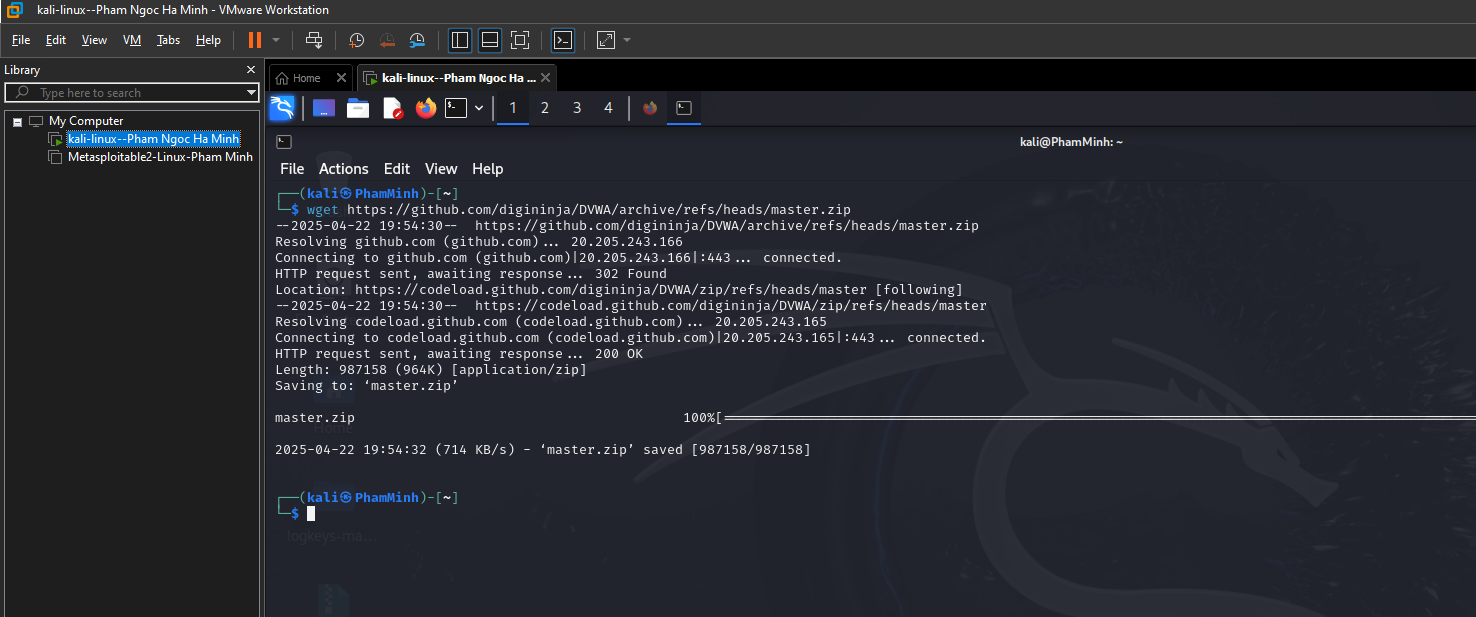
**BÁO CÁO BÀI THỰC HÀNH SỐ 9**

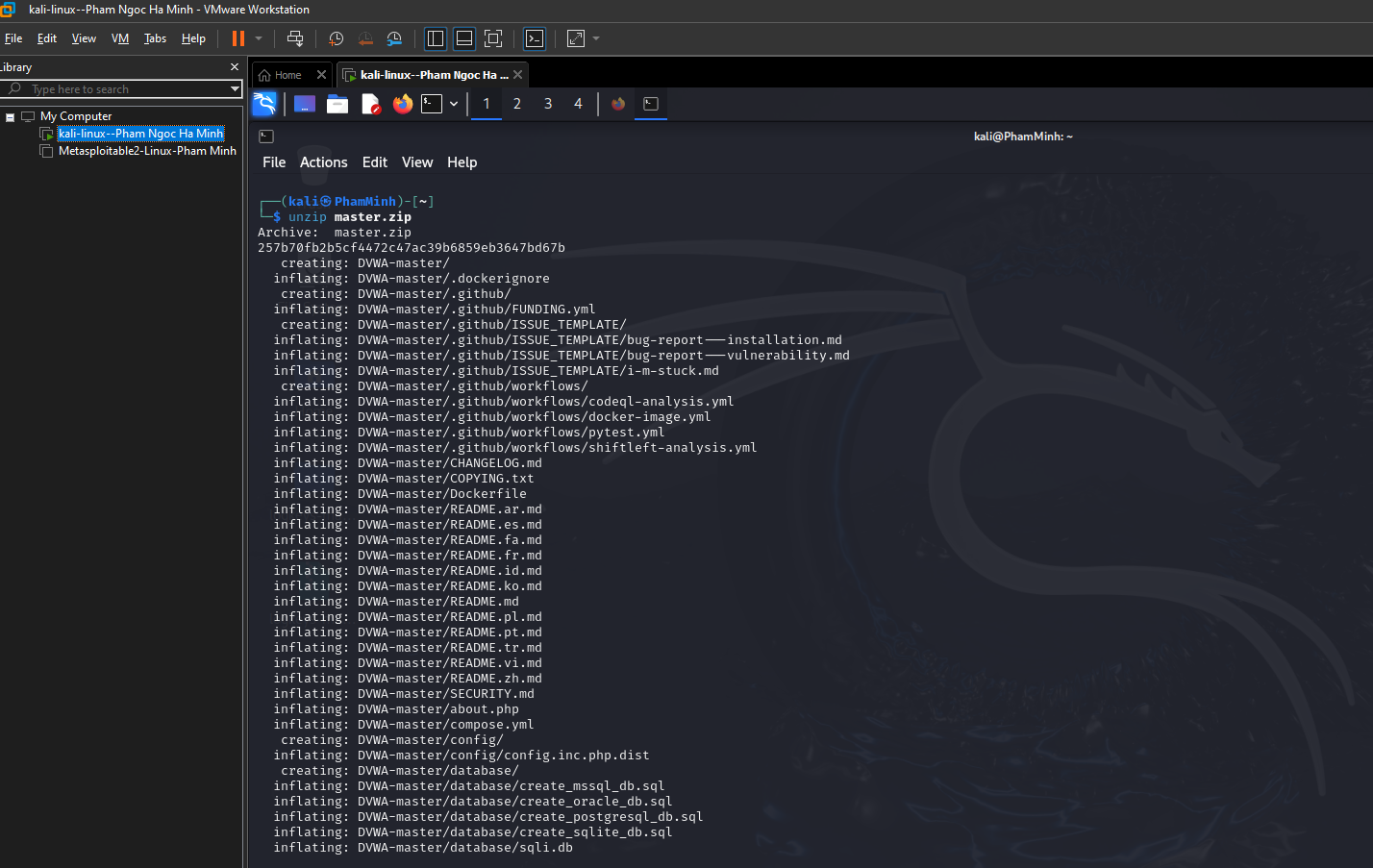
Họ Tên : Phạm Ngọc Hà Minh

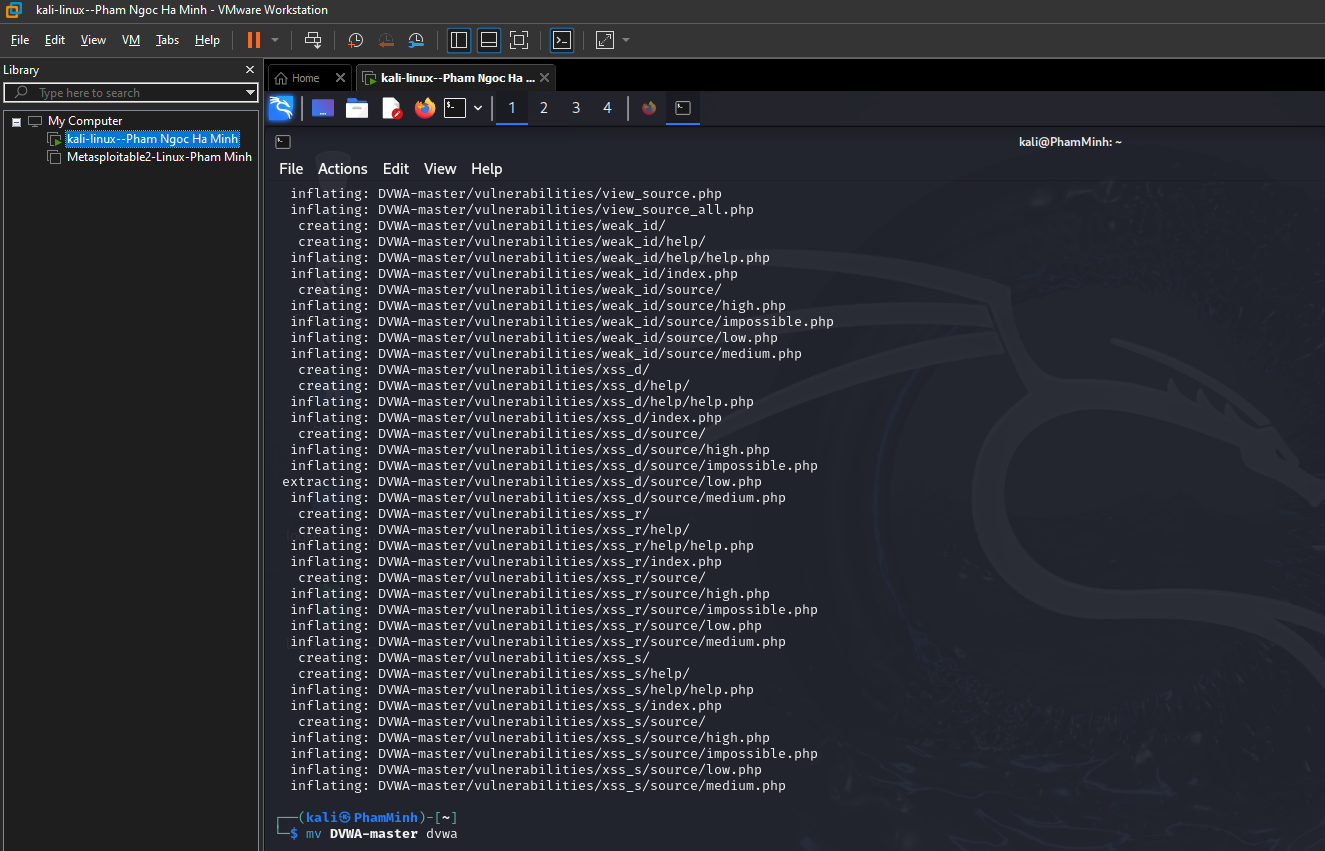
MSSV: 1050080061

Cài đặt

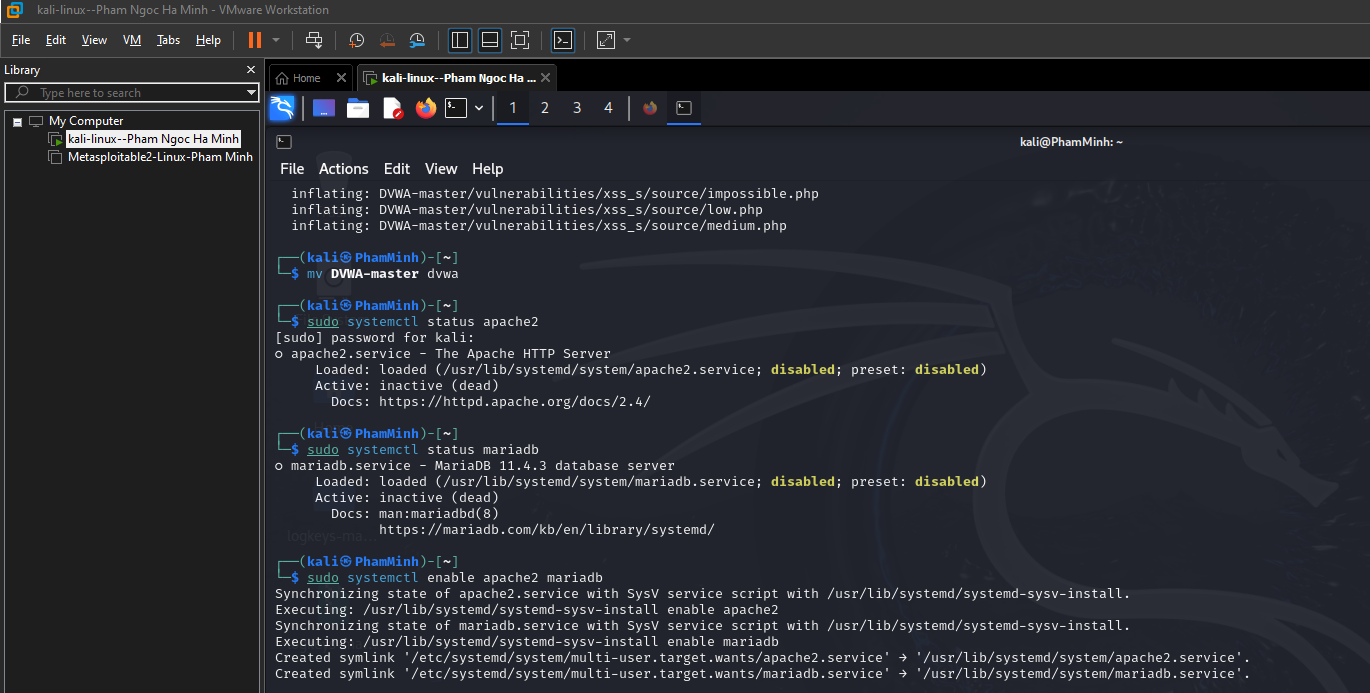
Bước 1: Download mã nguồn

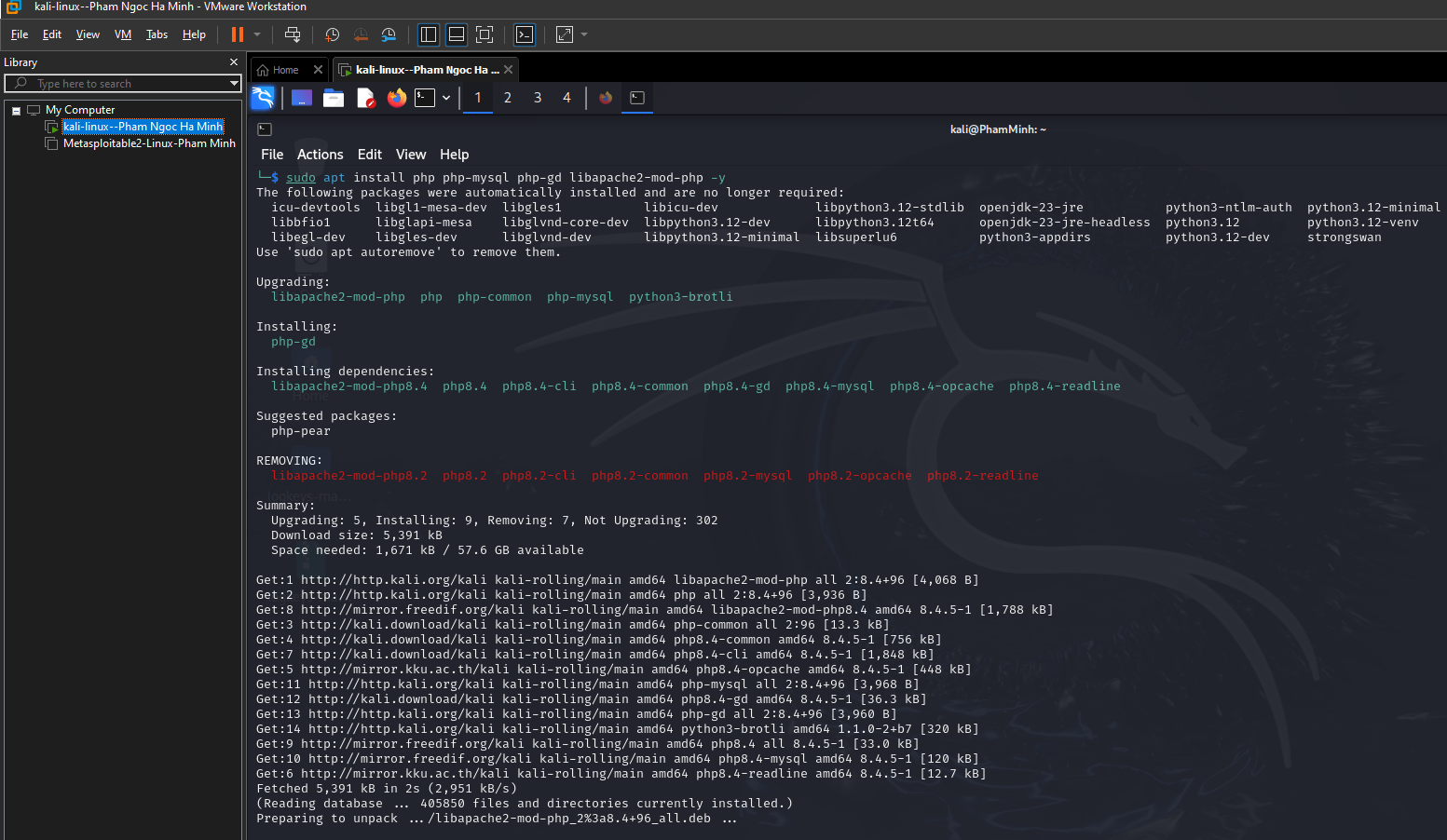


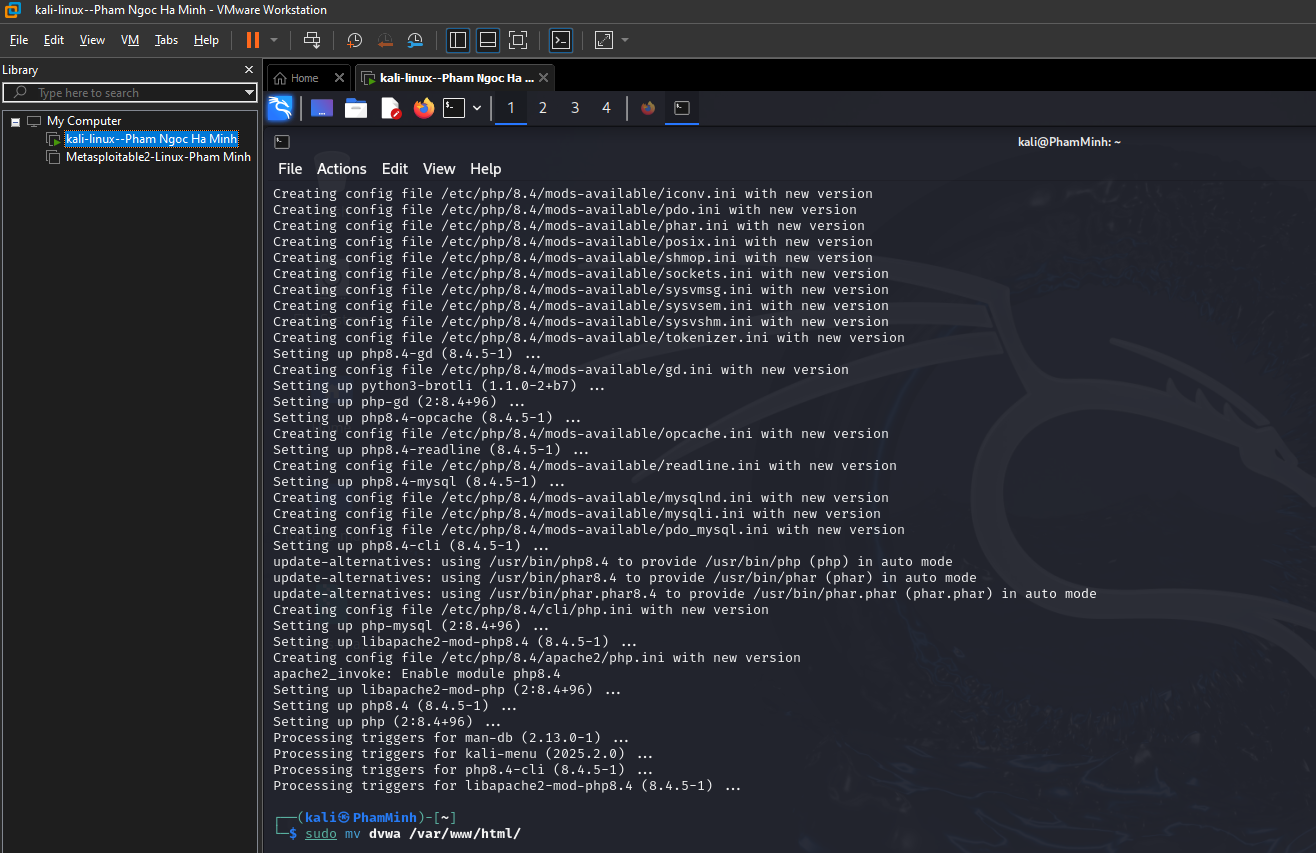
Bước 2: Giải nén file đã download được và đổi tên thành “dvwa”



Bước 3: Sao chép thư mục sau khi đã giải nén vào thư mục webroot của Web Server







Bước 4: Tạo một CSDL có tên là “dvwa”

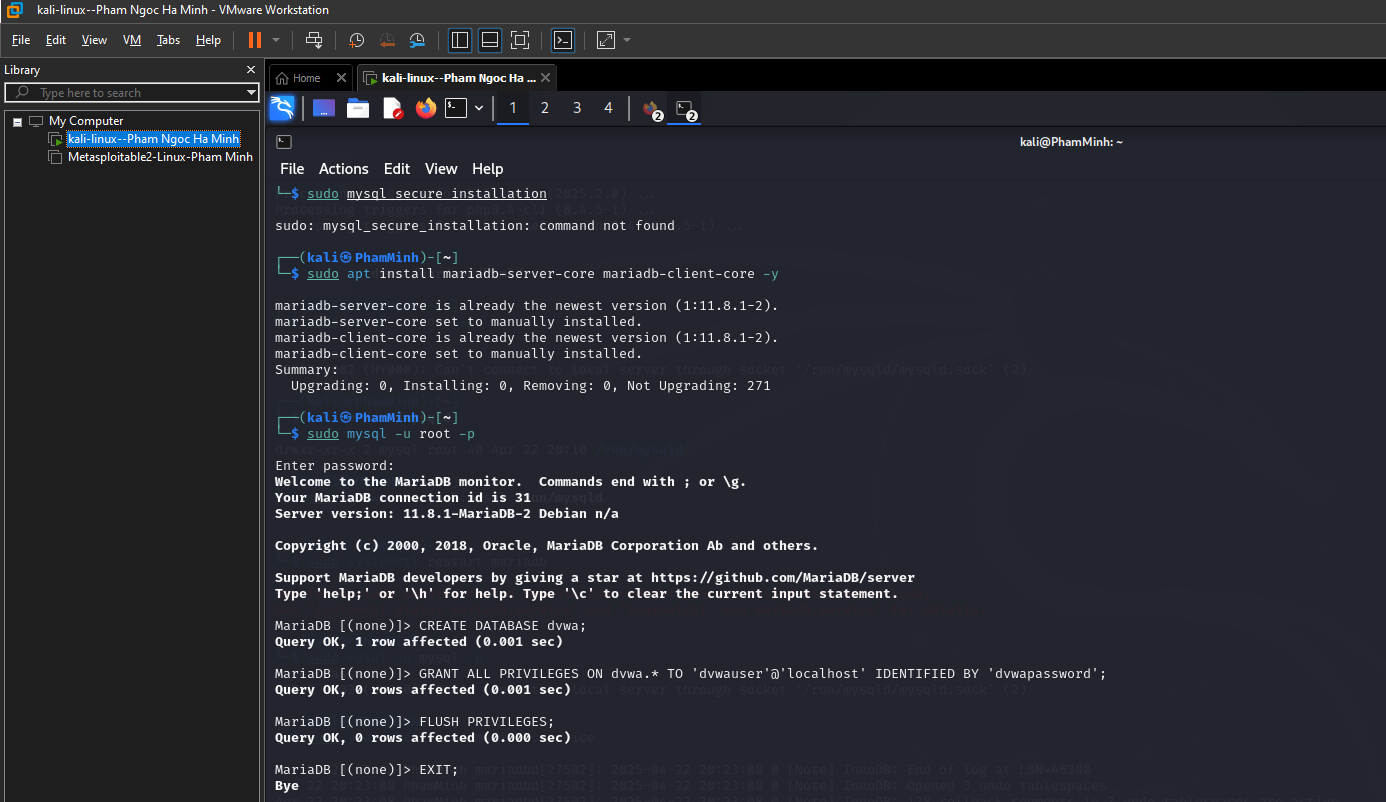
Tạo CSDL và user:

CREATE DATABASE dvwa;

GRANT ALL PRIVILEGES ON dvwa.\* TO 'dvwauser'@'localhost' IDENTIFIED BY 'dvwap assword';

FLUSH PRIVILEGES;

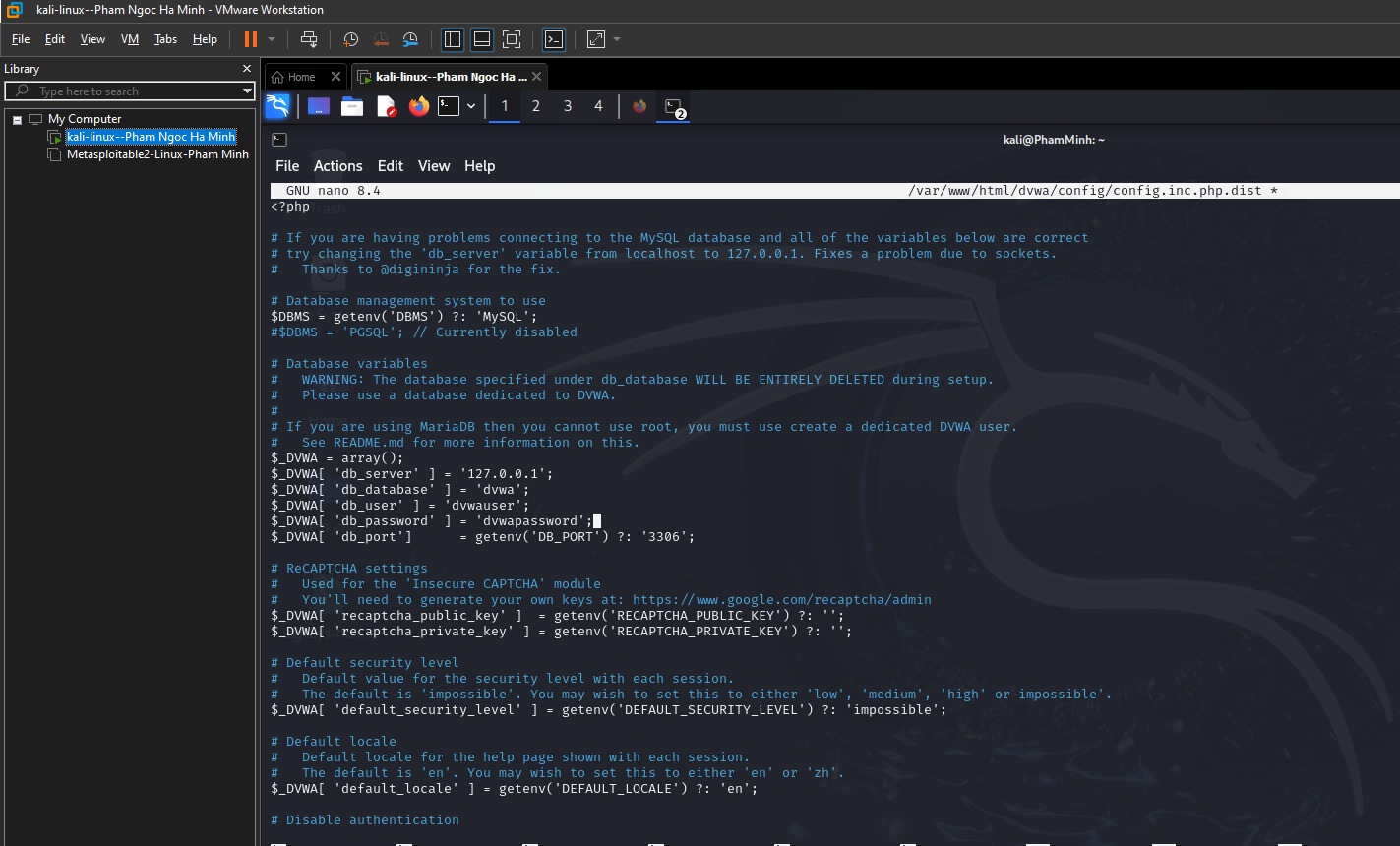
EXIT;



Bước 5: sửa nội dung file dvwa/config/config.inc.php

Cấu hình DVWA

sudo nano /var/www/html/dvwa/config/config.inc.php.dist



Lưu lại và đổi tên file:

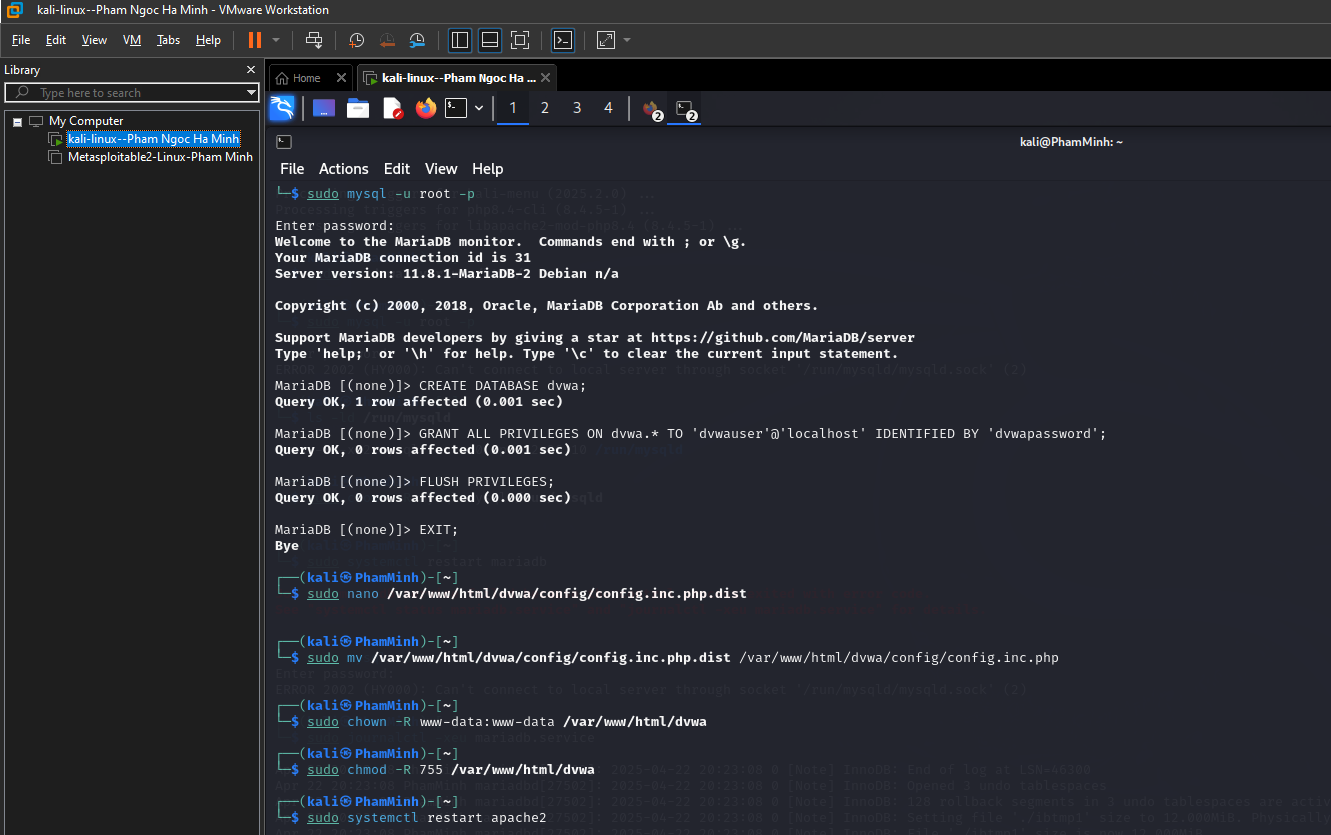
sudo mv /var/www/html/dvwa/config/config.inc.php.dist /var/www/html/dvwa/config/config.inc.php

Cấp quyền thư mục

sudo chown -R www-data:www-data /var/www/html/dvwa

sudo chmod -R 755 /var/www/html/dvwa

Khởi động lại Apache: sudo systemctl restart apache2



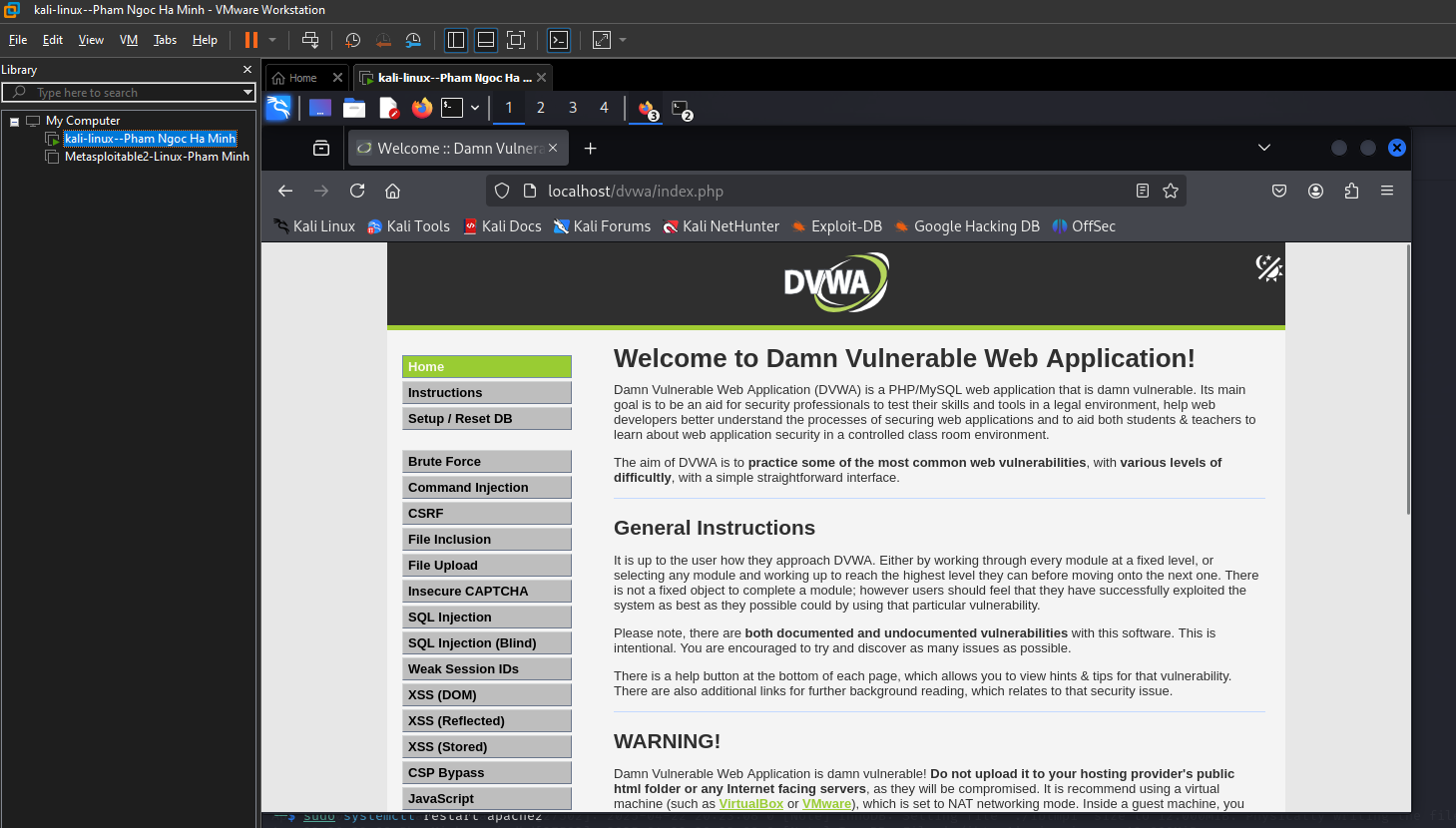
Bước 6: Sau khi cài đặt thành công

Truy cập DVWA: Mở trình duyệt và truy cập: http://localhost/dvwa/setup.php

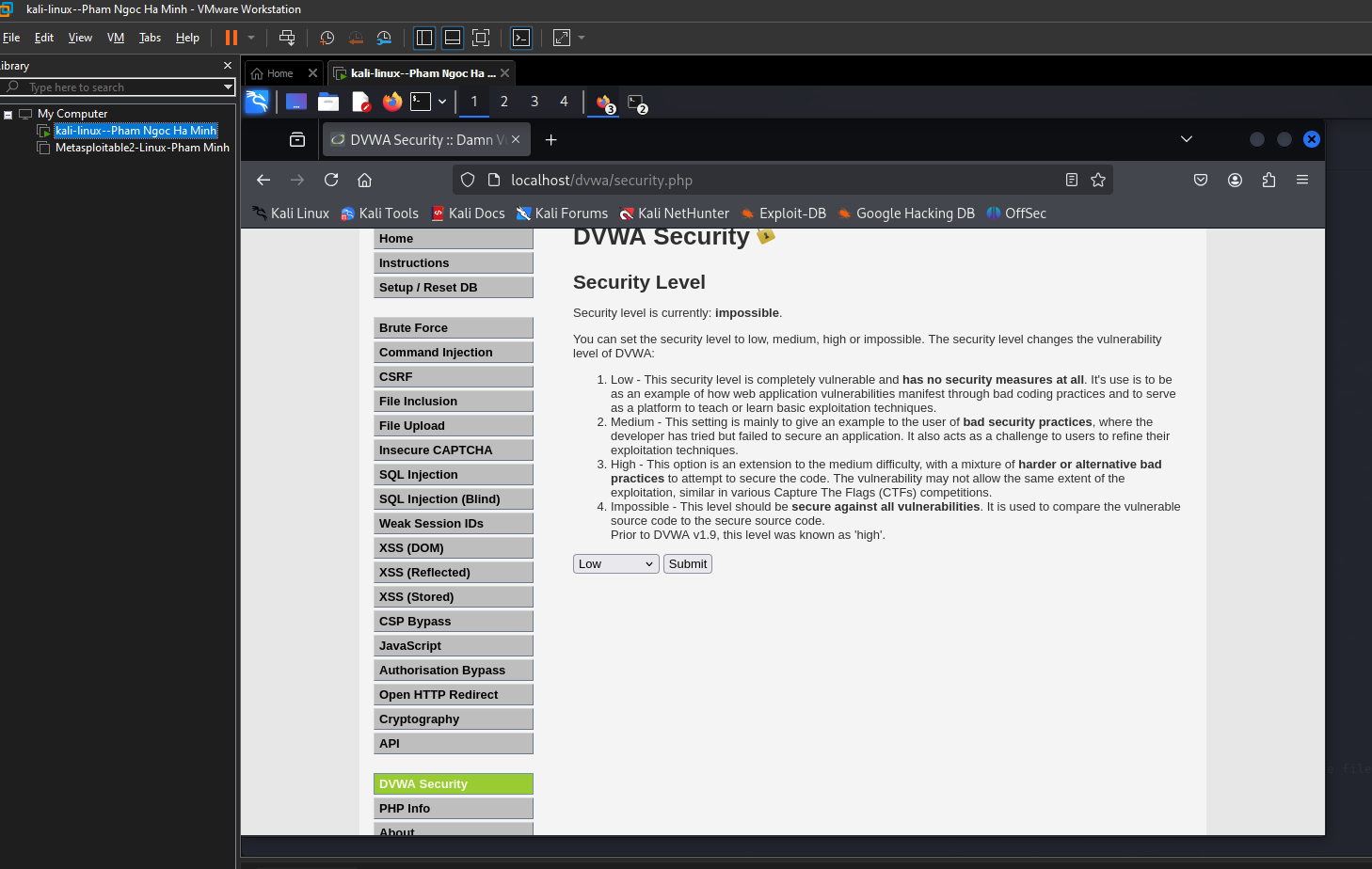
Đăng nhập DVWA

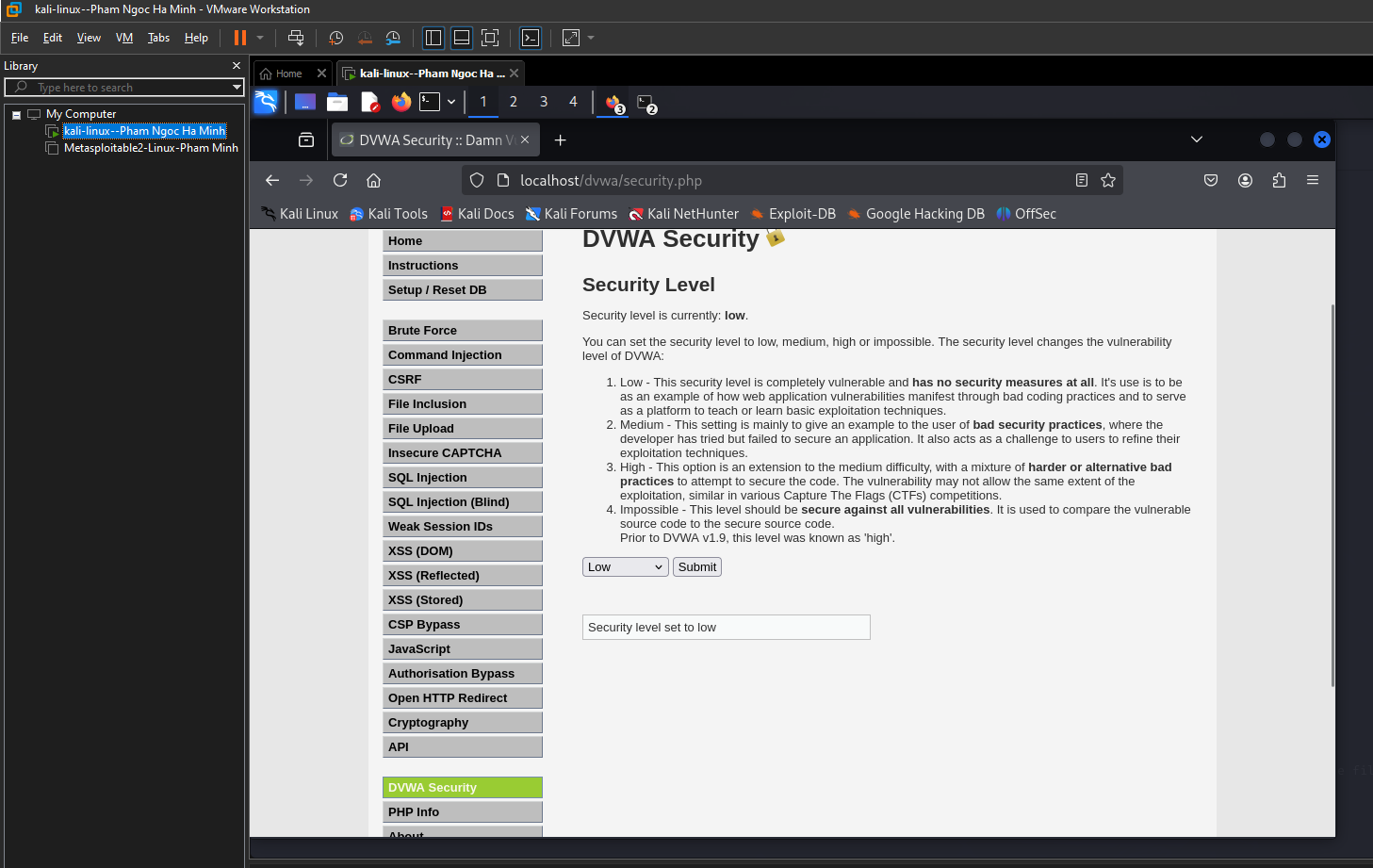
Username: admin

Password: password



Bước 7: Thiết lập giá trị cho DVWA Security là “Low” để phần bắt đầu luyện tập dễ dàng hơn





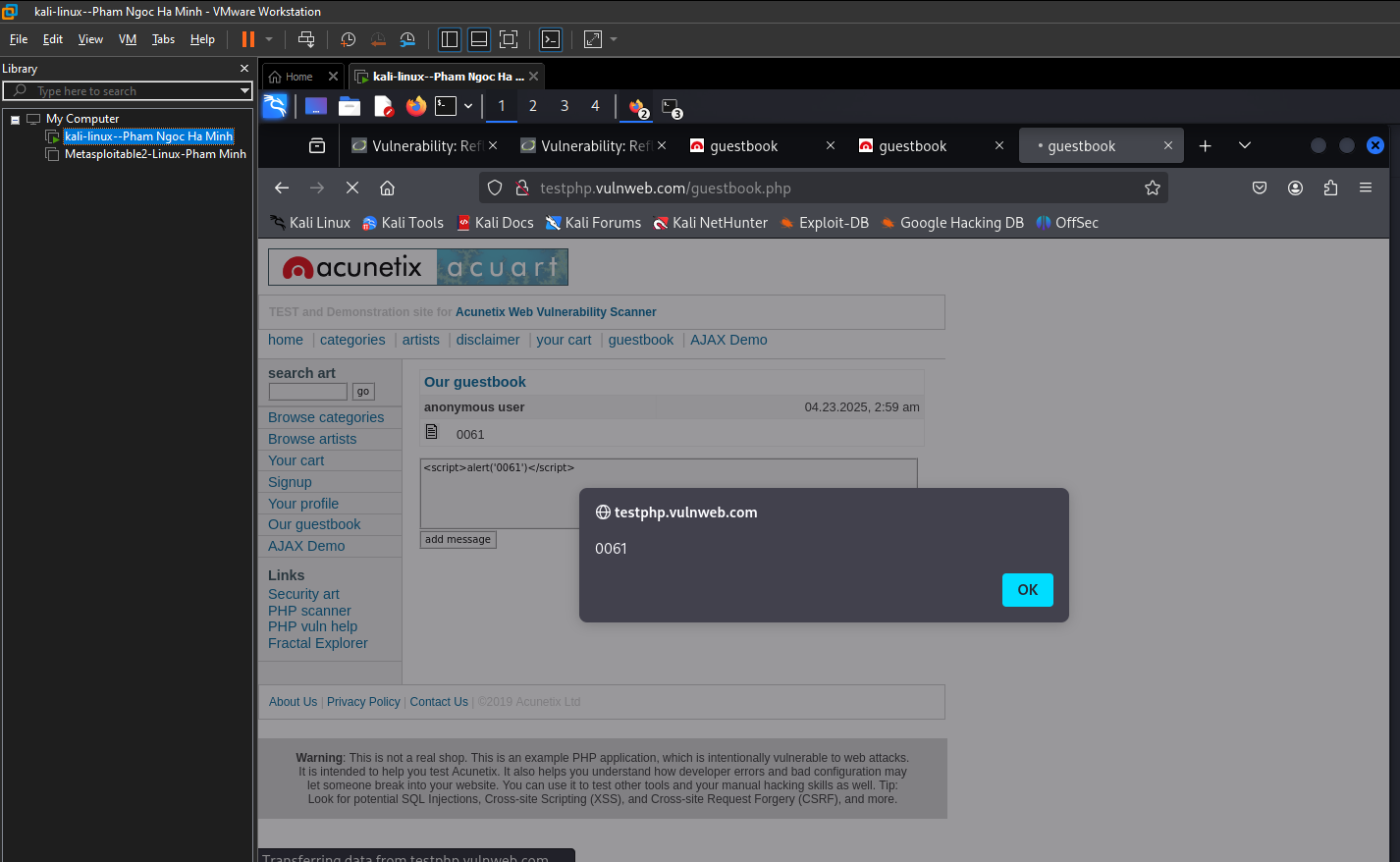
**Câu 1:** **Thực hiện kiểm thử lỗ hổng XSS để thực thi đoạn mã hiển thị thông báo có chứa 4 số cuối MSSV.**

Dùng 2 trang web <http://localhost/dvwa/vulnerabilities/xss_r/> và http://testphp.vulnweb.com/guestbook.php

Thực hiện kiểm thử lỗ hổng XSS để thực thi đoạn mã hiển thị thông báo có chứa 4 số cuối MSSV.

Hãy giải thích:

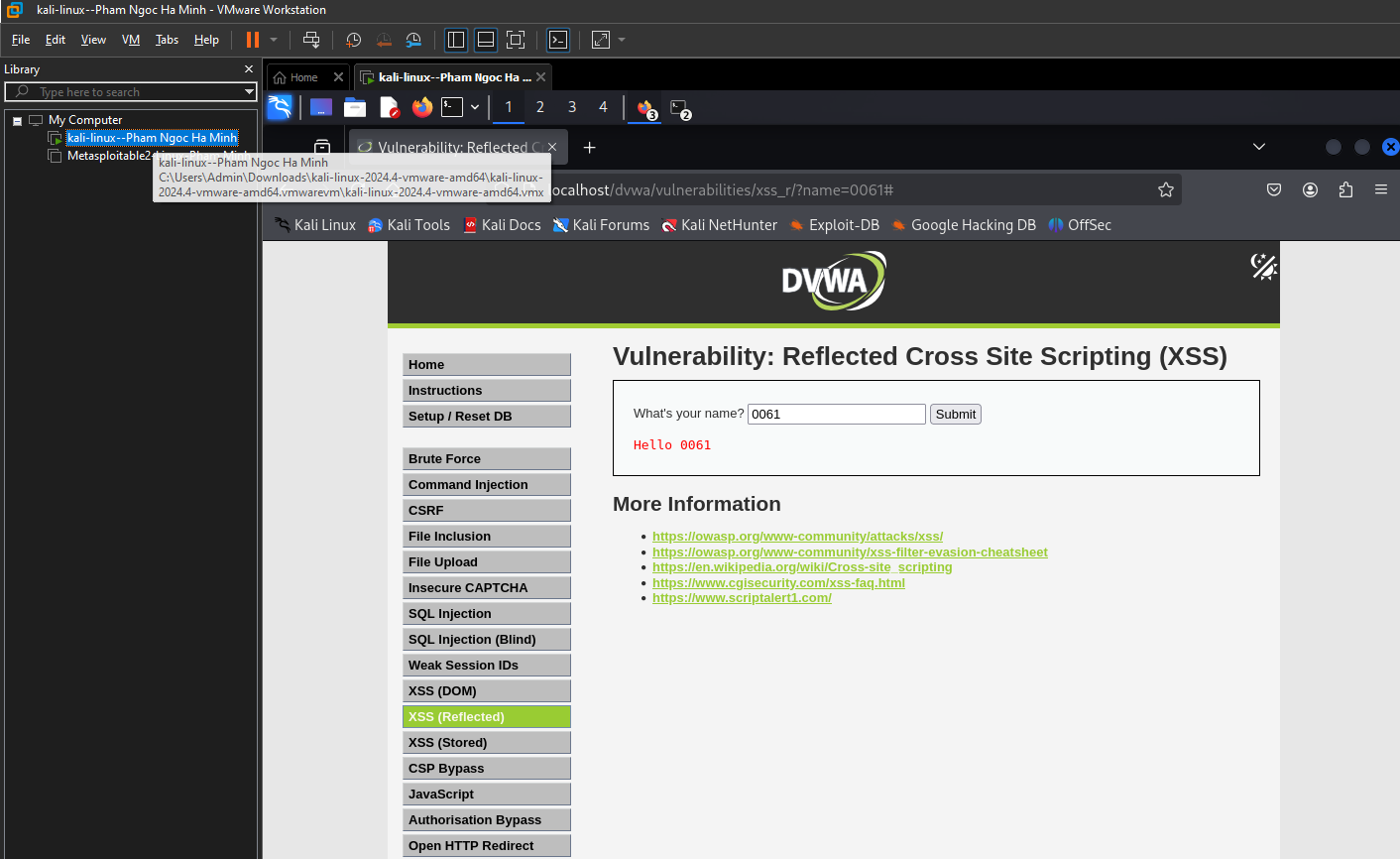
- Xác định các tham số đầu vào: 0.25 điểm



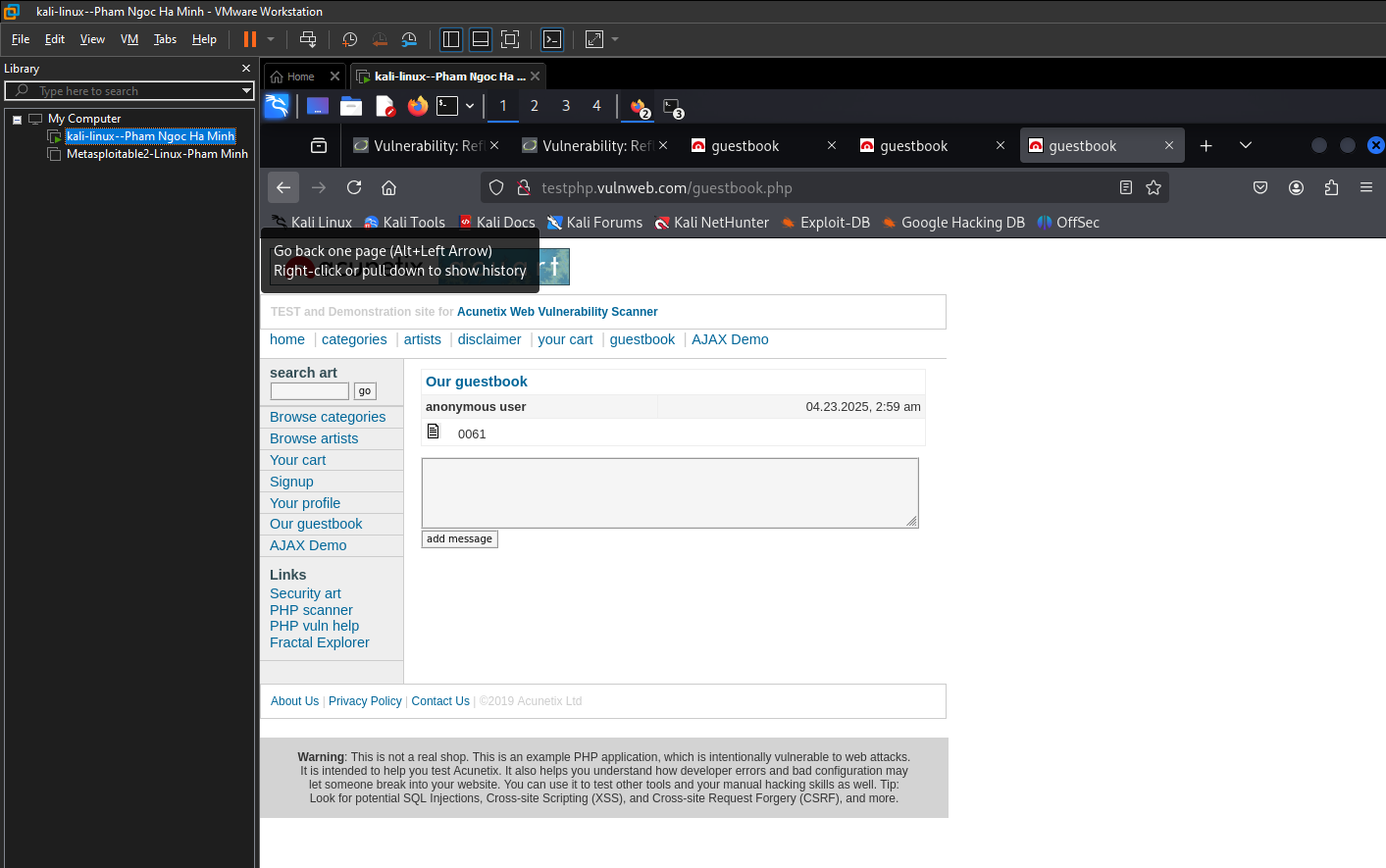
Tham số đầu vào là 0061

- Xác định giá trị đầu vào xuất hiện trong trang kết quả: 0.25 điểm

<http://localhost/dvwa/vulnerabilities/xss_r/?name=0061#>

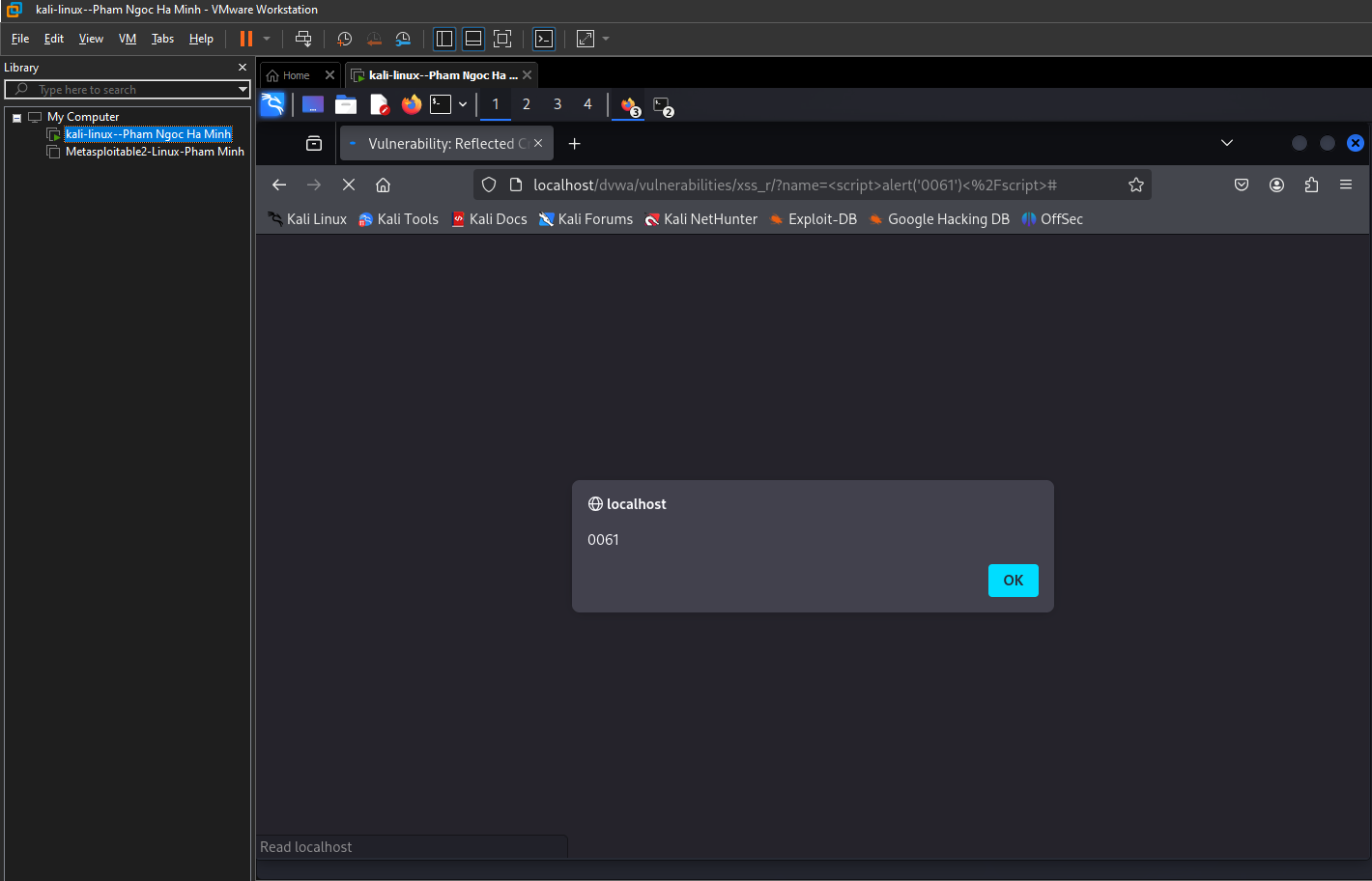


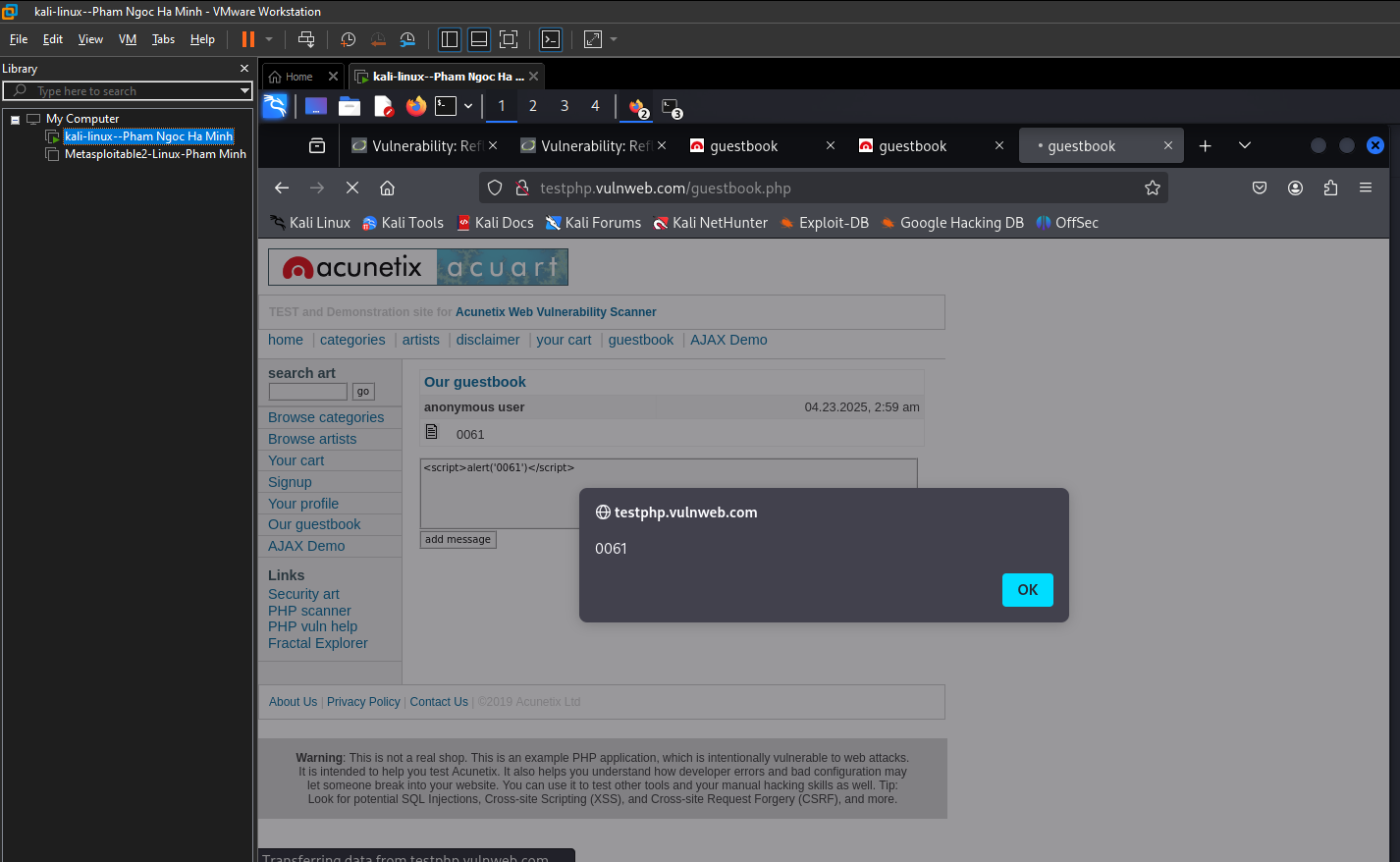
<http://testphp.vulnweb.com/guestbook.php>



- Các bước kiểm thử và phán đoán: 0-1 điểm theo các bước kiểm thử và logic trong phán đoán

Nhập đoạn mã XSS : <script>alert('0061')</script> hoặc truy cập http://localhost/dvwa/vulnerabilities/xss\_r/?name=<script>alert('0061')</script>

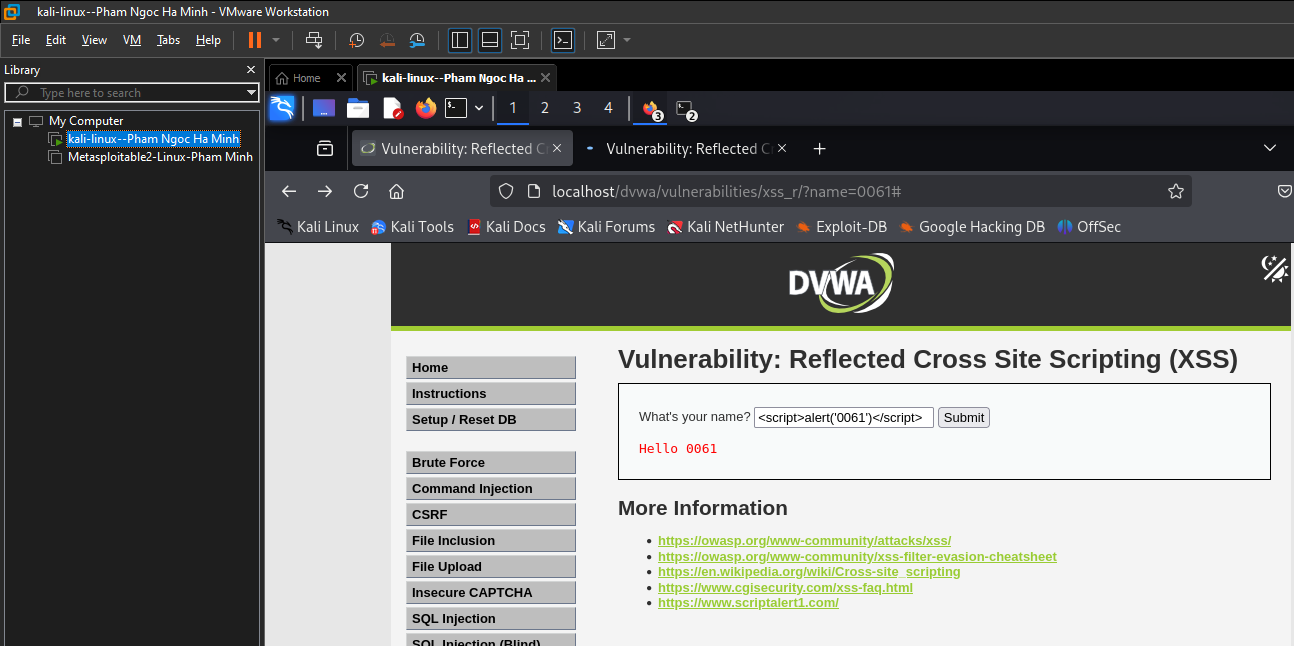


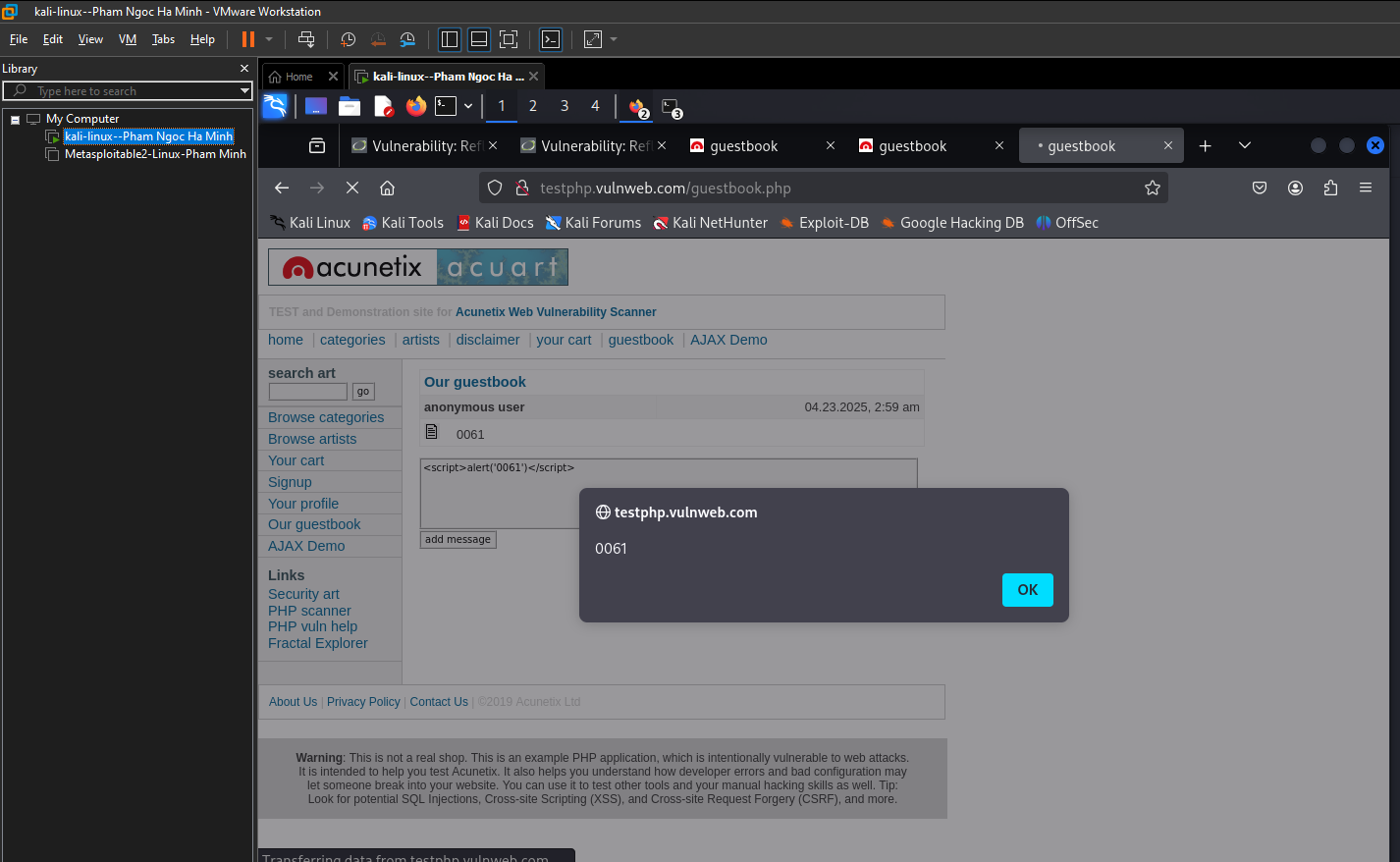


* Nếu trình duyệt hiển thị hộp thoại JavaScript chứa “0061”, đã khai thác thành công lỗ hổng XSS.
* Vì giá trị người dùng nhập (<script>...</script>) không được mã hóa hoặc lọc → JavaScript được thực thi.

- Giá trị kiểm thử cho thấy lỗ hổng và giải thích: 0.5 điểm

<script>alert('0061')</script>





**Giải thích:**

* Đây là một đoạn mã JavaScript hợp lệ.
* Khi chèn vào input và trang web không lọc nội dung → mã được thực thi trên trình duyệt.
* Lỗ hổng nằm ở việc giá trị người dùng nhập được phản hồi trực tiếp lên trang HTML mà không qua xử lý hay mã hóa an toàn, dẫn đến XSS Reflected.

**Câu 2: Thực hiện kiểm thử lỗ hổng XSS để thực thi đoạn mã hiển thị thông báo có chứa tên của sinh viên.**

Dùng 2 trang web <http://localhost/dvwa/vulnerabilities/xss_r/> và http://testphp.vulnweb.com/guestbook.php

Thực hiện kiểm thử lỗ hổng XSS để thực thi đoạn mã hiển thị thông báo có chứa tên của sinh viên.

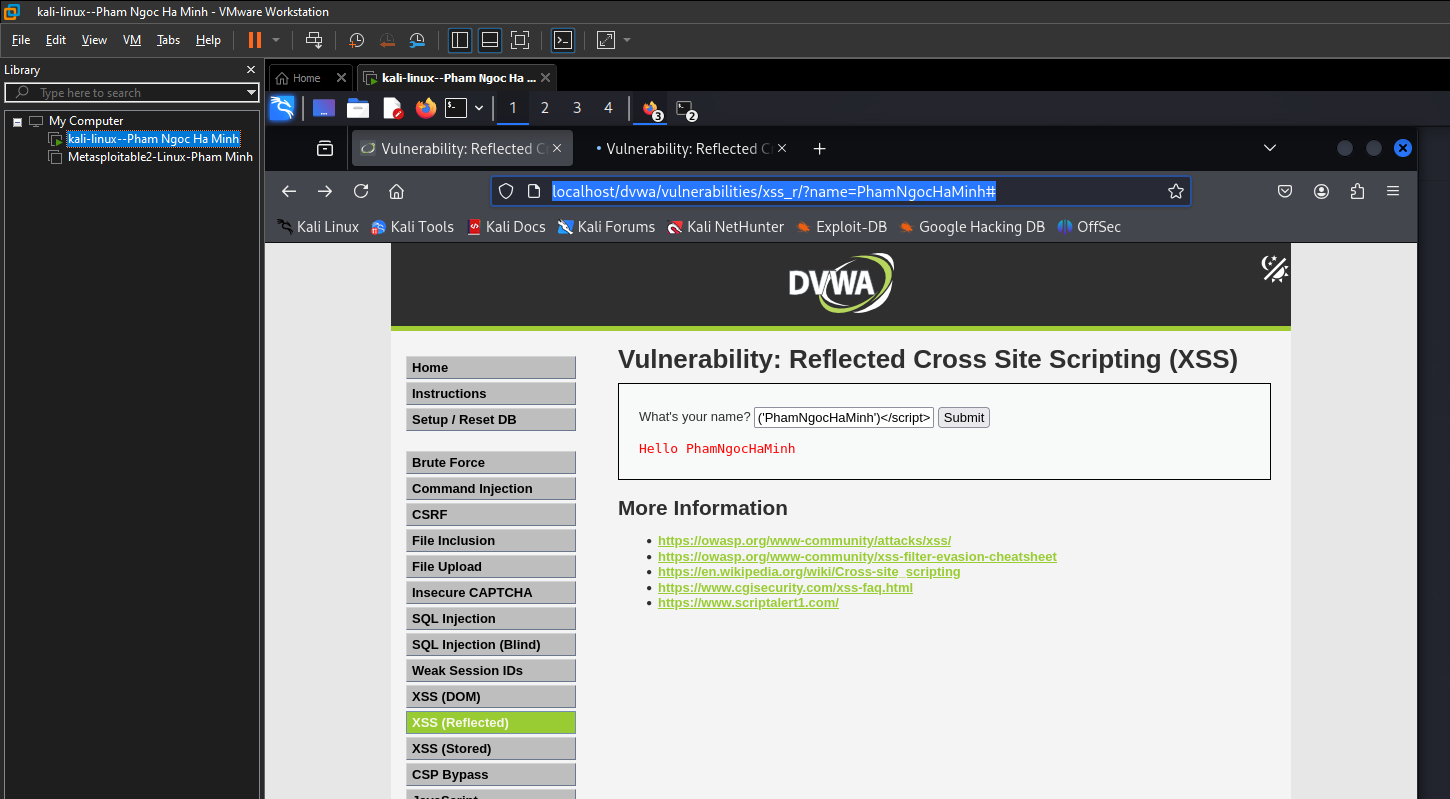
Điểm cho mỗi tham số đầu vào đã kiểm thử được:

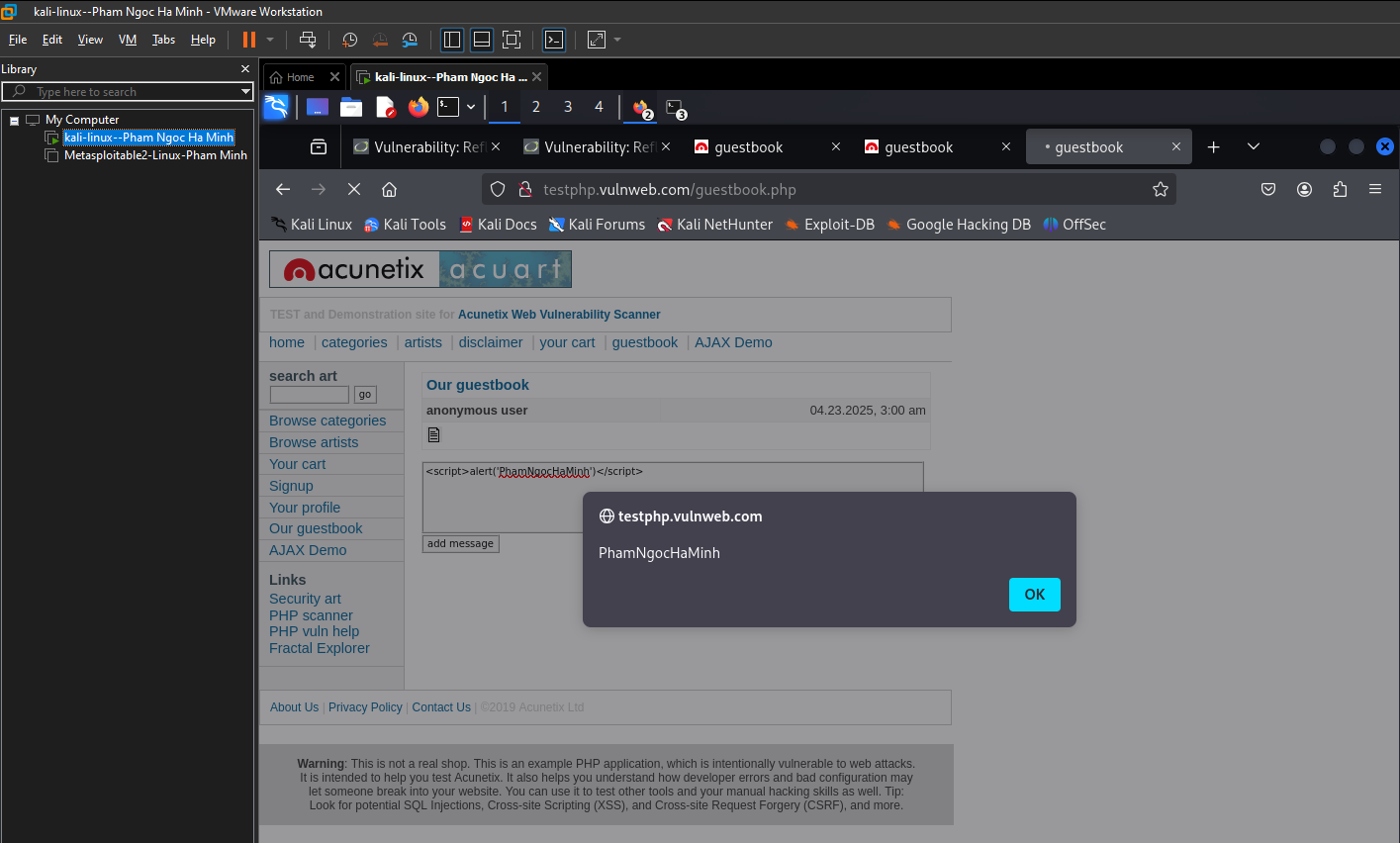
- Xác định các tham số đầu vào: 0.25 điểm

Tham số đầu vào : PhamNgocHaMinh

- Xác định giá trị đầu vào xuất hiện trong trang kết quả: 0.25 điểm

<http://localhost/dvwa/vulnerabilities/xss_r/?name=PhamNgocHaMinh#>

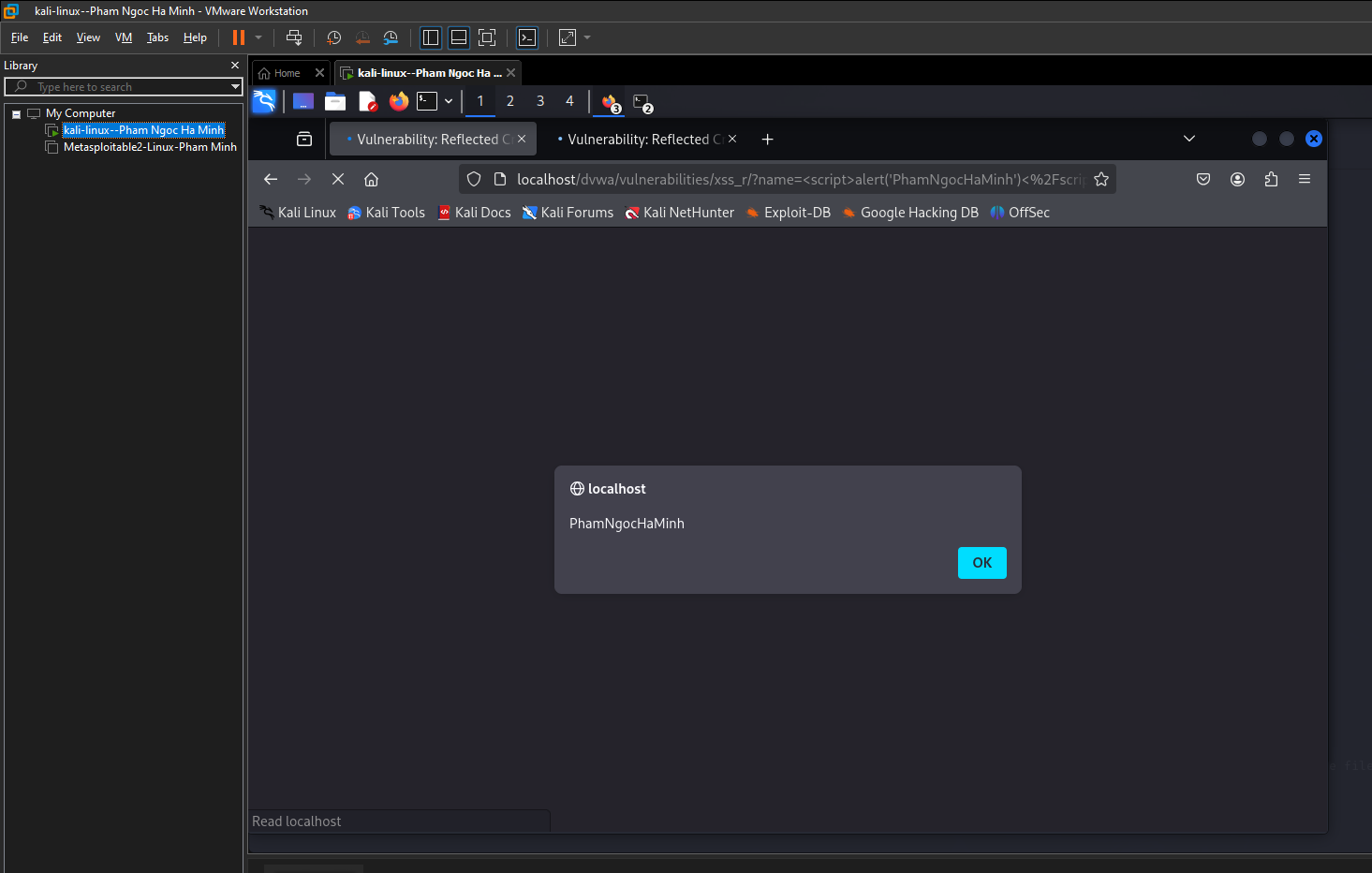


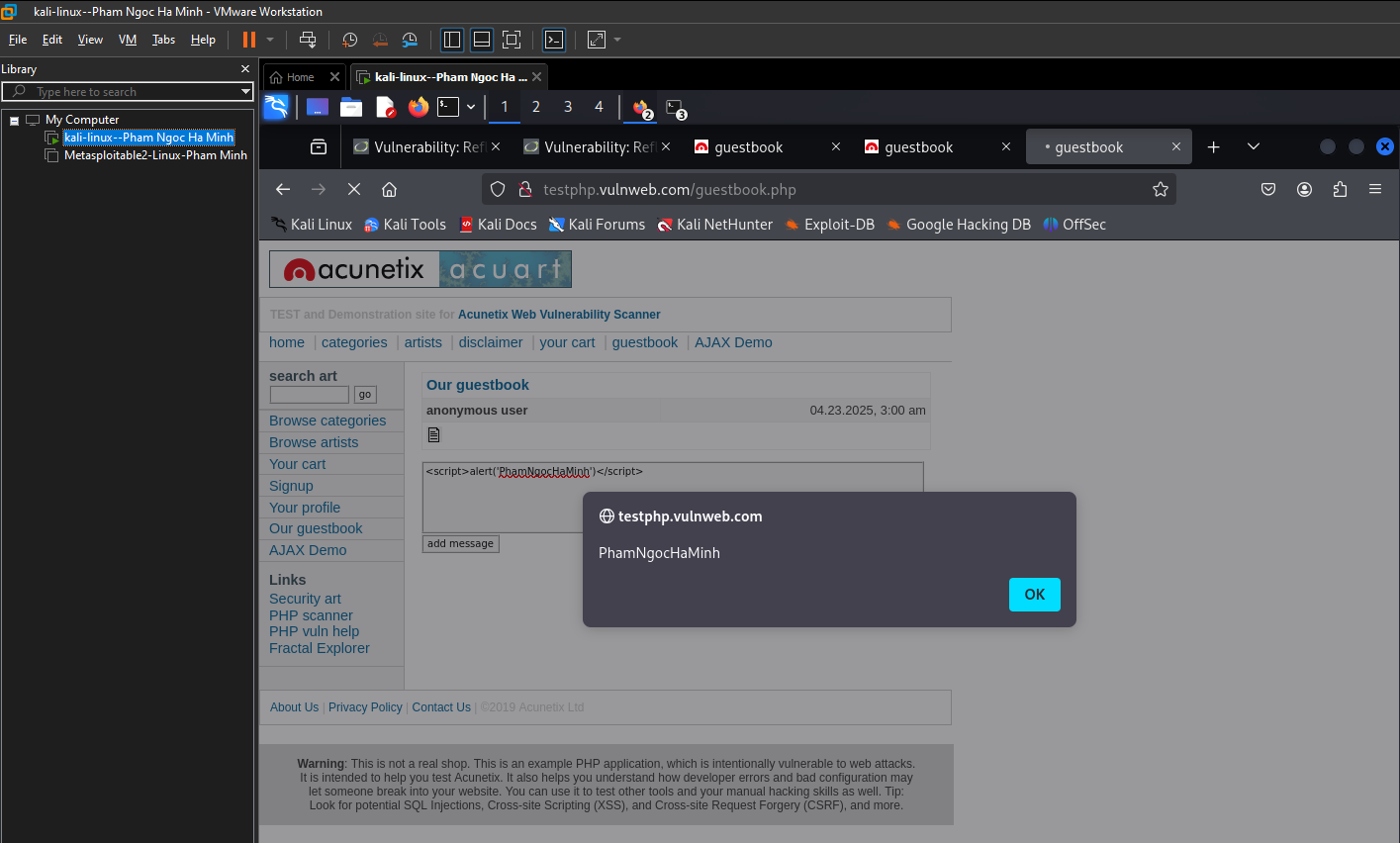


- Các bước kiểm thử và phán đoán: 0-1 điểm theo các bước kiểm thử và logic trong phán đoán

Nhập đoạn mã XSS : <script>alert('PhamNgocHaMinh')</script>

[http://localhost/dvwa/vulnerabilities/xss\_r/?name=<script>alert('PhamNgocHaMinh')</script](http://localhost/dvwa/vulnerabilities/xss_r/?name=%3cscript%3ealert('PhamNgocHaMinh')%3c/script)>

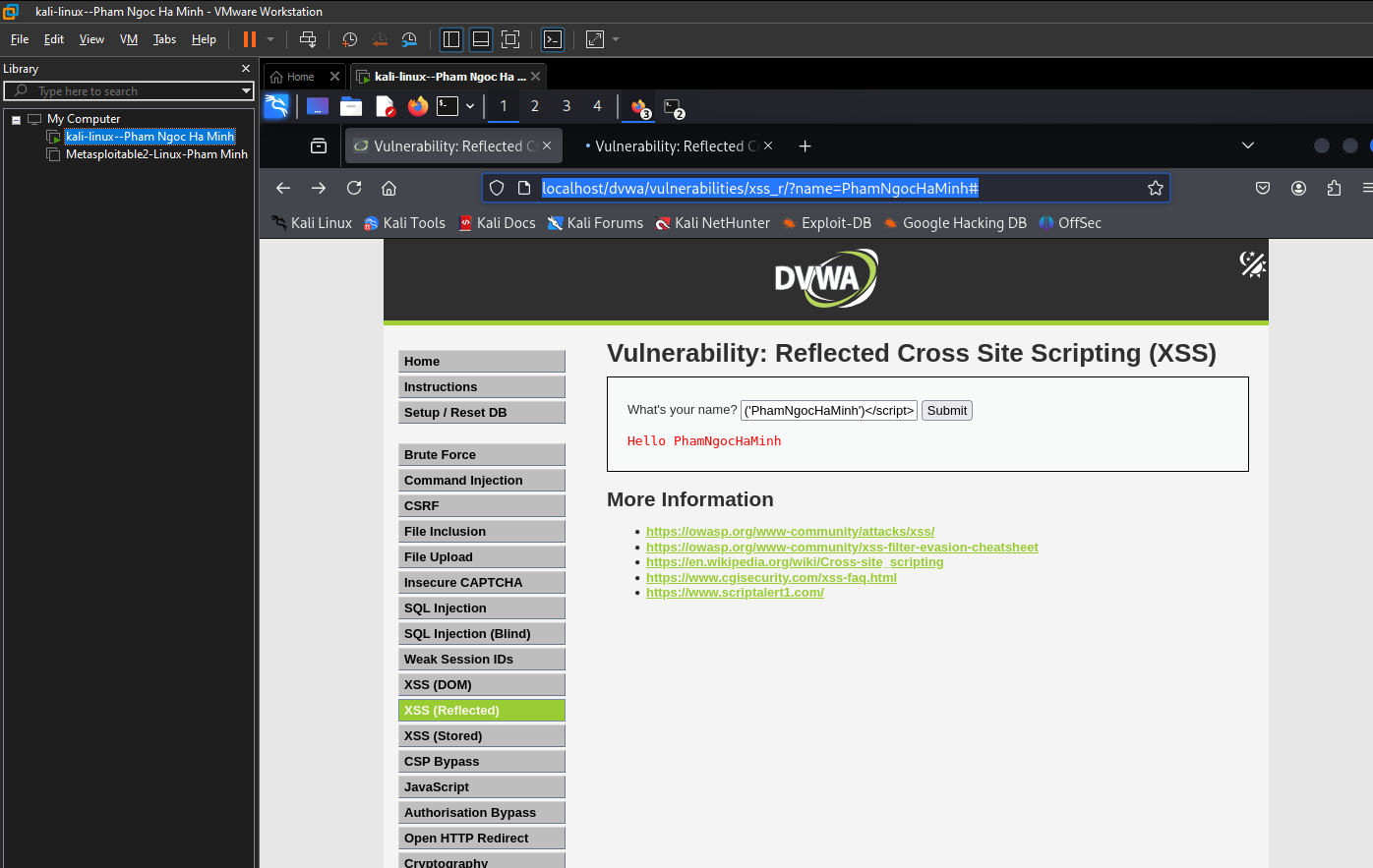


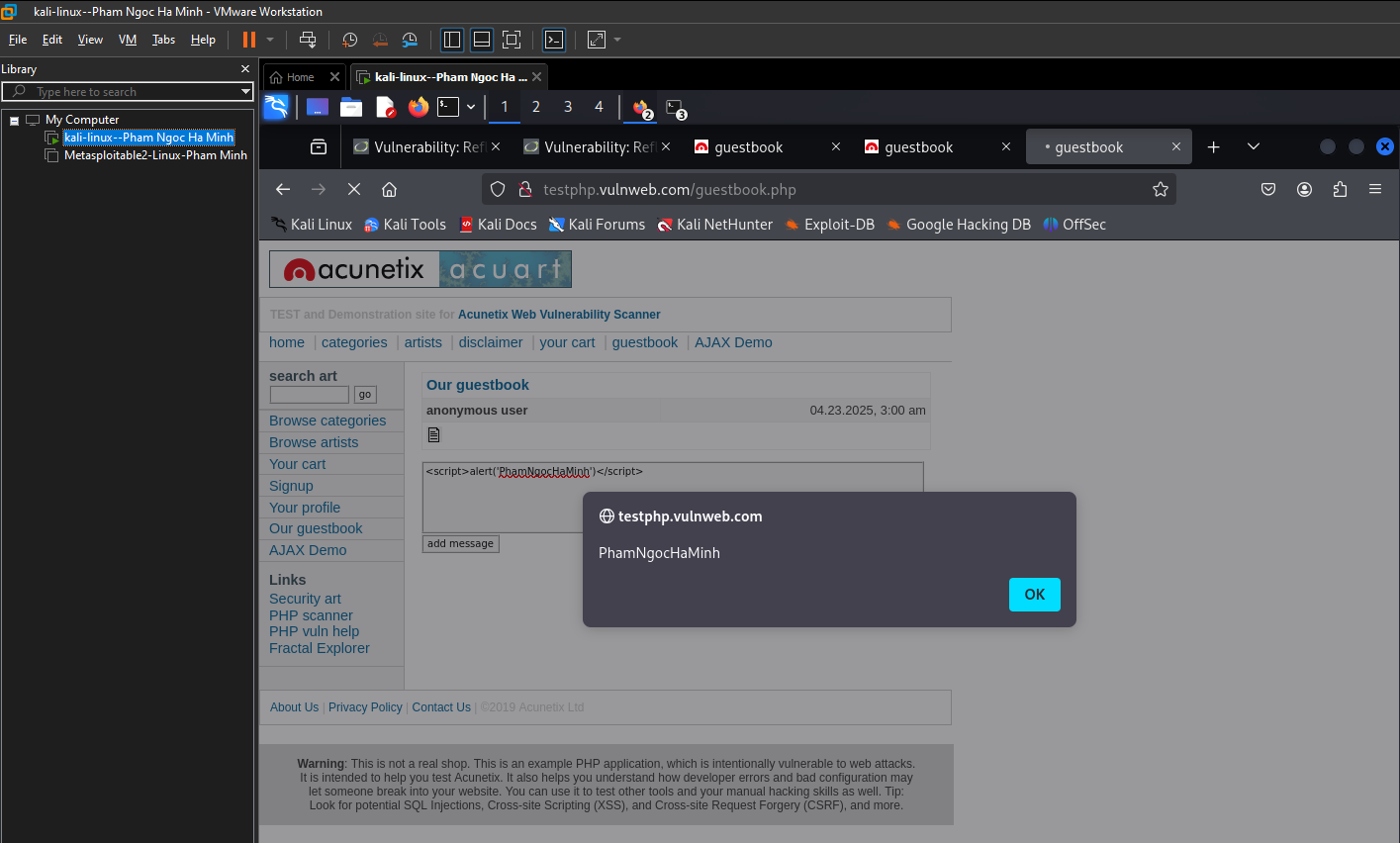


* Nếu trình duyệt hiển thị hộp thoại JavaScript chứa “PhamNgocHaMinh”, đã khai thác thành công lỗ hổng XSS.
* Vì giá trị người dùng nhập (<script>...</script>) không được mã hóa hoặc lọc → JavaScript được thực thi.

- Giá trị kiểm thử cho thấy lỗ hổng và giải thích: 0.5 điểm

<script>alert('PhamNgocHaMinh')</script>





**Giải thích:**

* Đây là một đoạn mã JavaScript hợp lệ.
* Khi chèn vào input "name" và trang web không lọc nội dung → mã được thực thi trên trình duyệt.
* Lỗ hổng nằm ở việc giá trị người dùng nhập được phản hồi trực tiếp lên trang HTML mà không qua xử lý hay mã hóa an toàn, dẫn đến XSS Reflected.

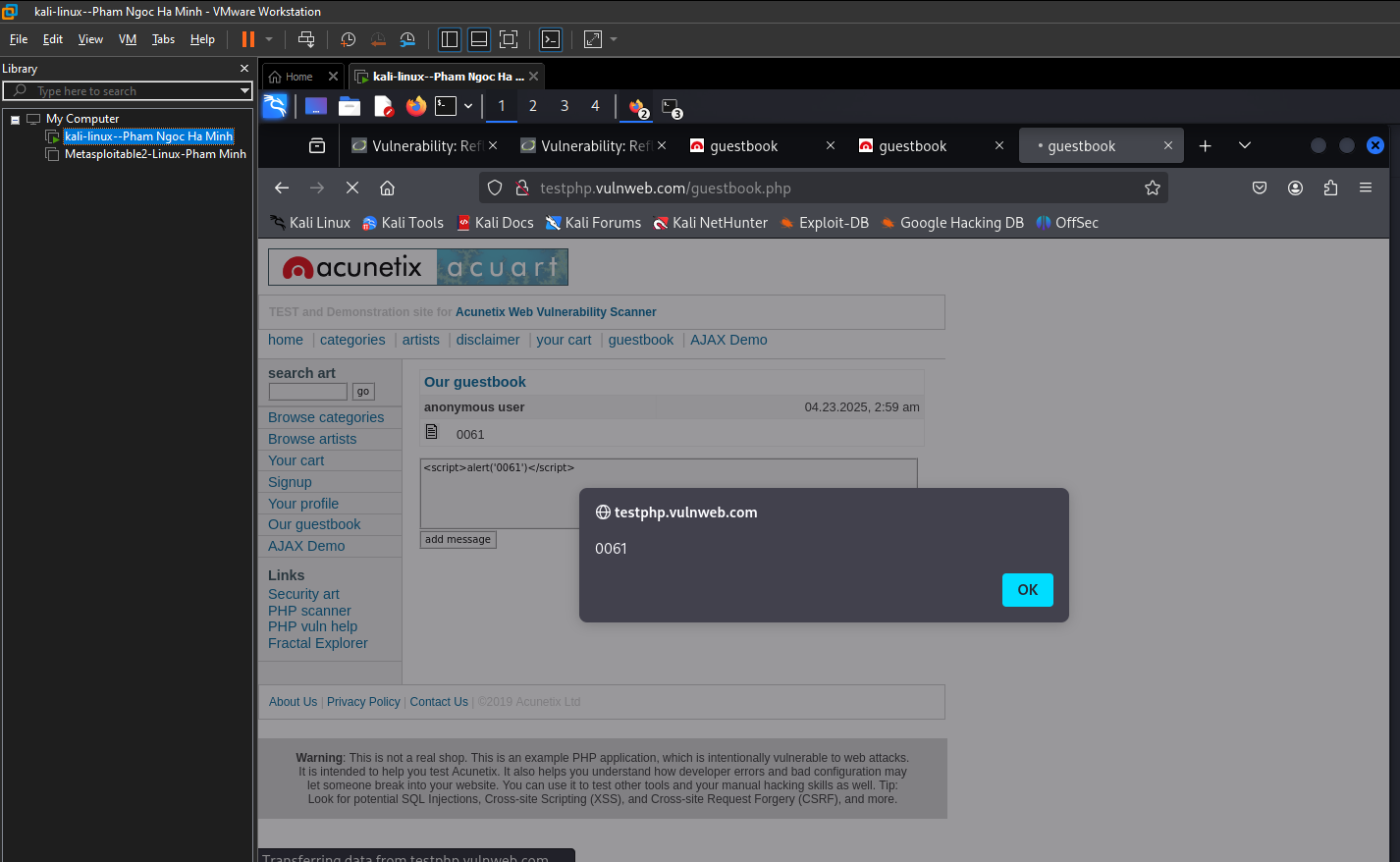
**Câu 3: Thực hiện kiểm thử lỗ hổng XSS để thực thi đoạn mã hiển thị thông báo có chứa 4 số cuối MSSV.**

Dùng 2 trang web <http://localhost/dvwa/vulnerabilities/xss_r/> và http://testphp.vulnweb.com/guestbook.php

Thực hiện kiểm thử lỗ hổng XSS để thực thi đoạn mã hiển thị thông báo có chứa 4 số cuối MSSV.

Hãy giải thích:

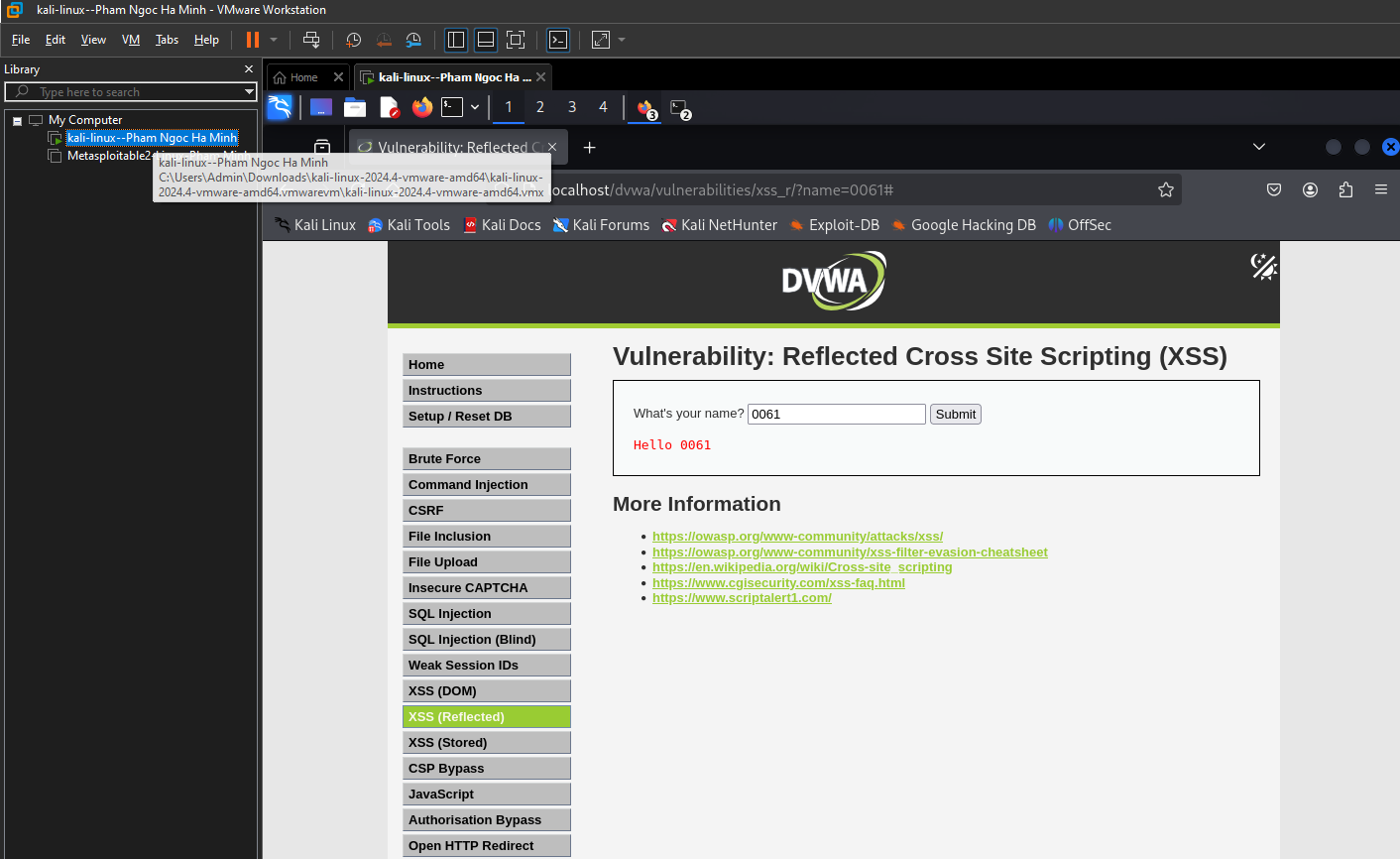
- Xác định các tham số đầu vào: 0.25 điểm



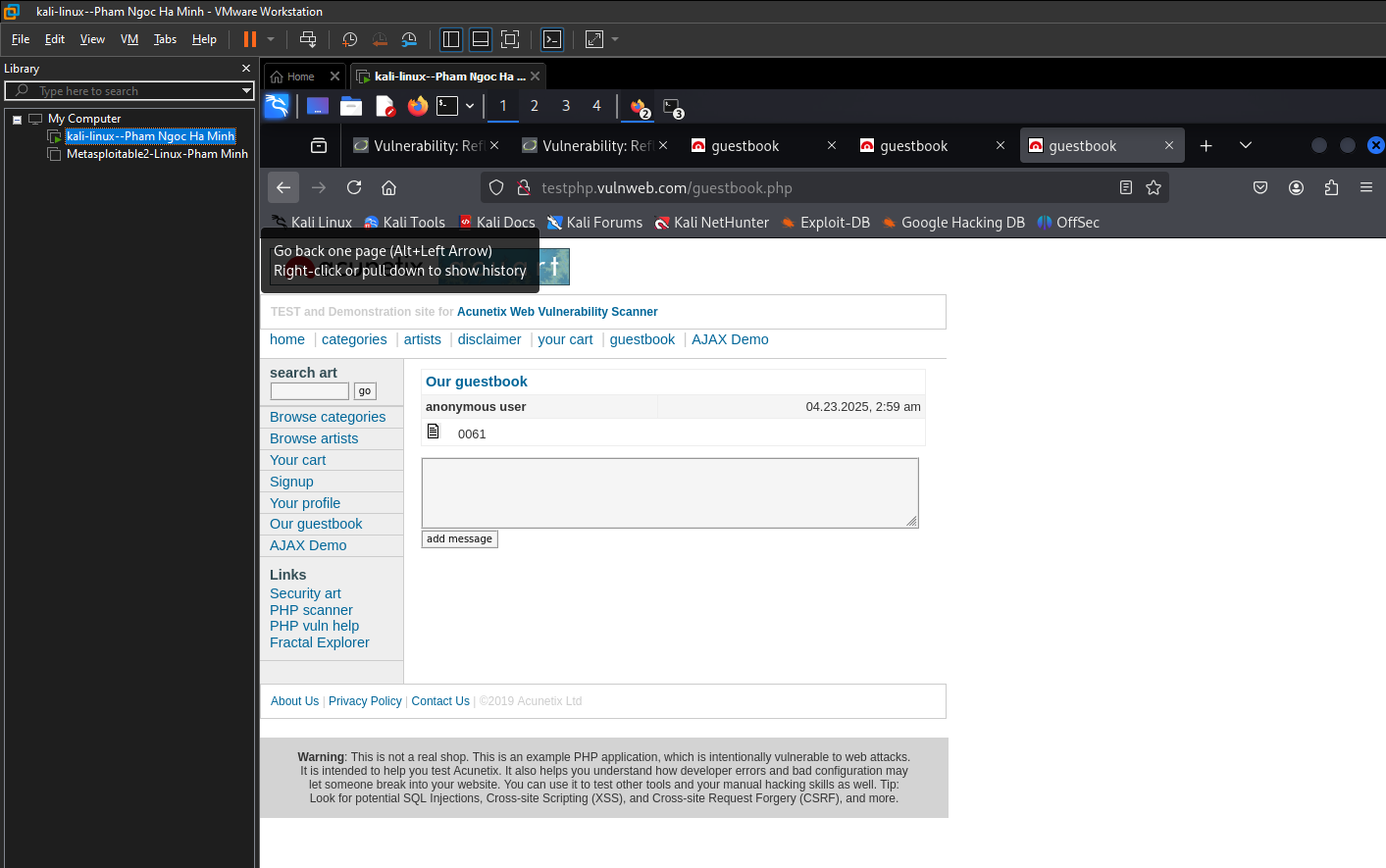
Tham số đầu vào là 0061

- Xác định giá trị đầu vào xuất hiện trong trang kết quả: 0.25 điểm

<http://localhost/dvwa/vulnerabilities/xss_r/?name=0061#>

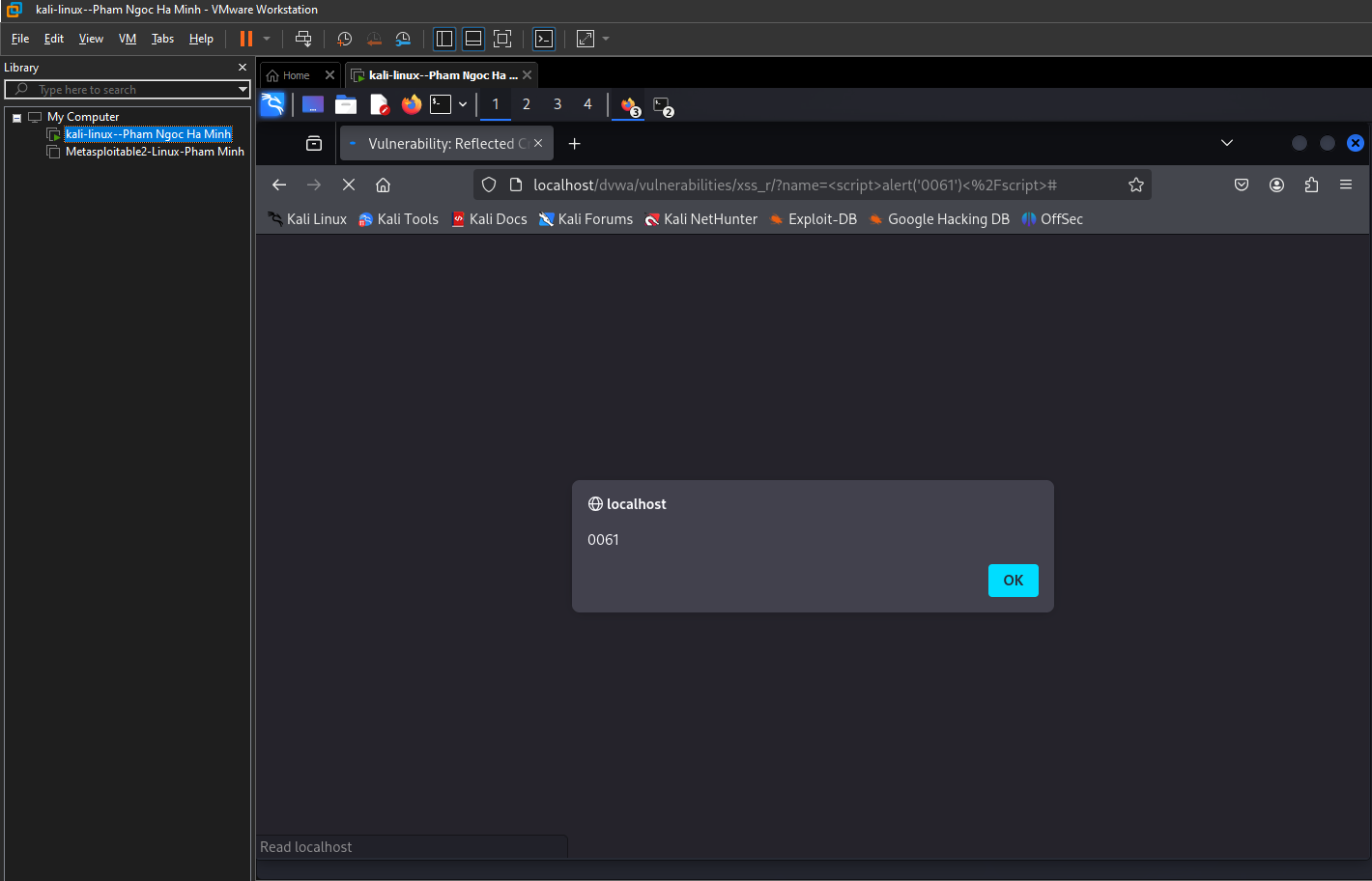


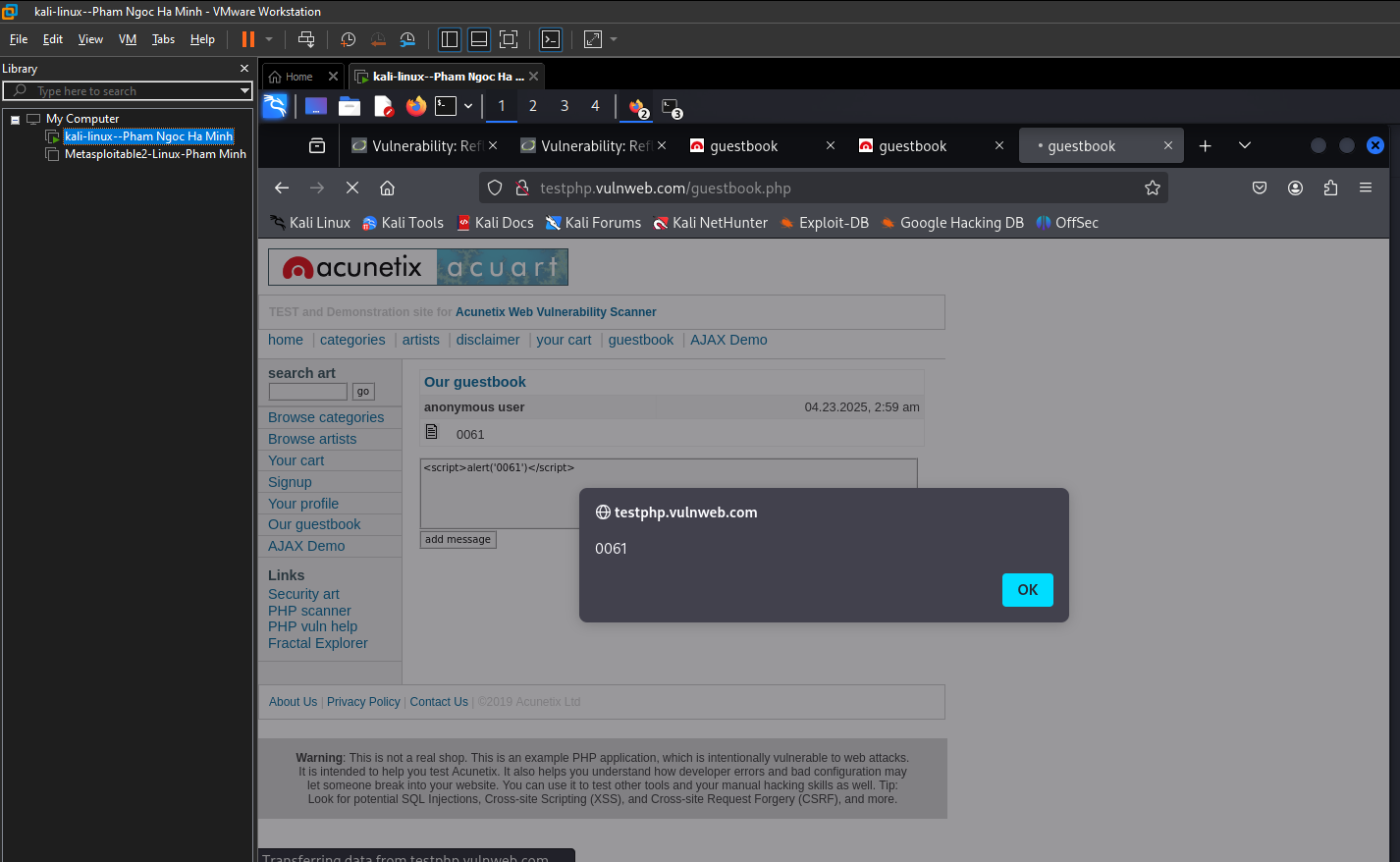
<http://testphp.vulnweb.com/guestbook.php>



- Các bước kiểm thử và phán đoán: 0-1 điểm theo các bước kiểm thử và logic trong phán đoán

Nhập đoạn mã XSS : <script>alert('0061')</script> hoặc truy cập http://localhost/dvwa/vulnerabilities/xss\_r/?name=<script>alert('0061')</script>

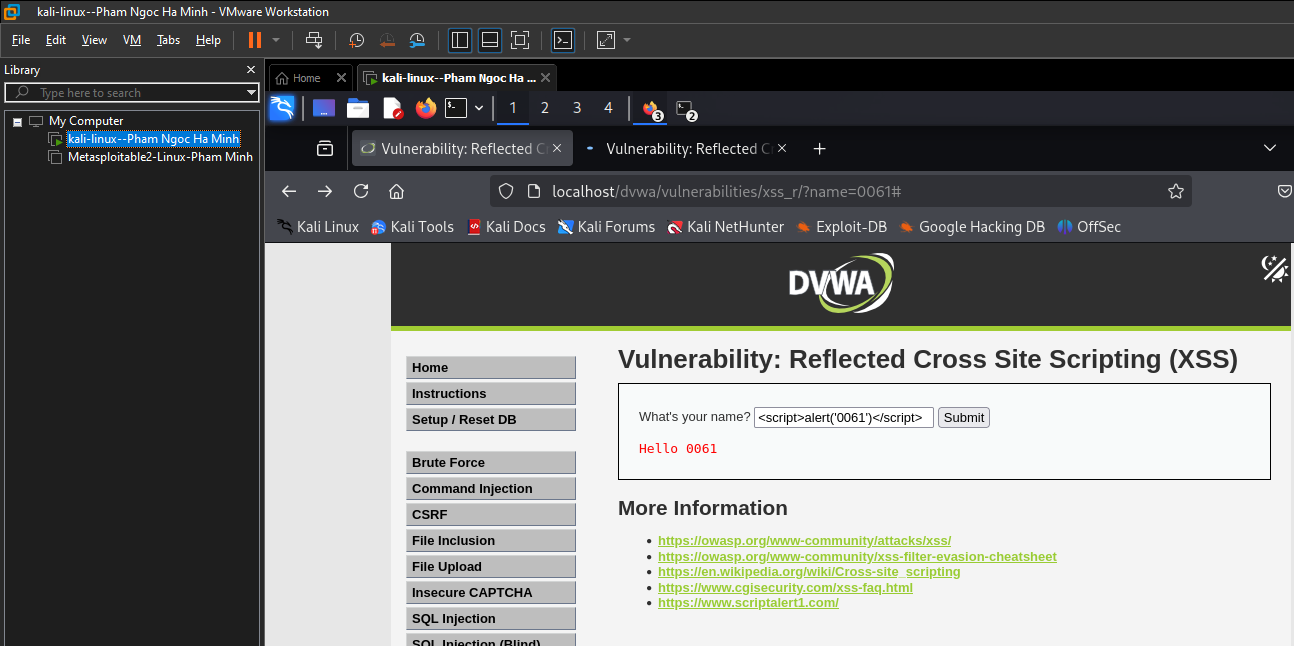


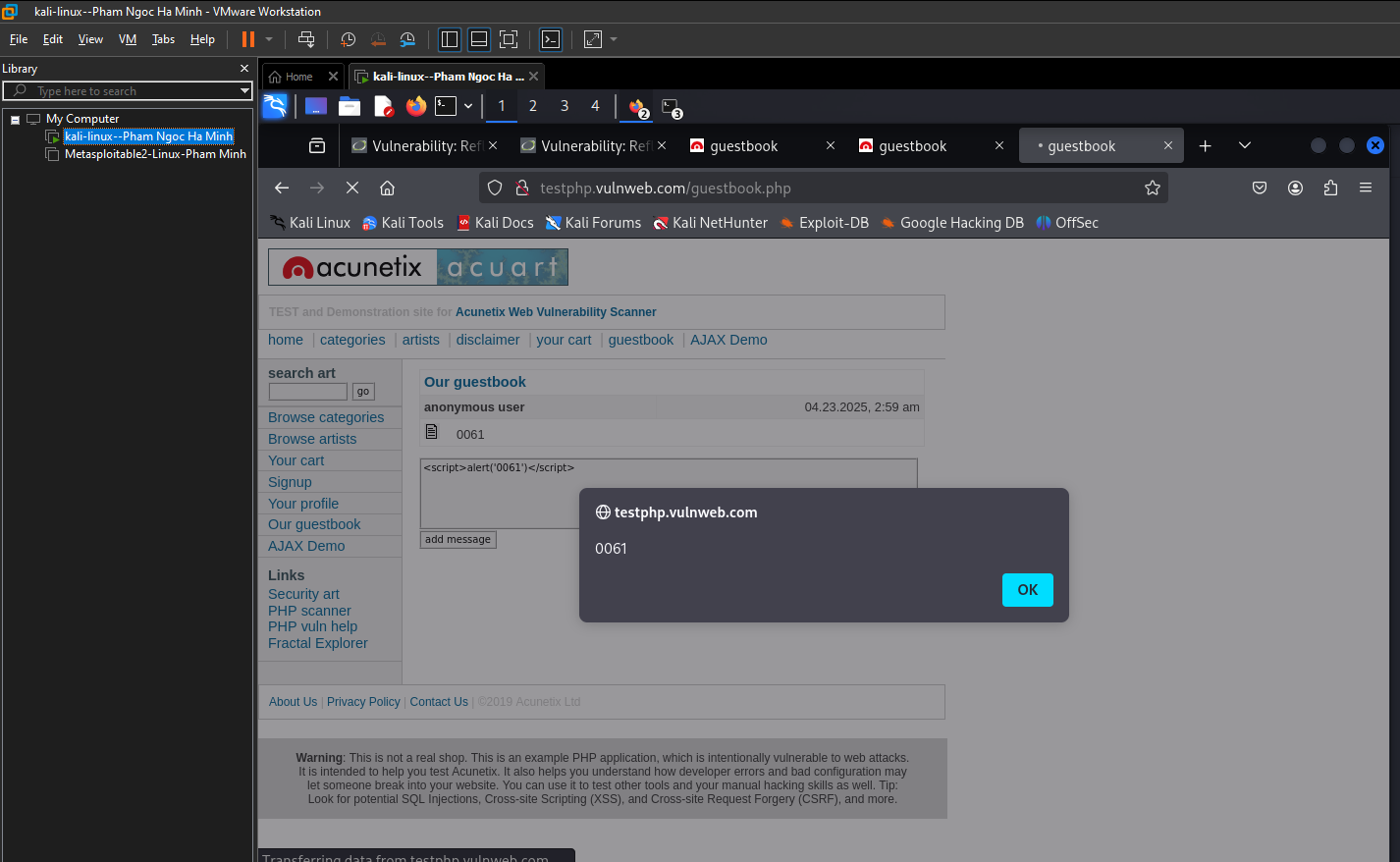


* Nếu trình duyệt hiển thị hộp thoại JavaScript chứa “0061”, đã khai thác thành công lỗ hổng XSS.
* Vì giá trị người dùng nhập (<script>...</script>) không được mã hóa hoặc lọc → JavaScript được thực thi.

- Giá trị kiểm thử cho thấy lỗ hổng và giải thích: 0.5 điểm

<script>alert('0061')</script>





**Giải thích:**

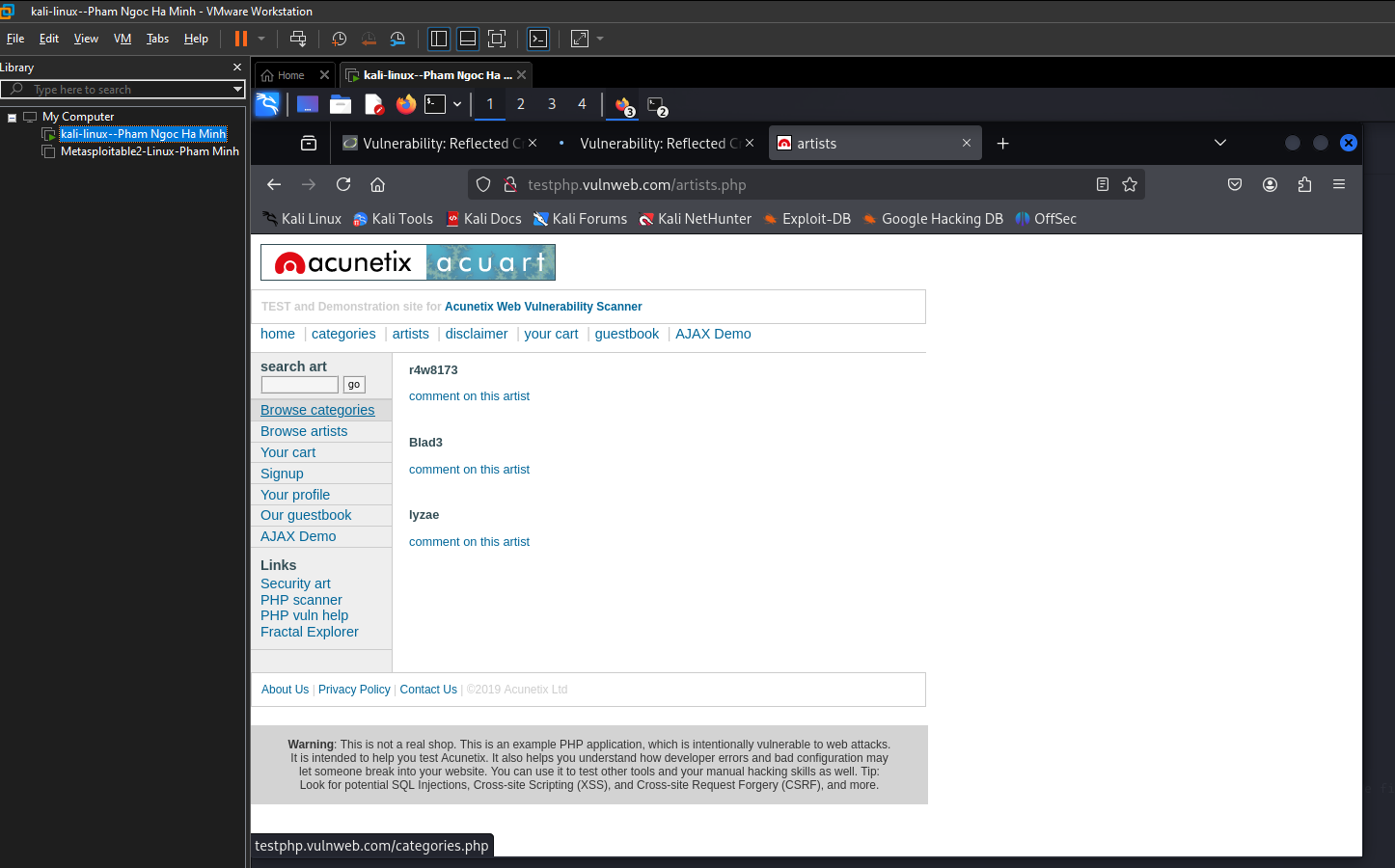
* Đây là một đoạn mã JavaScript hợp lệ.
* Khi chèn vào input và trang web không lọc nội dung → mã được thực thi trên trình duyệt.
* Lỗ hổng nằm ở việc giá trị người dùng nhập được phản hồi trực tiếp lên trang HTML mà không qua xử lý hay mã hóa an toàn, dẫn đến XSS Reflected.

**Phần 2: Kiểm thử CSRF (2 điểm)**

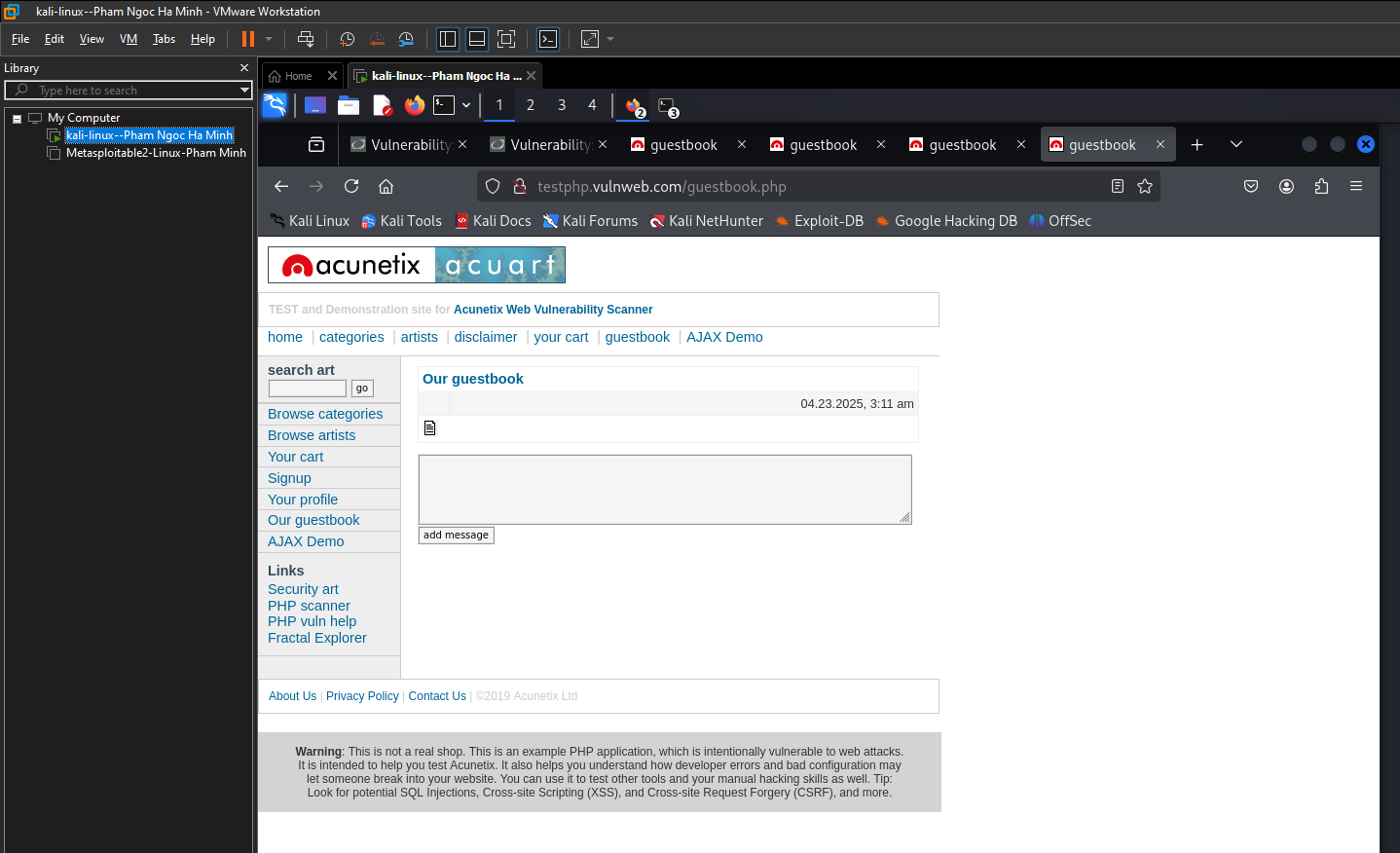
Trình bày các bước thực hiện và kết quả kiểm thử lỗ hổng CSRF trên website chọn 1 trong các trang sau để thử nghiệm

Truy cập [**http://testphp.vulnweb.com/artists.php**](http://testphp.vulnweb.com/artists.php)

Chọn phần guestbook để demo http://testphp.vulnweb.com/guestbook.php

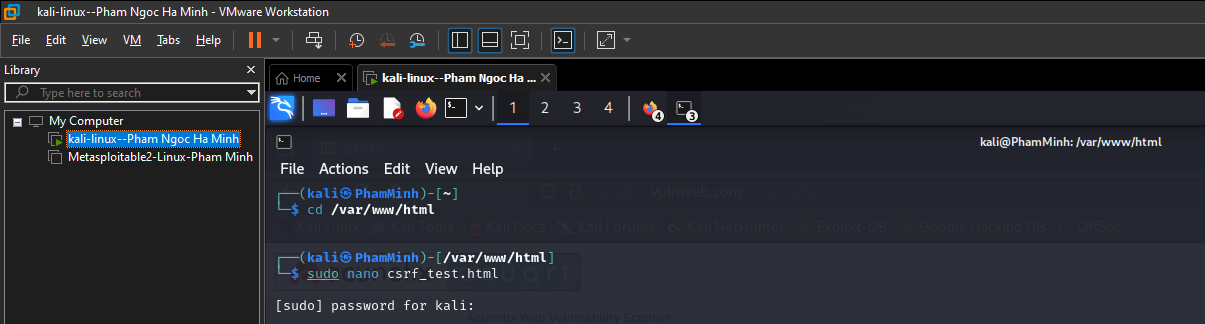


Chọn guestbook.php thực hiện POST request



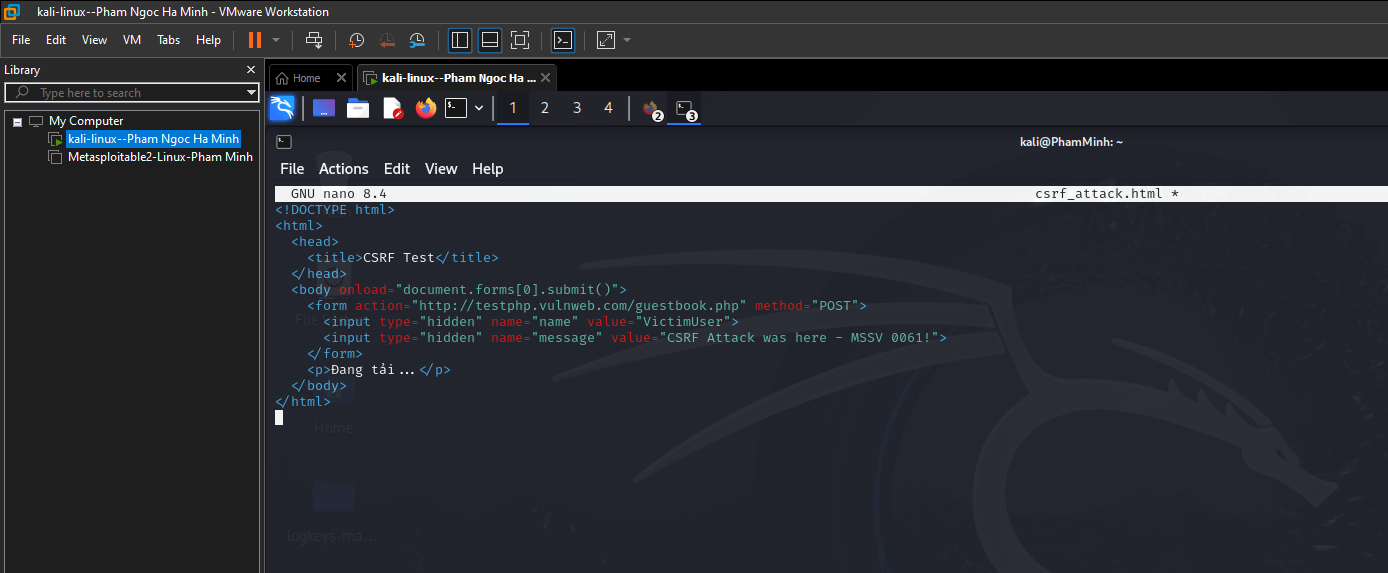
cd /var/www/html

tạo file tấn công nano csrf\_attack.html



Nhập nội dung tấn công

Tạo tệp HTML giả mạo



Lưu và mở bằng trình duyệt firefox csrf\_attack.html

Thông báo "VictimUser" trên trang **guestbook.php** sau khi thực hiện tấn công CSRF, điều đó có thể cho thấy rằng tấn công CSRF đã thành công và một bình luận đã được thực hiện mà không có sự đồng ý của người dùng.

