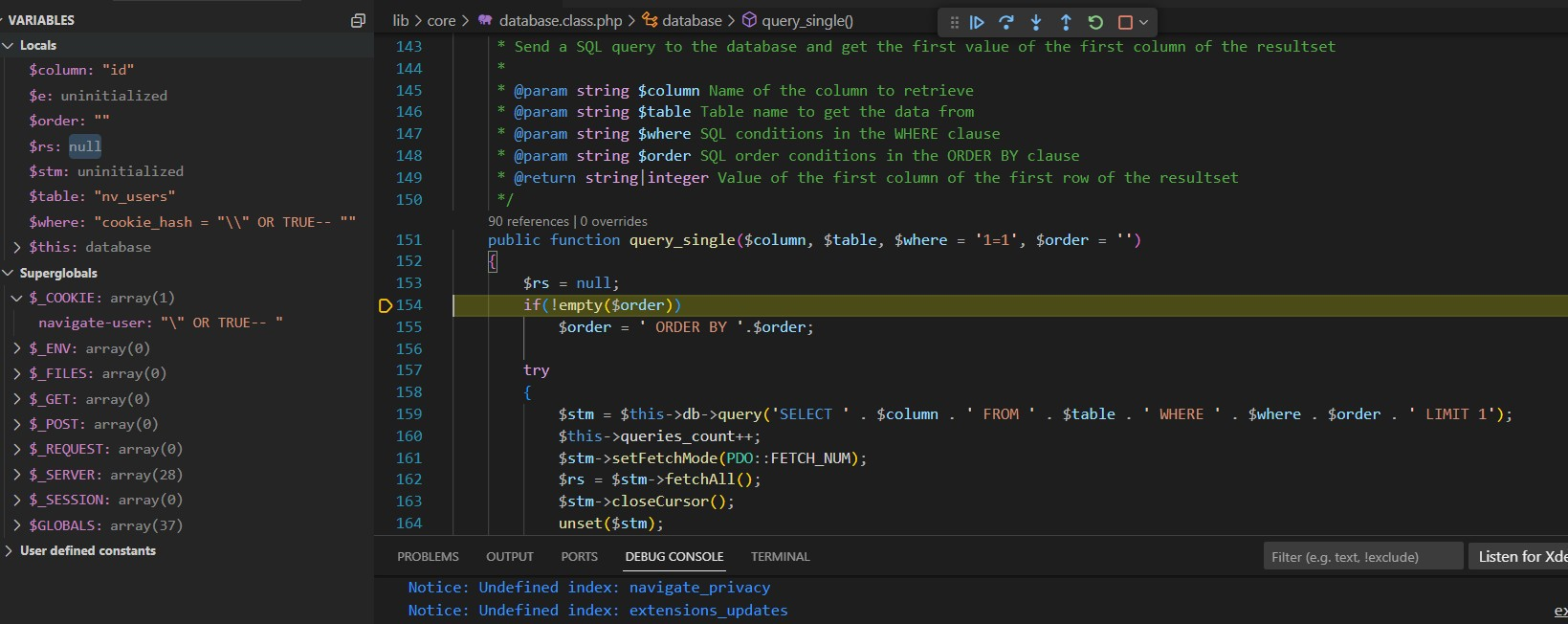
Vào thời điểm phát hiện ra CVE, ở phiên bản sau nhà phát triển đã thêm một dòng để filter các ký tự để tránh SQL attacks



Tiến hành debug



Mình cứ tưởng **cookie\_hash** là hash của cookie người dùng nhưng sau khi qua hàm protect thì cookie vẫn y nguyên chỉ bị filter đi các ký tự

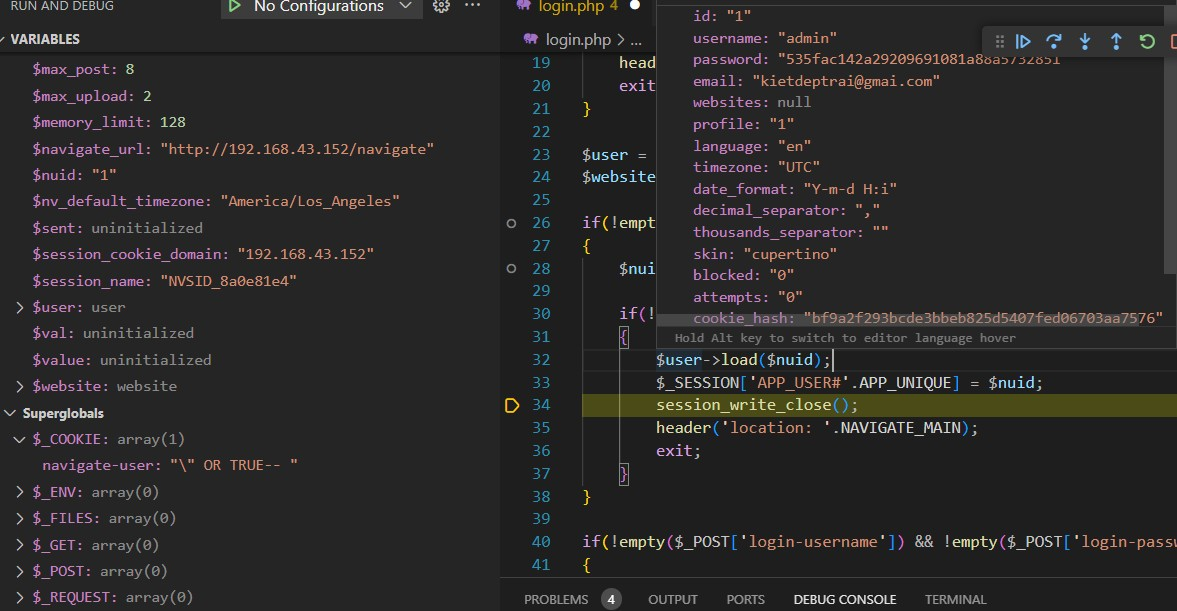


Vì vậy sau khi được nối vào chuỗi cookie\_hash và cho vào tham số thứ 3 (where) của **query\_single** một câu sql để lấy ra id người dùng từ db, ta có thể thực hiện sql injection tại đây.

Trả về của hàm là id của người dùng

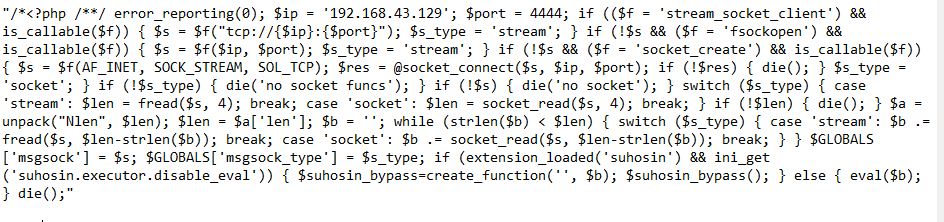
Tiếp tục sử dụng $user->load để lấy ra thông tin người dùng.

Vậy là ta có thể bypass login thông qua sql injection tham số Cookie



## File Upload (Payload webshell)

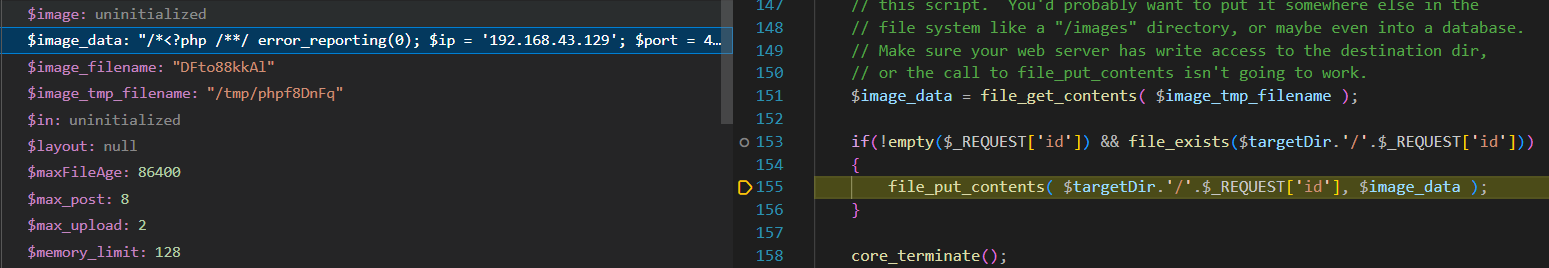
Có thể chuẩn bị đoạn php để thực hiện RCE và mở cổng listen 4444 để nhận shell



Hoặc một câu get shell đơn giản

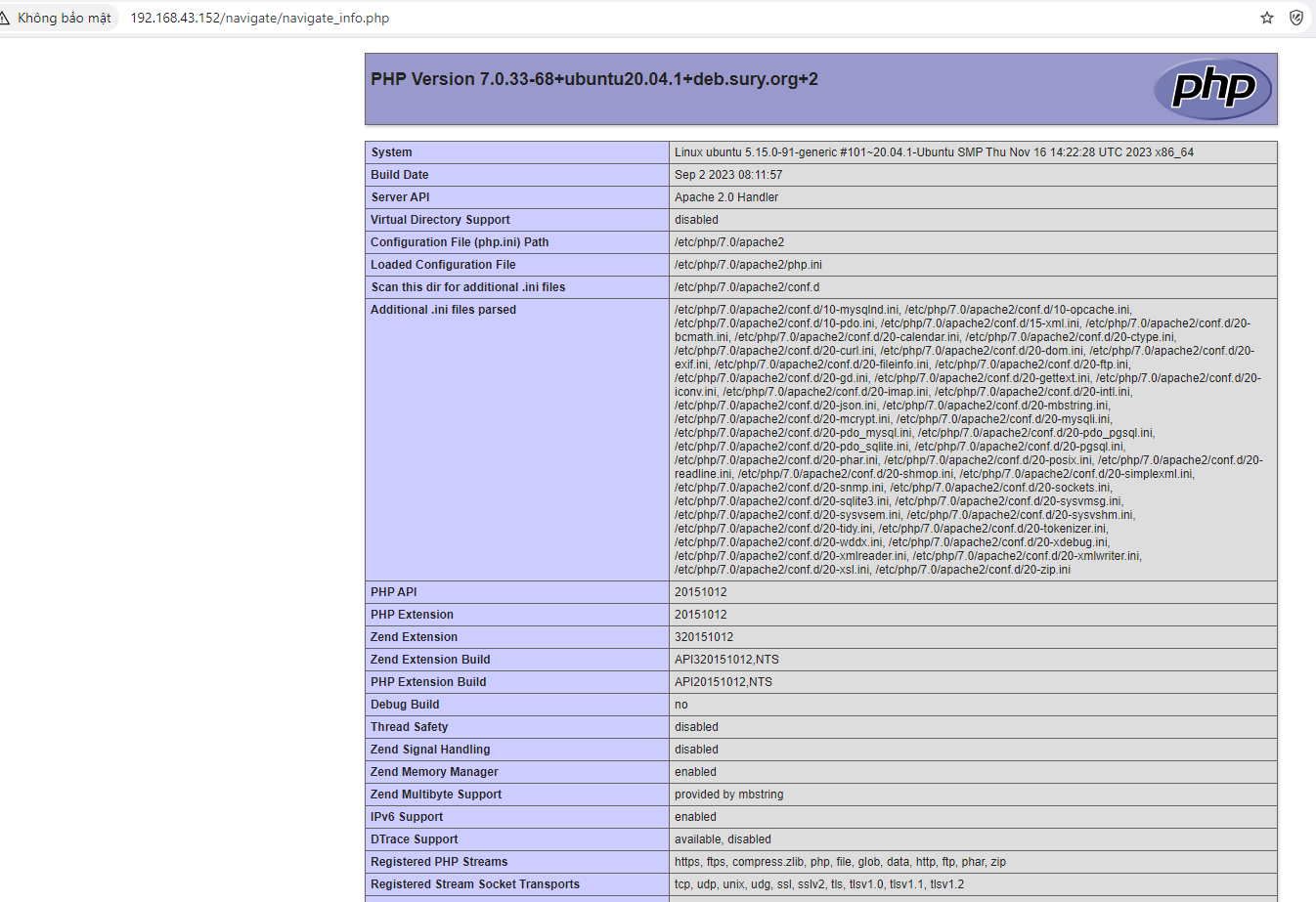


Sau đó lưu vào một file dưới dạng image để tiếp tục thực hiện **CVE 2018-17553**

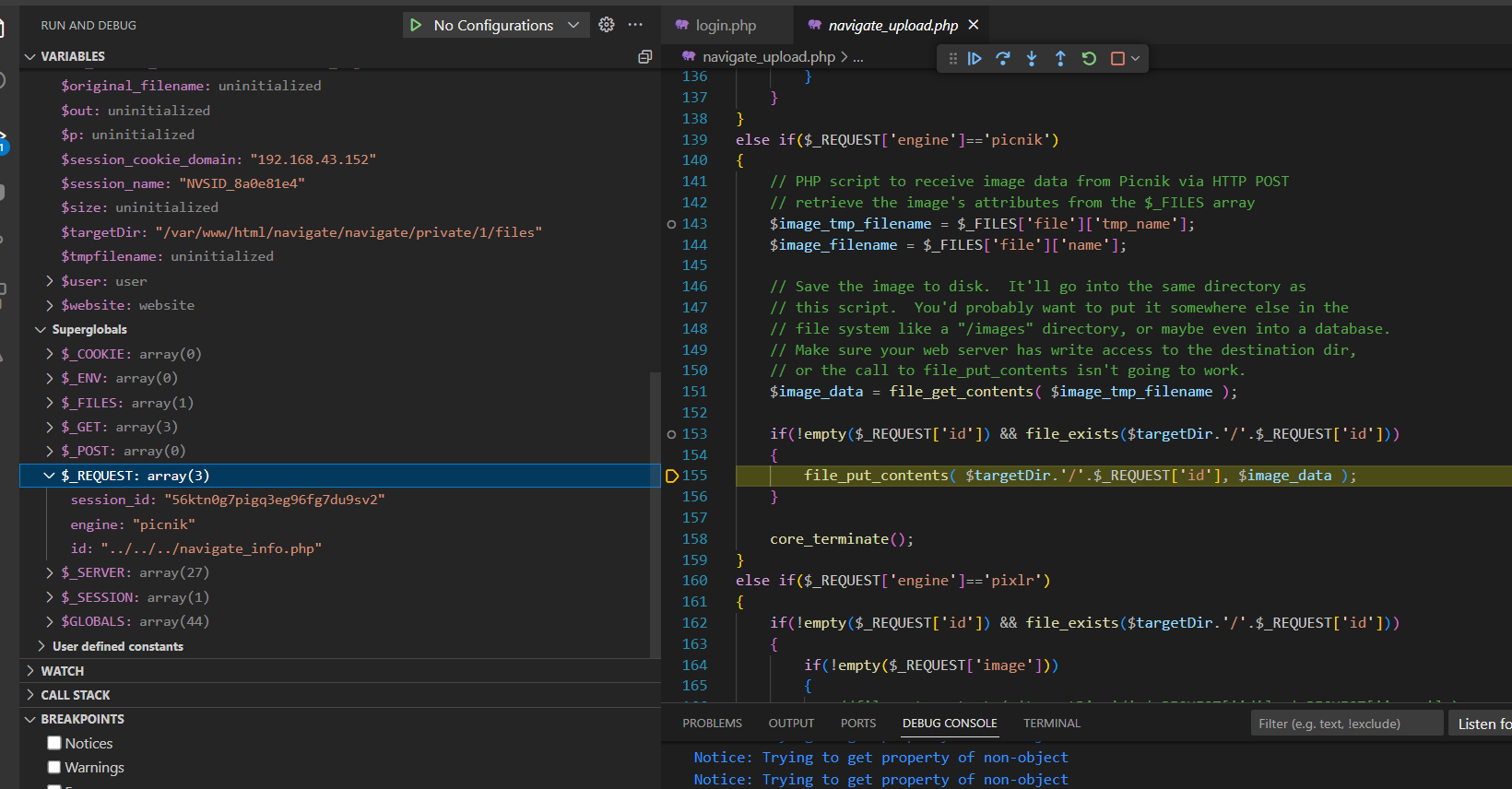


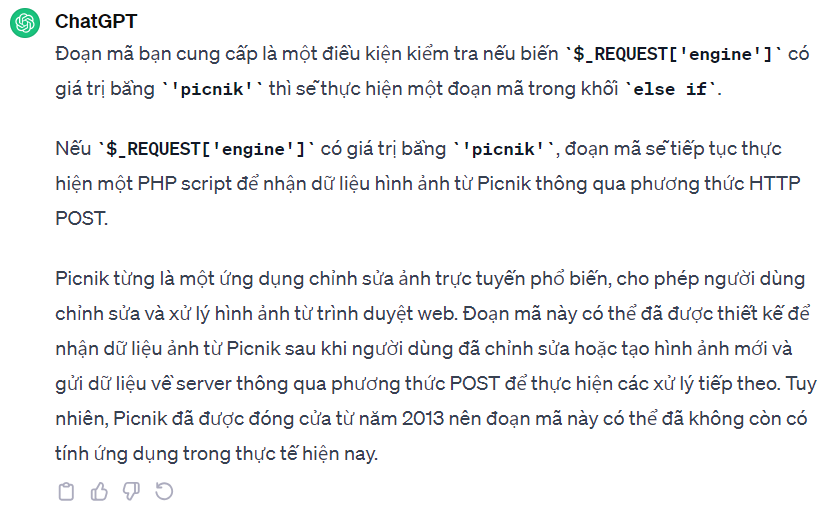
## Path traversal

Bình thường trang navigate\_info.php có thể truy cập không cần xác thực vì vậy ta có thể tạo webshell tại trang này.



Đặt bp tại dòng xử lý tham số request **Engine** là **picnik** tại file **navigate\_info.php**





Mình thắc mắc picnik là gì, thì hóa ra nó là một ứng dụng chỉnh sửa ảnh trực tuyến, nhưng đã đóng từ 2013. Hàm này đã được Navigate CMS loại bỏ ở các bản cập nhật sau và thay xử lý IF ELSE thành switch case

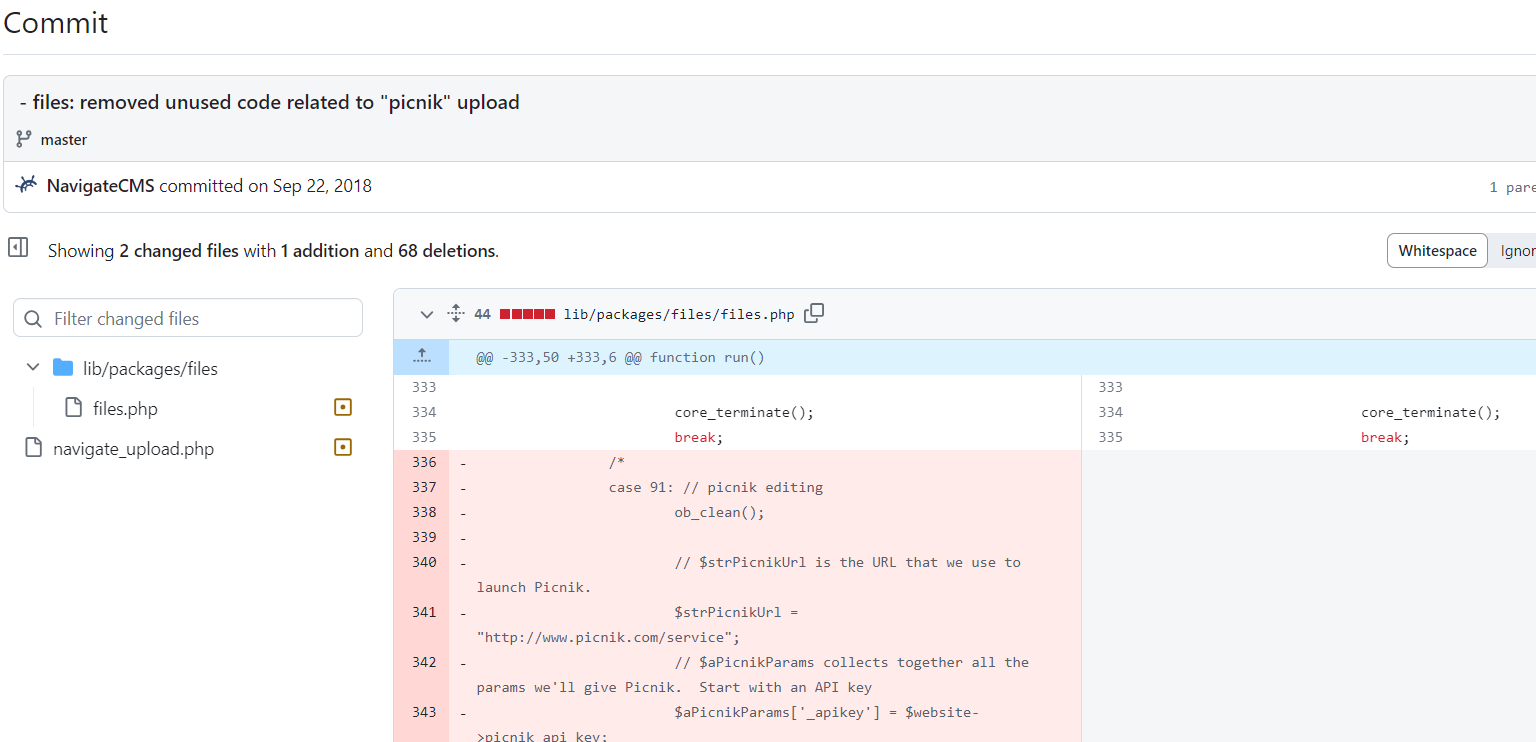
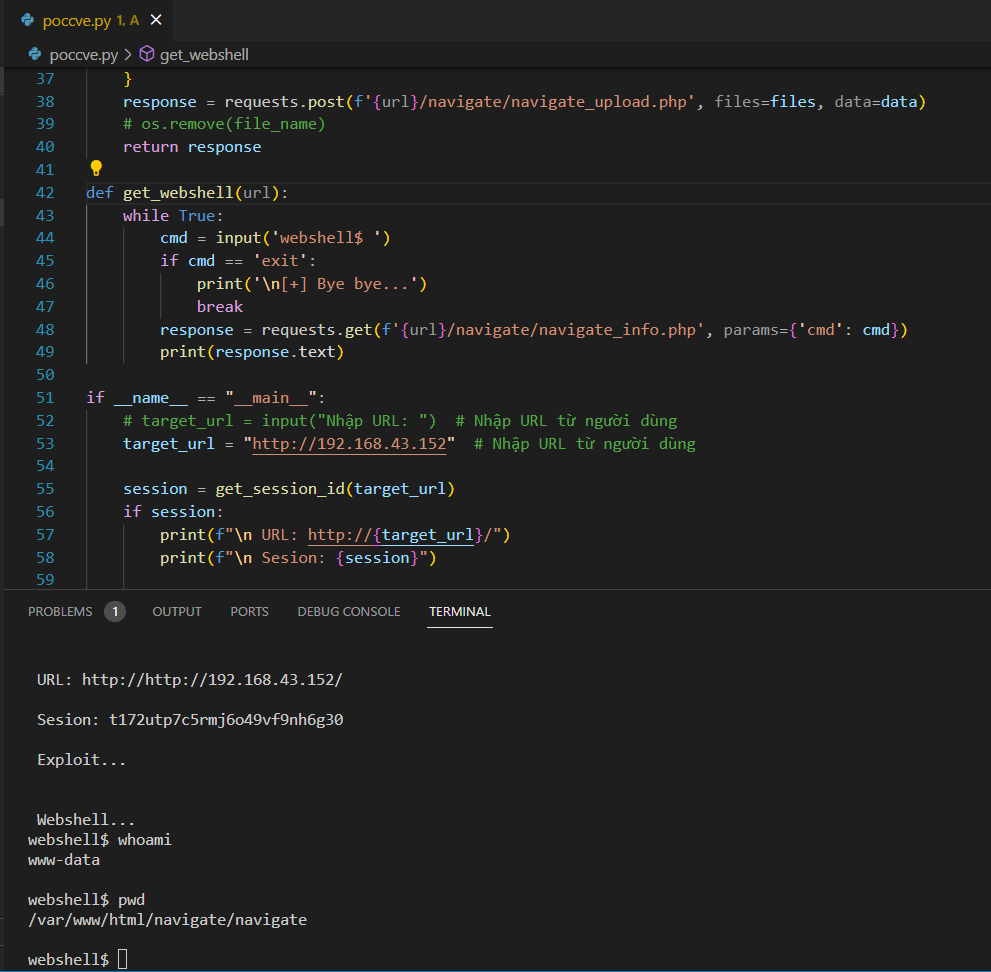




Image chứa payload được hàm **file\_put\_contents** lưu **content** tới vị trí */var/www/html/navigate/navigate/private/1/files../../../navigate\_info.php*

Như vậy nội dung file navigate\_info.php được chuyển thành payload webshell trong image chúng ta upload bên trên

Dựa trên ý tưởng trên mình đã viết lại PoC bằng python để tự động attack CVE và tạo webshell



Hoặc ta có thể thay đổi một chút payload để tạo webshell trên browser

