**BÁO CÁO BÀI THỰC HÀNH SỐ 9**

**KIỂM THỬ LỖ HỔNG XSS và CSRF**

***Họ và tên: Đặng Hiển Danh***

***Mssv: 1050080218***

**Phần 1: Kiểm thử XSS**

Thực hiện kiểm thử lỗ hổng XSS trên website: chọn 1 trong các trang sau để thử nghiệm

1. [http://demo.testfire.net](http://demo.testfire.net/)`
2. [http://php.testsparker.com](http://php.testsparker.com/)
3. [http://testasp.vulnweb.com](http://testasp.vulnweb.com/)
4. [http://testasp.vulnweb.com](http://testasp.vulnweb.com/)
5. [http://testphp.vulnweb.com](http://testphp.vulnweb.com/)
6. [http://www.webscantest.com](http://www.webscantest.com/)
7. [http://testhtml5.vulnweb.com](http://testhtml5.vulnweb.com/)
8. [http://aspnet.testsparker.com](http://aspnet.testsparker.com/)
9. [http://zero.webappsecurity.com](http://zero.webappsecurity.com/)

**Câu 1**

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:

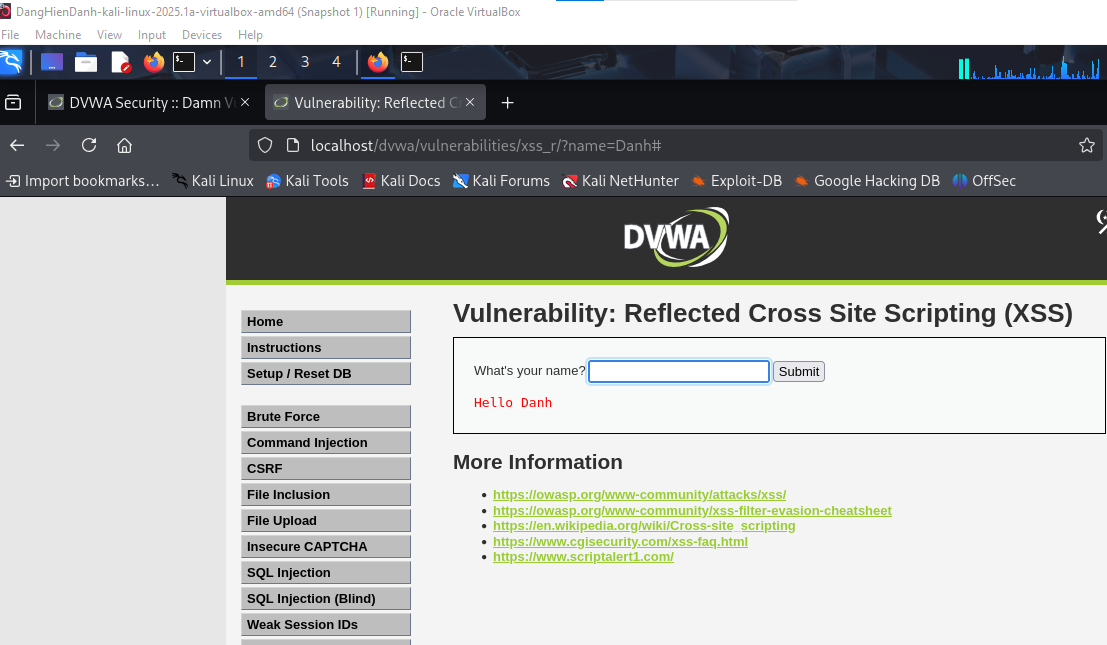
**-** Xác định các tham số đầu vào:

Tham số được truyền name

Thể hiện qua URL sau khi bấm "Submit": http://localhost/dvwa/vulnerabilities/xss\_r/?name=Danh

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

Sau khi nhập tên Danh và bấm Submit, trang trả về dòng:



Dòng kết quả cho thấy **giá trị của tham số** name **được hiển thị trực tiếp vào HTML** : Có thể bị chèn mã độc

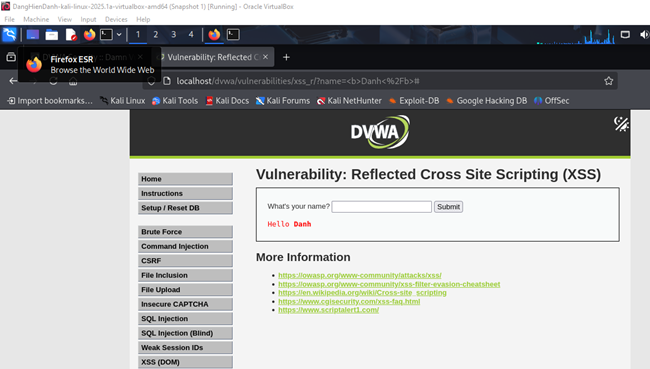
**-** Các bước kiểm thử và phán đoán: theo các bước kiểm thử và logic trong phán đoán

1. Nhập bình thường: Danh

Trang hiển thị Hello Danh

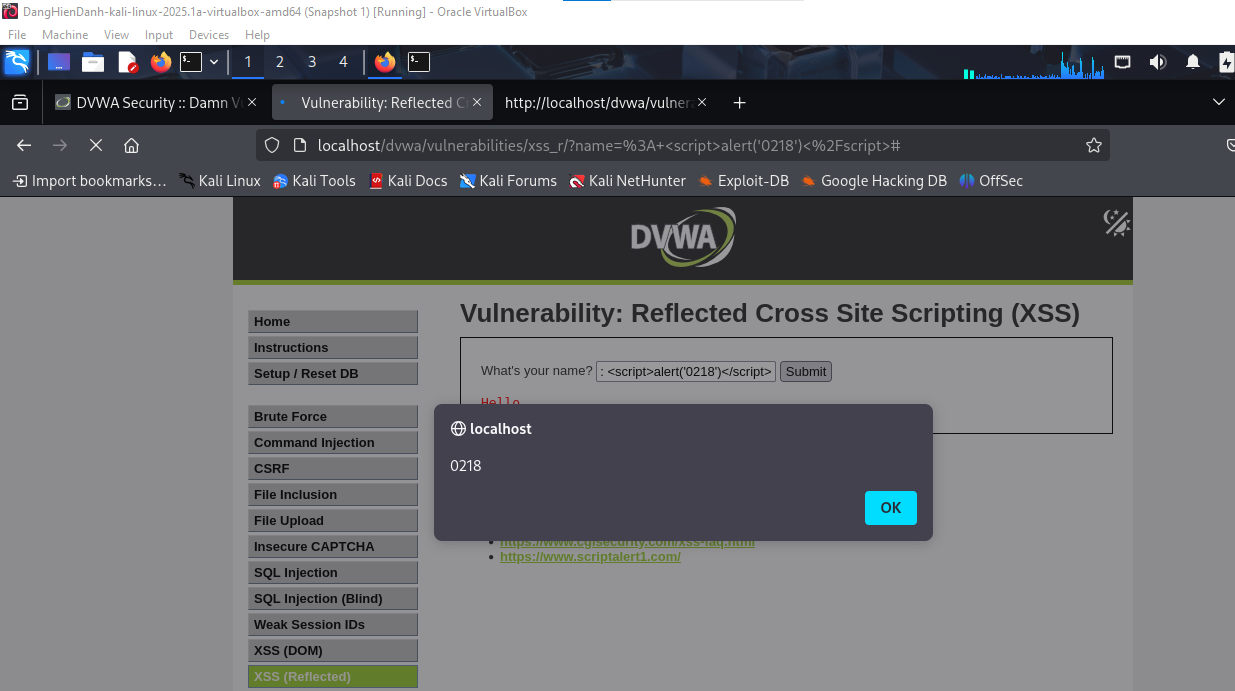
2. Thử HTML: <b>Danh</b>

Dòng chữ **Danh** được in đậm : HTML không bị lọc



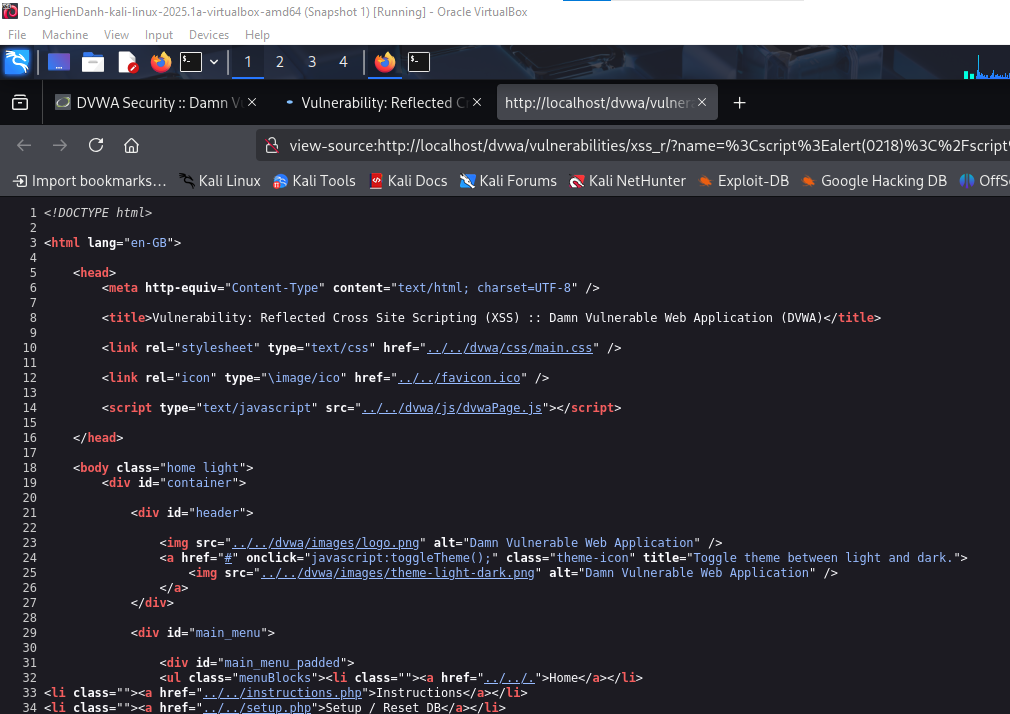
3. Thử JS: <script>alert(0218)</script>

Hộp thoại alert hiện ra : **đoạn mã JavaScript đã thực thi**



**4.** Kiểm tra mã nguồn (Ctrl+U)

Thấy đoạn mã đầu vào được **nhúng trực tiếp vào HTML**

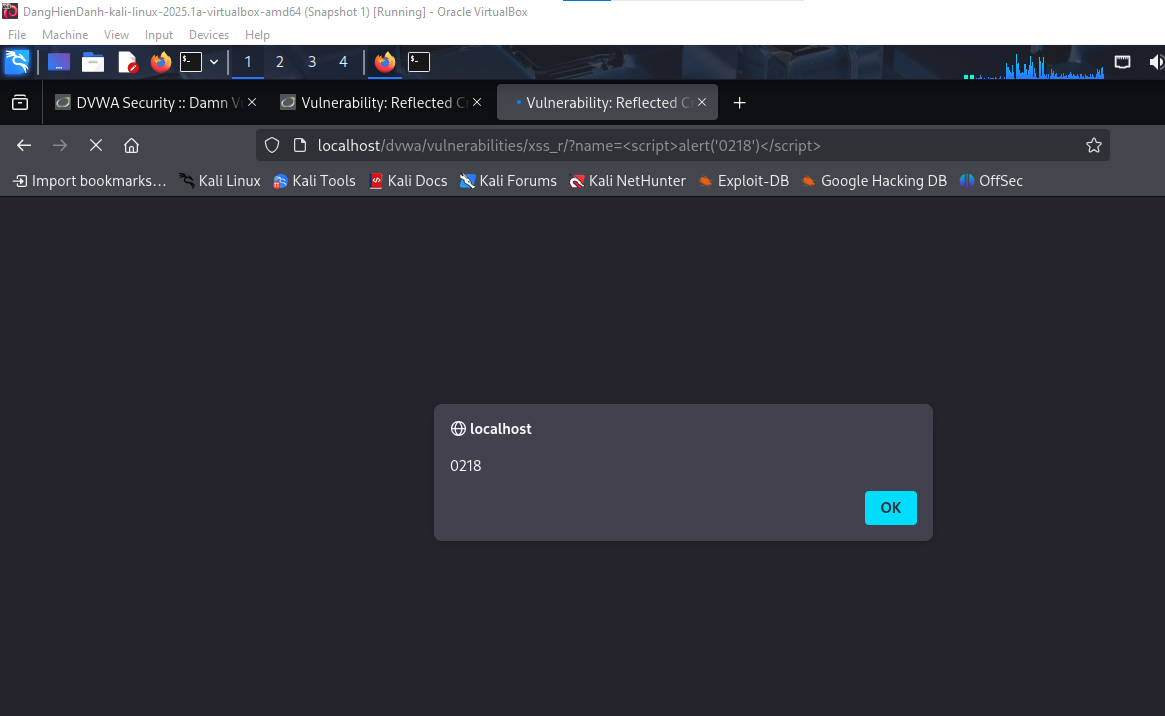


- Giá trị kiểm thử cho thấy lỗ hổng và giải thích:

Payload kiểm thử: <script>alert('0218')</script>

Khi người dùng truy cập URL sau: [http://localhost/dvwa/vulnerabilities/xss\_r/?name=<script>alert('0218')</script](http://localhost/dvwa/vulnerabilities/xss_r/?name=%3cscript%3ealert('0218')%3c/script)>

Trình duyệt hiển thị hộp thoại 0218 : **XSS Reflected thành công**



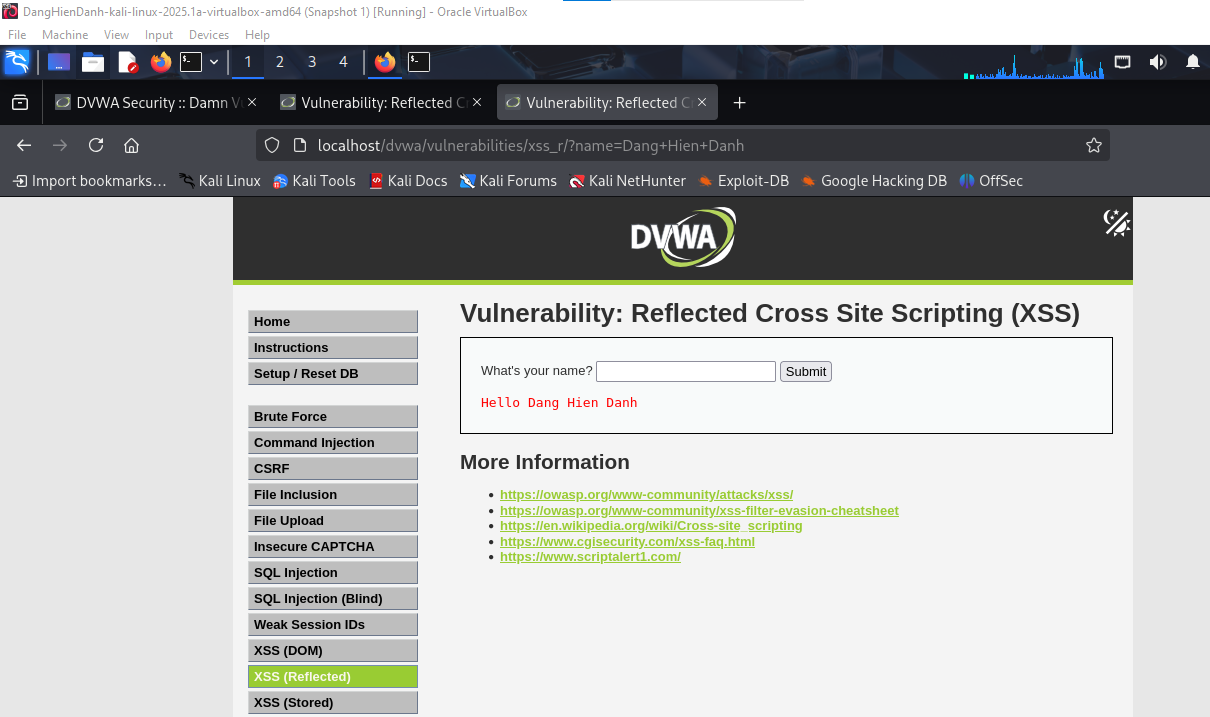
**Câu 2**

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:

Tham số là: name

<http://localhost/dvwa/vulnerabilities/xss_r/?name=Dang+Hien+Danh>



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

Nhập: Dang Hien Danh

Kết quả trả về:

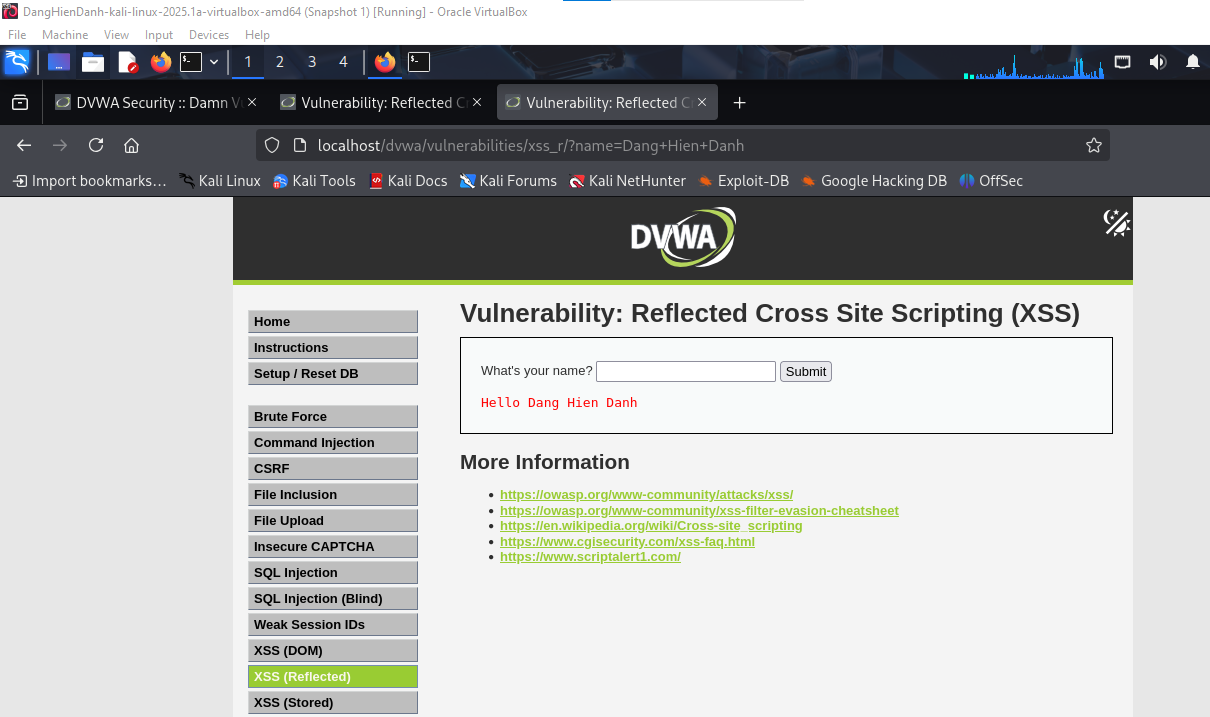
Hello Dang Hien Danh

Nghĩa là **giá trị nhập vào được hiển thị trực tiếp trên giao diện HTML** : Có nguy cơ bị lợi dụng để chèn mã JavaScript

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

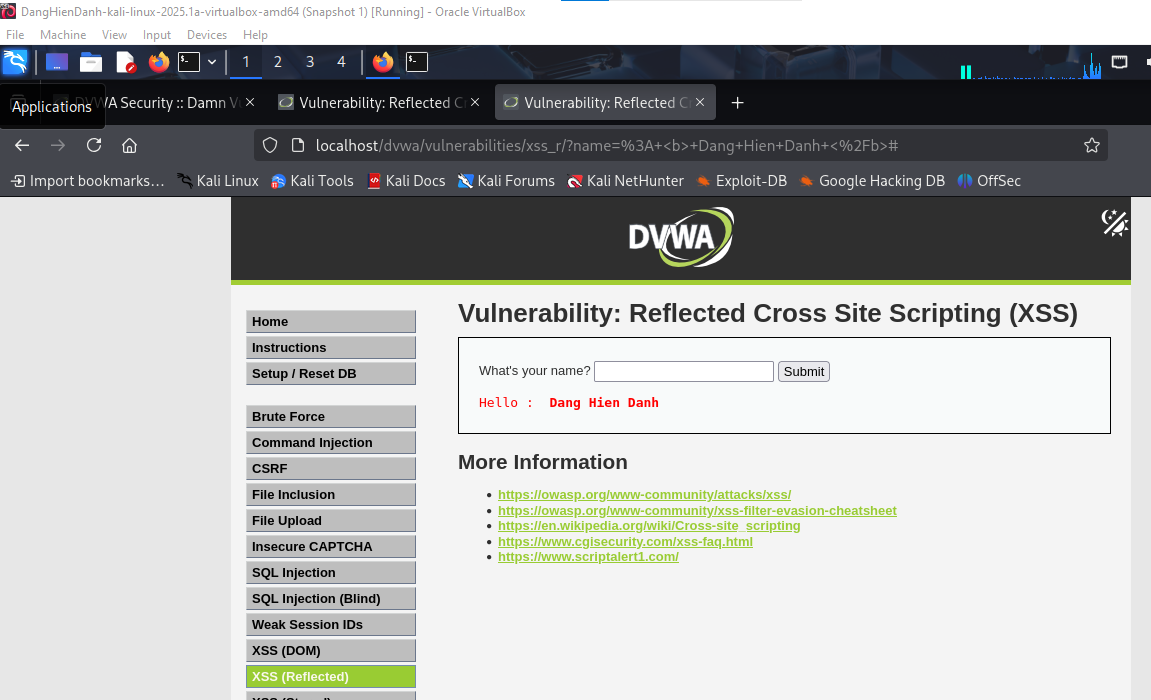
1. Nhập giá trị bình thường: Dang Hien Danh

Hiển thị đúng nội dung



2. Nhập HTML: <b> Dang Hien Danh </b>

Tên hiển thị in đậm : Trang không lọc HTML



3. Nhập JavaScript: <script>alert(' Dang Hien Danh’)</script>

Hộp thoại xuất hiện : JavaScript thực thi

4. Xem mã nguồn trang (Ctrl+U)

|  |
| --- |
|  |

|  |
| --- |
| Giá trị name được chèn trực tiếp vào HTML |
|  |

Dẫn đến khả năng **Reflected XSS** vì không có xử lý đầu vào

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

Giá trị kiểm thử: <script>alert(' Dang Hien Danh’)</script>

URL mẫu: http://localhost/dvwa/vulnerabilities/xss\_r/?name=<script>alert(' Dang Hien Danh)</script>

Trình duyệt thực thi đoạn JavaScript trong biến name - hiển thị hộp thoại

Điều này chứng minh website **không lọc/escape đầu vào**

Hacker có thể lợi dụng điểm này để chèn mã độc (đánh cắp cookie, điều hướng, tạo popup giả…)

**Câu 3**

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**

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

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

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

**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

1. [http://demo.testfire.net](http://demo.testfire.net/)
2. [http://php.testsparker.com](http://php.testsparker.com/)
3. [http://testasp.vulnweb.com](http://testasp.vulnweb.com/)
4. [http://testasp.vulnweb.com](http://testasp.vulnweb.com/)
5. [http://testphp.vulnweb.com](http://testphp.vulnweb.com/)
6. [http://www.webscantest.com](http://www.webscantest.com/)
7. [http://testhtml5.vulnweb.com](http://testhtml5.vulnweb.com/)
8. [http://aspnet.testsparker.com](http://aspnet.testsparker.com/)
9. [http://zero.webappsecurity.com](http://zero.webappsecurity.com/)