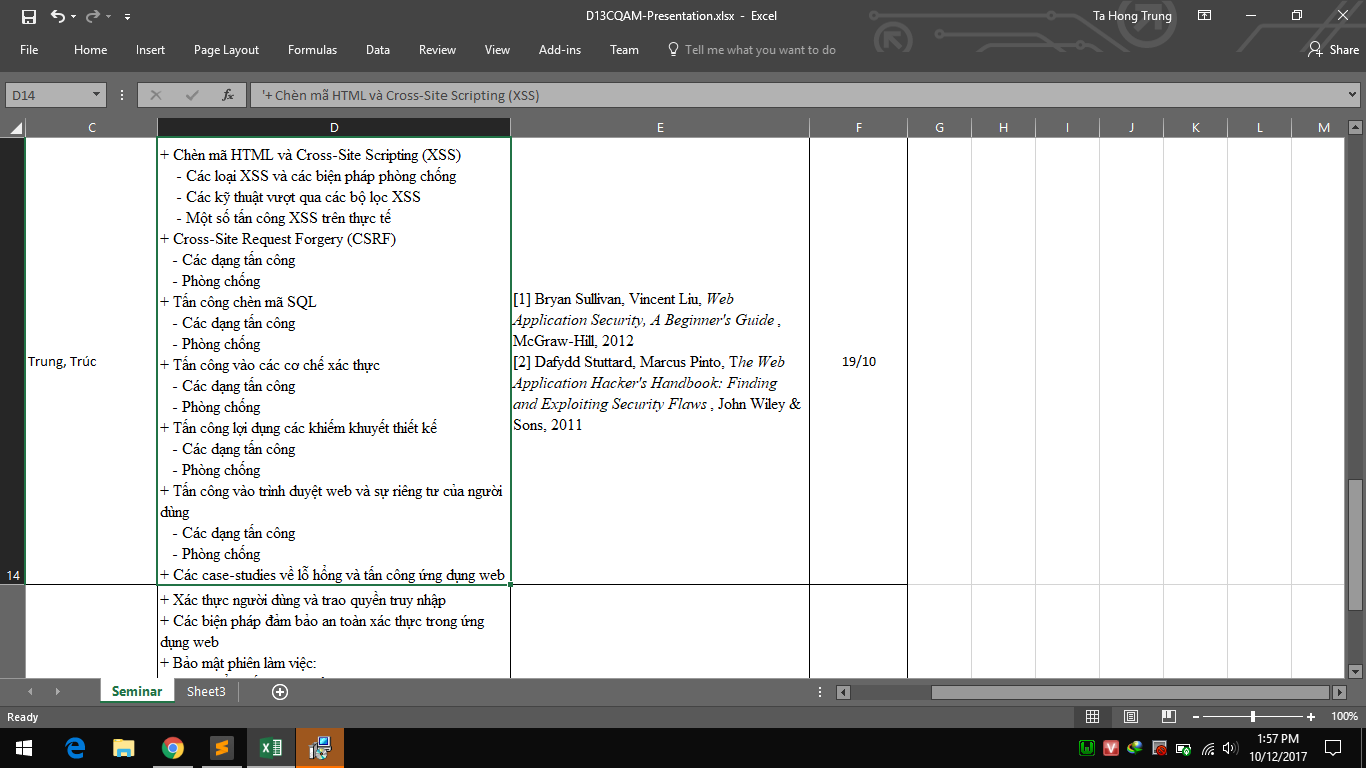
****

1. **Khái niệm**

* **XSS (Cross-Site Scripting)**:  
  Là một trong những kĩ thuật tấn công phổ biến nhất hiện nay, đồng thời nó cũng là một trong những vấn đề bảo mật quan trọng đối với các nhà phát triển web và cả những người sử dụng web. XSS là một kỹ thuật tấn công bằng cách chèn vào các website động những thẻ HTML hay những đoạn scrip nguy hiểm có thể gây hại cho những người sử dụng khác. Trong đó, những đoạn mã nguy hiểm được chèn vào hầu hết được viết bằng các Client-Site Scrip như JavaScrip, Jscrip.. và cũng có thể là các thẻ HTML.
* **CSRF (Cross-site Request Forgery)**:  
  Là kỹ thuật tấn công bằng cách sử dụng quyền chứng thực của người dùng đối với một website. Nó là kỹ thuật tấn công vào người dùng, dựa vào đó hacker có thể thực thi những thao tác phải yêu cầu sự chứng thực,  đây là kỹ thuật tấn công dựa vào mượn quyền trái phép.CSRF còn được gọi là "session riding", "XSRF"
* **SQL injection**:  
  SQL Injection là một kỹ thuật lợi dụng những lỗ hổng về câu truy vấn lấy dữ liệu của những website không an toán, đây là một kỹ thuật tấn công rất phổ biến và sự thành công của nó cũng tương đối cao.

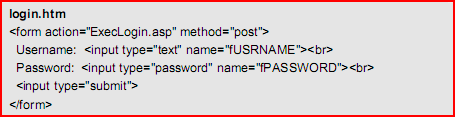
1. **SQL INJECTION**
   1. **Các dạng tấn công**

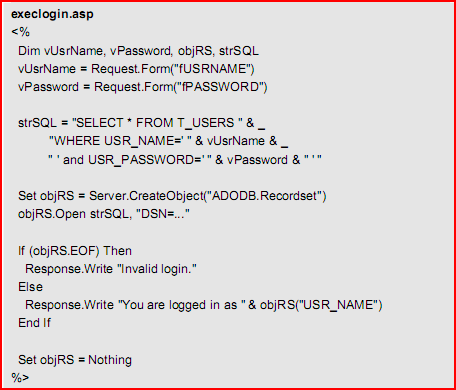
Có sáu dạng thông thường bao gồm:

* + ***Vượt qua kiểm tra lúc đăng nhập (Aauthorization bypass)***

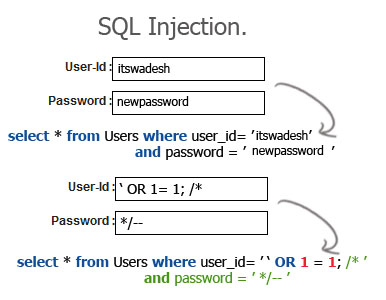
Với dạng tấn công này, tin tặc có thể dễ dàng vượt qua các trang đăng nhập nhờ vào lỗi khi dùng các câu lệnh SQL thao tác trên cơ sở dữ liệu của ứng dụng web.

Sau khi người dùng nhập thông tin vào, hệ thống sẽ kiểm tra tên đăng nhập và mật khẩu có hợp lệ hay không để quyết định cho phép hay từ chối thực hiện tiếp.





Thoạt nhìn, đoạn mã trong trang execlogin.asp dường như không chứa bất cứ một lỗ hổng về an toàn nào. Người dùng không thể đăng nhập mà không có tên đăng nhập và mật khẩu hợp lệ. Tuy nhiên, đoạn mã này thực sự không an toàn và là tiền đề cho một lỗi SQL injection. Đặc biệt, chỗ sơ hở nằm ở chỗ dữ liệu nhập vào từ người dùng được dùng để xây dựng trực tiếp câu lệnh SQL. Chính điều này cho phép những kẻ tấn công có thể điều khiển câu truy vấn sẽ được thực hiện.



* + ***Sử dụng câu lệnh INSERT***

Thông thường các ứng dụng web cho phép người dùng đăng kí một tài khoản để tham gia. Chức năng không thể thiếu là sau khi đăng kí thành công, người dùng có thể xem và hiệu chỉnh thông tin của mình. SQL injection có thể được dùng khi hệ thống không kiểm tra tính hợp lệ của thông tin nhập vào.

Từ khoá INSERT dùng để đưa thông tin vào cơ sở dữ liệu. Thông thường câu lệnh INSERT được dùng trong các trường hợp như: thông tin đăng kí người sử dụng, guestbook…v..v…

Kĩ thuật “;”, “--“ được dùng như đã từng dùng với câu lệnh SELECT, phải đảm bảo đúng số lượng và kiểu giá trị được nhập vào nhằm tránh lỗi về cú pháp (nếu không xác định được kiểu dữ liệu có thể nhập tất cả là số).

VD:

SQLString= “INSERT INTO User VALUES („” & strUsername & “‟, „” & strName& “‟, „” & strPassWord & “‟,‟”& strLimitSize & “‟)”

Một câu lệnh INSERT có thể có cú pháp dạng:

INSERT INTO *TableName*

VALUES('*Value One*', '*Value Two*', '*Value Three*').

Nếu hacker nhập vào tại: *Value One* chuỗi:

**‘ + SELECT TOP 1 FieldName FROM TableName + ’**

Lúc này câu truy vấn sẽ là:

INSERT INTO TableName

VALUES(' ' + (**SELECT TOP 1 FieldName FROM TableName**) + ' ', 'abc', 'def')

Lúc thực hiện lệnh xem thông tin, xem như bạn đã yêu cầu thực hiện thêm một lệnh nữa đó là:

**SELECT TOP 1 FieldName FROM TableName**

Vậy tất cả những dữ liệu nằm trên website của bạn, hacker đều có thể lấy cắp được.

<%

strSQL = "INSERT INTO TableName VALUES(' " & strValueOne & " ', ' " \_

& strValueTwo & " ', ' " & strValueThree & " ') "

Set objRS = Server.CreateObject("ADODB.Recordset")

objRS.Open strSQL, "DSN=..."

…

Set objRS = Nothing

%>

Hình 2.5 Một đoạn mã ví dụ tấn công bằng câu lệnh Insert.

* + ***Sử dụng câu lệnh HAVING***

HAVING sử dụng cùng chung với mệnh đề GROUP BY là phương pháp hữu hiệu để nhận thông tin bảng, trường, …

* + ***Sử dụng câu lệnh kết hợp UNION***

Lệnh SELECT được dùng để lấy thông tin từ cơ sở dữ liệu. Thông thường vị trí có thể được chèn thêm vào một mệnh đề SELECT là sau WHERE. Để có thể trả về nhiều dòng thông tin trong bảng, thay đổi điều kiện trong mệnh đề WHERE bằng cách chèn thêm UNION SELECT.

Ví dụ:

Query =“SELECT username FROM User WHERE username like ‘%” $username“’UNION SELECT password from User”

Câu lệnh trên trả về một tập kết quả là sự kết hợp giữa Username và Password trong bảng User.

* + ***Sử dụng các stored-procedures***

Việc tấn công bằng stored-procedures sẽ gây tác hại rất lớn nếu ứng dụng được thực thi với quyền quản trị hệ thống 'sa'.

Ví dụ, nếu ta thay đoạn mã tiêm vào dạng: **' ; EXEC xp\_cmdshell ‘cmd.exe dir C: '**.

Lúc này hệ thống sẽ thực hiện lệnh liệt kê thư mục trên ổ đĩa C:\ cài đặt server. Việc phá hoại kiểu nào tuỳ thuộc vào câu lệnh đằng sau cmd.exe.

Stored Procedure được sử dụng trong lập trình Web với mục đích giảm sự phức tạp trong ứng dụng và tránh sự tấn công trong kĩ thuật SQL Injection. Tuy nhiên hacker vẫn có thể lợi dụng những Stored Procedure để tấn công vào hệ thống.

Ví dụ: Stored procedure sp\_login gồm hai tham số là username và password. Nếu nhập:

Username: nhimmap

Password: ‘;shutdown--

Lệnh gọi stored procedure như sau

exec sp\_login ‘nhimmap’,‘’;shutdown--’

Lệnh shutdown thực hiện dừng SQL Server ngay lập tức.

**2.Phòng chống**

* Nhận dữ liệu kiểu INT
* Viết lại đường dẫn có thể chống SQL Injection
* Sử dụng hàm sprintf và mysql\_real\_escape\_string để các định kiểu dữ liệu cho câu truy vấn.

1. **CSRF (Cross-site Request Forgery)**:
   1. **Các dạng tấn công**

Tấn công sử dụng kỹ thuật này dành cho người am hiểu về hệ thống, có thể đã từng phát triển hệ thống đó, hoặc một mã nguồn mở, hoặc một mã nguồn nào đó đã được công khai code. Hacker thực hiện gửi tin nhắn dến Admin, khi admin đọc tin nhắn này trình duyệt sẽ request đến link đó và lấy cookie của trình duyệt và tiến hành active. Trường hợp không gửi được mail, giả sử ta biết rằng admin đang login hacker có thể send 1 trang web mà hacher lập ra, trong đó có đoạn code độc hại rồi send qua yahoo hay gì gì đó, khi đó admin viếng thăm vào và thực hiện các thao tác trên. Như vậy hacker thực hiện một truy vấn trái phép dựa vào chính người dùng

* 1. **Phòng chống**
* Thông thường để tránh tấn công ta sẽ chia làm hai đối tượng, một là đối tượng coder và hai là đối tượng người dùng cuối (user).  
  – Với đối tượng người dùng cuối thì:  
  – Hạn chế sử dụng login vào hệ thống khi nói chuyện tiếp xúc với những người lạ qua các kênh khác nhau, những email không rõ nguồn gốc. Khi không dùng hệ thống thì lập tức logout.  
  – Nên login vào một máy riêng và không cho người thứ 2 tiếp xúc với máy đó.  
  – Thay đổi mật khẩu liên tục, và chọn những mật khẩu khó đoán, có kỹ tự đặc biệt. Vì hiện nay có rất nhiều phần mềm dò pass.  
  – Với đối tượng coder:  
  – Thực hiện tạo những token auto và random với từng máy, từng trình duyệt và thiết lập thời gian sống cho token đó.  
  – Không sử dụng phương thức GET với những request mà có ảnh hưởng đến CSDL.  
  – Khi lấy dữ liệu từ người dùng thì kiểm tra chặt chẽ.  
  – URL trong admin càng khó nhớ càng bí hiểm càng tốt.

1. **Chèn mã HTML và XSS**
   1. **Các loại XSS và biện pháp phòng chống**

Cho phép kẻ tấn công nhúng mã đọc Javacsript, VBScript, ActiveX, HTML, hoặc Flash vào một trang năng động, dễ bị đánh lừa người sử dụng, thực hiện kịch bản trên máy tính của mình để thu thập dữ liệu. Kỹ thuật này không tấn công vào CSDL hệ thống như SQL injection mà chúng tấn công trực tiếp từ phía người dùng bằng cách xâm nhập hệ thống bằng các đoạn mã đơn giản để lấy cắp cookies và session từ đó chúng có thể thao túng người dùng cướp quyền truy cập tài khoản mà không cần tới mật khẩu.

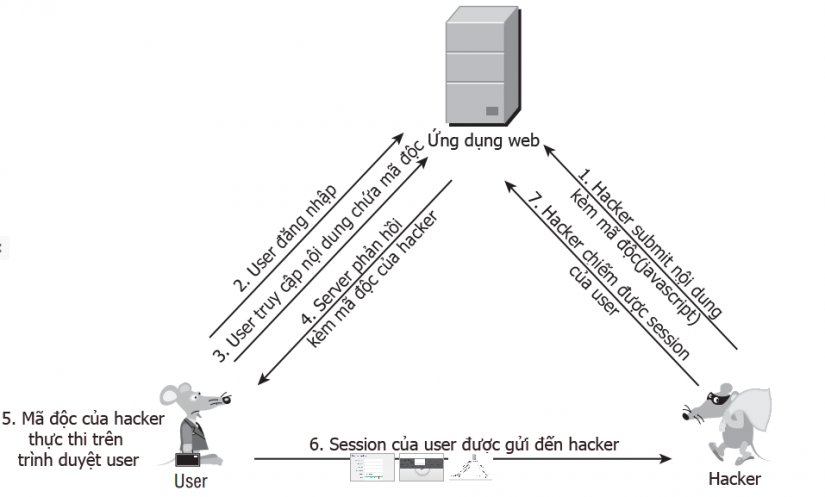
* Non-persistent(Reflected) là loại phổ biến nhất: Loại này xuất hiện khi dữ liệu được cung cấp từ một web client nào đó. Hacker khi muốn tấn công thì điều đầu tiên là sẽ phải tìm ra lỗ hổng bảo mật trên website bằng cách gắn một đoạn mã test vào web client để web client gửi đến server và chờ phản hồi của web server để tìm ra lỗ hổng bảo mật.Hacker tấn công dựa vào sự thiếu chú ý về việc lọc dữ liệu vào từ URL vủa website và gắn thêm những đoạn mã độc vào đây để thực hiện hành vi tấn công website. Loại này thì chỉ có tác dụng trong một lần.

Ví dụ : Có thể một request được gửi từ các form dữ liệu hoặc cũng có thể chỉ là các URL: http://www.example.com/search?query=alert('XSS was found !');

Stored XSS: Là một biến thể tàn phá gây hậu quả rất nặng nề. Loại này xảy ra khi dữ liệu do các hacker cung cấp được lưu trữ trên các máy chủ thông qua một số chức năng trên website và từ đó về sau thì các dữ liệu này hiển nhiên được hiển thị một cách bình thường trên các trình duyệt của người dùng mà không cần tới HTML riêng nữa. Và khi người dùng click vào những phần bị gắn mã độc thì đã bị dính XSS. Đoạn mã chèn thêm vào được lưu trữ vào CSDL trên server dưới dạng các comment trong blog, mesage, forum hay visitor log

Ví dụ: Khi đăng ký thành viên, phần giới thiêu về bản thân, nếu hacker nhập vào mã XSS và website không kiểm tra kỹ dữ liệu đầu vào, thì mỗi khi truy cập trang thành viên của hacker đó, bạn sẽ bị khai thác.

Mô hình XSS:



* 1. **Các kỹ thuật vượt qua các bộ lọc XSS**

**a)** bypass magic\_quotes\_gpc  
- Khi php.ini ấn định magic\_quotes\_gpc=on, nó có nghĩa là server sẽ không cho phép ta dùng " và ' . Vì vậy khi ta chèn <script>alert('hello world!')</script> sẽ không thực hiện được vì đã phạm luật.  
- Thử convert sang ascii xem sao: hello world! ==> String.fromCharCode(104, 101, 108, 108, 111, 32, 119, 111, 114, 108, 100, 33) ==> <script>alert(String.fromCharCode(104, 101, 108, 108, 111, 32, 119, 111, 114, 108, 100, 33))</script>. Câu này hoàn toàn hợp lệ và không phạm luật.  
  
**b)** bypass with cryption in full html  
- Rất đơn giản, chúng ta chỉ cần mã hoá (dùng urlencode của hackbar hoặc các tool khác)  
- Ví dụ: <script>alert('hello world!')</script>  
==> %3Cscript%3Ealert%28%27hello%20world%21%27%29%3C%2fscript%3E (nếu cần mã hoá luôn chữ).  
  
**c)** bypass with Obfuscation  
Trường hợp này khi bộ lọc kiểm tra chuổi alert, script v.v.v.  
Để vượt được nó ta chỉ cần làm sai lệch 1 tý ví dụ: AlERt, ScRipT v.v.v.  
Ví dụ: <ScRipT>AlERt(123456)</ScRipT>  
  
**d)** bypass with trying around method (sử dụng các method)  
- Vấn đề này hay nhắc đến ở các khung tìm kiếm dùng thẻ input, lúc này ta chỉ cần thêm "> ("> nghĩa là đóng thẻ input nếu như value nằm cuối cùng) sau đó chèn thêm <script>alert('hello world!')</script>. ví dụ  
http://target.com/search.php?search="><script>alert('hello world!')</script>

* 1. **Một số tấn công XSS trên thực tế**
  2. **Phòng chống**

1. **Tấn Công vào cơ chế xác thực**
   1. **Các dạng:**
      1. Xem trộm thông tin (Release of Message Content)

Trong trường hợp này Trudy chặn các thông điệp Alice gửi cho Bob, và xem được nội dung của thông điệp.

Alice

Bob

Network

Đ

ọ

c n

ộ

i dung thông

đi

ệ

p

c

ủ

a Alice

Trudy



**Hình 1-1. Xem trộm thông điệp**

* + 1. Thay đổi thông điệp (Modification of Message)

Trudy chặn các thông điệp Alice gửi cho Bob và ngăn không cho các thông điệp này đến đích. Sau đó Trudy thay đổi nội dung của thông điệp và gửi tiếp cho Bob. Bob nghĩ rằng nhận được thông điệp nguyên bản ban đầu của Alice mà không biết rằng chúng đã bị sửa đổi.

Alice

Bob

Network

S

ử

a thông đi

ệ

p

c

ủ

a

Alice

g

ử

i cho

Bob

Trudy



* + 1. Mạo danh (Masquerade)

Trong trường hợp này Trudy giả là Alice gửi thông điệp cho Bob. Bob không biết điều này và nghĩ rằng thông điệp là của Alice.

Network

Trudy

gi

ả

là

Alice

g

ở

i

thông đi

ệ

p

cho Bob

Trudy



Alice Bob

**Hình 1-3. Mạo danh**

* + 1. Phát lại thông điệp (Replay)

Trudy sao chép lại thông điệp Alice gửi cho Bob. Sau đó một thời gian Trudy gửi bản sao chép này cho Bob. Bob tin rằng thông điệp thứ hai vẫn là từ Alice, nội dung hai thông điệp là giống nhau. Thoạt đầu có thể nghĩ rằng việc phát lại này là vô hại, tuy nhiên trong nhiều trường hợp cũng gây ra tác hại không kém so với việc giả mạo thông điệp. Xét tình huống sau: giả sử Bob là ngân hàng còn Alice là một khách hàng. Alice gửi thông điệp đề nghị Bob chuyển cho Trudy 1000$. Alice có áp dụng các biện pháp như chữ ký điện tử với mục đích không cho Trudy mạo danh cũng như sửa thông điệp. Tuy nhiên nếu Trudy sao chép và phát lại thông điệp thì các biện pháp bảo vệ này không có ý nghĩa. Bob tin rằng Alice gửi tiếp một thông điệp mới để chuyển thêm cho Trudy 1000$ nữa.

Alice

Bob

Network

Sao chép thông đi

ệ

p

c

ủ

a

Alice

và g

ử

i l

ạ

i sau cho Bob

Trudy



**Hình 1-4. Phát lại thông điệp**

* Dùng các đoạn mã phá hoại (Malware): như virus, worm, trojan, backdoor… những đoạn mã độc này phát tán lan truyền từ máy tính này qua máy tính khác dựa trên sự bất cẩn của người sử dụng, hay dựa trên các lỗi của phần mềm. Lợi dụng các quyền được cấp cho người sử dụng (chẳng hạn rất nhiều người login vào máy tính với quyền administrator), các đoạn mã này thực hiện các lệnh phá hoại hoặc dò tìm password của quản trị hệ thống để gửi cho hacker, cài đặt các cổng hậu để hacker bên ngoài xâm nhập.
* Thực hiện các hành vi xâm phạm (Intrusion): việc thiết kế các phần mềm có nhiểu lỗ hổng, dẫn đến các hacker lợi dụng để thực hiện những lệnh phá hoại. Những lệnh này thường là không được phép đối với người bên ngoài, nhưng lỗ hổng của phần mềm dẫn đến được phép. Trong những trường hợp đặc biệt, lỗ hổng phần mềm cho phép thực hiện những lệnh phá hoại mà ngay cả người thiết kế chương trình không ngờ tới. Hoặc hacker có thể sử dụng các cổng hậu do các backdoor tạo ra để xâm nhập.
  1. **Phòng chống**

Để khắc phục các hành động phá hoại này, người ta dùng các chương trình có chức năng gác cổng, phòng chống. Những chương trình này dò tìm virus hoặc dò tìm các hành vi xâm phạm đển ngăn chặn chúng, không cho chúng thực hiện hoặc xâm nhập. Đó là các chương trình chống virus, chương trình firewall… Ngoài ra các nhà phát triển phần mềm cần có quy trình xây dựng và kiểm lỗi phần mềm nhằm hạn chế tối đa những lỗ hổng bảo mật có thể có.

Hệ Thống Thông Tin

Con ng

ư

ờ

i

:

hacker.

Ph

ầ

n m

ề

m: virus, worm…



-

Các tài nguyên tính toán

(

b

ộ

nh

ớ

, chíp x

ử

lý…)

-

D

ữ

li

ệ

u

-

Các ti

ế

n trình

-

Ph

ầ

n m

ề

m

-

Các tài nguyên m

ạ

ng

Kênh truy c

ậ

p

Ch

ứ

c năng

gác c

ổ

ng