OLE Object are still Dangerous Today
— Exploiting Microsoft Office

wh1tc @ Sangfor
Dr. Zhiniang Peng @ Sangfor

Whoami

Zhiniang Peng @edwardzpeng

the Chief Architect & Principal Security Researcher at Sangfor

PhD in Cryptography, interested in all areas of Computer Science

Work in Defensive & Offensive security

Published many research in both Industry & Academia

https://sites.google.com/site/zhiniangpeng

Whoami

wh1tc @wh1tc

Security Researcher at Sangfor

Focus on Fuzzing & Windows Applications

Ranked #7 on MSRC MVR in 2023

Agenda

- Introduction
- Review of Office Vulnerabilities in last 10 years
- Finding and Exploiting Vulnerabilities in Office
- Summary and Future Work

Introduction

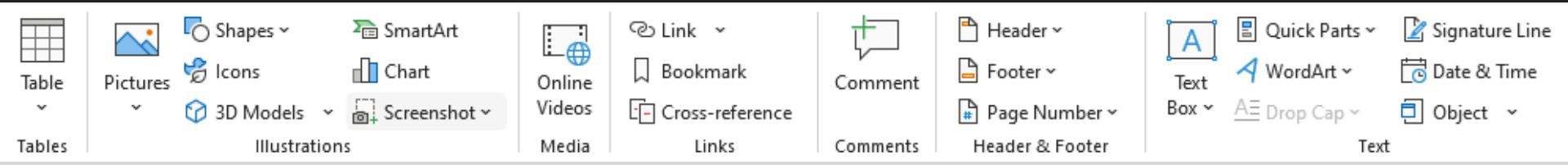
Introduction to Office

- Microsoft Office is a suite of productivity software
  - Word, Excel, PowerPoint, Publisher and so on.

- We are focused on desktop application installed in Windows.
  - Most popular

Introduction to Office

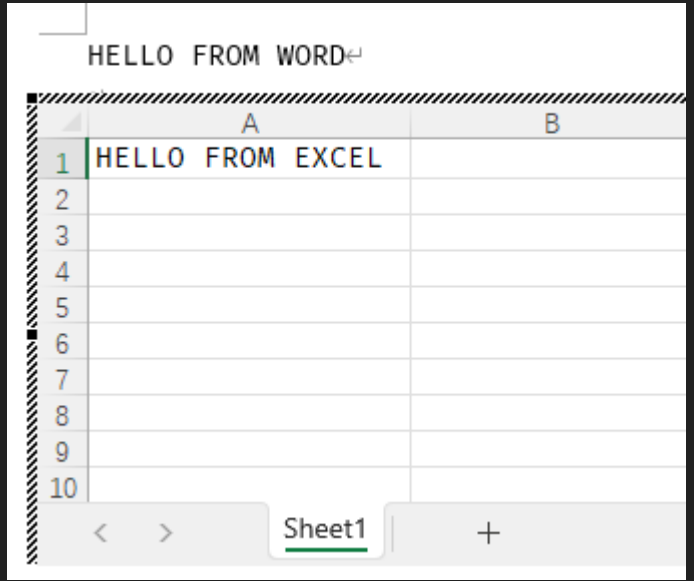
- Microsoft Office has many features
  - Using Word as example:



- Insert Text
- Insert Image
- Insert Tables
- Insert Object?
  - OLE

OLE

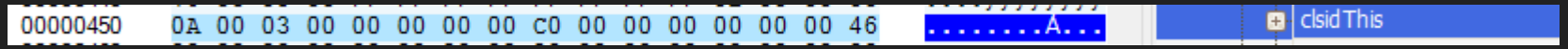
- Object Linking and Embedding
  - Allows users to create and edit documents containing "objects" created by multiple applications.
  - Such as: spreadsheets, bitmaps, pdf



OLE in Office

- Using a docx document with an embedded OLE object (Bitmap Image Object) as an example.
  - In the OpenXML format, OLE objects are present in the form of OLESSFormat.

  - The object class GUID（CLSID） that is stored in the root directory entry can be used for COM activation of the document's application.
    - You can use the OffVis tool to observe

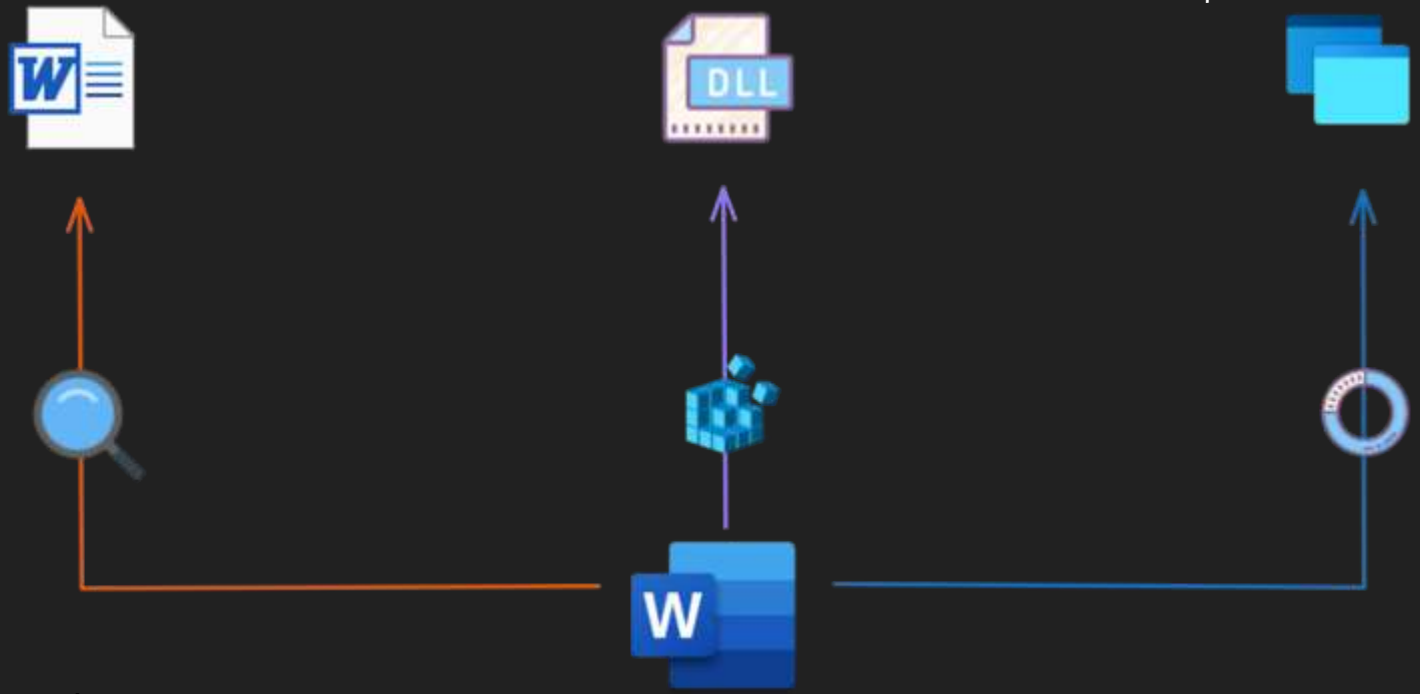


How OLE works in Office

1. get CLISD from document

2. CoCreateInstance

3. IpersistStorage



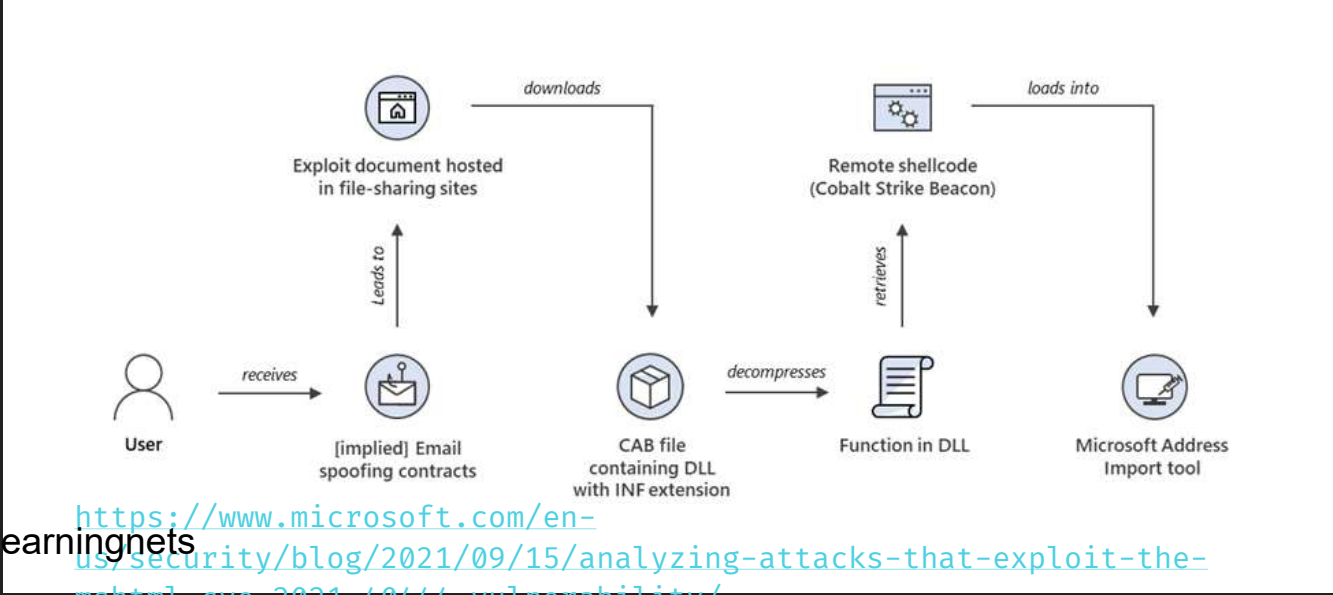
Review of Office Vuls in last 10 years

Motivation

- The disclosure of CVE-2021-40444
  - https://msrc.microsoft.com/update-guide/vulnerability/CVE-2021-40444
  - Attracted our attention, we started researching Office
  - Logic bug
    - Severe and widespread impact
    - Reliable exploit

Attack chain of CVE-2021-40444

- StdOleLink (htmlfile via URL moniker)
- Execute JavaScript in evil.html
  - downloads cab file and drop evil.inf in the 'Temp' directory.
  - Open URL Scheme: .cpl:../../../AppData/Local/Temp/evil.inf



https://www.microsoft.com/en-us/security/blog/2021/09/15/analyzing-attacks-that-exploit-the-mshtml-cve-2021-40444-vulnerability/

Previous vulnerability in the wild

- Office have a long and complicated vulnerabilities history
- We try to do some summary before actually hunting vulnerabilities
- Qi Li and Quan Jin at BlueHat Shanghai 2019 has a good summary
  - https://images.seebug.org/archive/Catch_Multiple_Zero-Days_Using_Sandbox-EN.pdf
  - We did our summary based on some of their slides
- Widely discussed vulnerabilities selected

Exploited vuls in last 10 years

2014	2015	2016	2017	2018
CVE-2014-1761 CVE-2014-4114 CVE-2014-6352	CVE-2015-1642 CVE-2015-2424 CVE-2015-2545 CVE-2015-5119 CVE-2015-5122	CVE-2016-4117 CVE-2016-7193 CVE-2016-7855	CVE-2017-0199 CVE-2017-0261 CVE-2017-0262 CVE-2017-8570 CVE-2017-8759 CVE-2017-11292 CVE-2017-11826 CVE-2017-11882	CVE-2018-0798 CVE-2018-0802 CVE-2018-4878 CVE-2018-5002 CVE-2018-8174 CVE-2018-8373 CVE-2018-15982
2019	2020	2021	2022	2023
	CVE-2020-0674 CVE-2020-0968	CVE-2021-40444	CVE-2022-30190 CVE-2022-41128	CVE-2023-36884

Short Summary

- We can observe a trend from the previous table
  - The number of exploitable vulnerabilities in the wild has decreased
    - The data originates from publicly disclosed online sources
  - Attackers tend to use logic bugs nowadays
    - The difficulty of exploiting memory corruption vulnerabilities has increased due to various mitigations
    - Logic bugs can be exploited reliably and are attacker-friendly

Vulnerability Classification

RTF Control Word Parsing Problem	Open XML Tag Parsing Problem	Embedded OLE object Parsing Problem	Office Embedded Flash 0day（* Subset of Embedded OLE）
CVE-2010-3333 CVE-2014-1761 CVE-2016-7193	CVE-2015-1641 CVE-2017-11826	CVE-2012-0158 CVE-2015-2424 MS15-132 CVE-2017-11882 CVE-2018-0798 CVE-2018-0802	CVE-2011-0609 CVE-2011-0611 CVE-2013-0634 CVE-2018-4878 CVE-2018-5002 CVE-2018-15982…
EPS File Parsing Problem	Moniker	Embedded IE 0day（*Subset of Moniker）	Other Office Logic Vulnerabilities
CVE-2015-2545 CVE-2017-0261 CVE-2017-0262	CVE-2017-0199 CVE-2017-8570 CVE-2017-8759	CVE-2018-8174 CVE-2018-8373 CVE-2020-0674 CVE-2020-0968 CVE-2021-40444 CVE-2022-30190 CVE-2022-41128 CVE-2023-36884	CVE-2014-4114 CVE-2014-6352 CVE-2015-0097

Vulnerability Classification

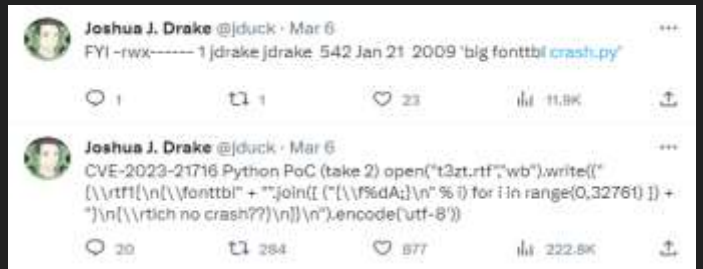
RTF Control Word Parsing Problem	Open XML Tag Parsing Problem	Embedded OLE object Parsing Problem	Office Embedded Flash 0day（* Subset of Embedded OLE）
CVE-2010-3333 CVE-2014-1761 CVE-2016-7193	CVE-2015-1641 CVE-2017-11826	CVE-2012-0158 CVE-2015-2424 MS15-132 CVE-2017-11882 CVE-2018-0798 CVE-2018-0802	CVE-2011-0609 CVE-2011-0611 CVE-2013-0634 CVE-2018-4878 CVE-2018-5002 CVE-2018-15982…
EPS File Parsing Problem	Moniker	Embedded IE 0day（*Subset of Moniker）	Other Office Logic Vulnerabilities
CVE-2015-2545 CVE-2017-0261 CVE-2017-0262	CVE-2017-0199 CVE-2017-8570 CVE-2017-8759	CVE-2018-8174 CVE-2018-8373 CVE-2020-0674 CVE-2020-0968 CVE-2021-40444 CVE-2022-30190 CVE-2023-36884	CVE-2014-4114 CVE-2014-6352 CVE-2015-0097

RTF Control Word/Open XML Tag Parsing Problem

- Combined because they both relate to specific markers or identifiers in their respective formats
- Both involve memory corruption vulnerabilities.
  - Exploiting these vulnerabilities requires precise control over memory
  - Normally you don’t have good primitive to do memory manipulation here
  - Exploitation is hard with nowadays’ mitigations

RTF Control Word/Open XML Tag Parsing Problem

- The related vulnerabilities still exist.
  - CVE-2023-21716 Microsoft Word Remote Code Execution Vulnerability
  - https://msrc.microsoft.com/update-guide/vulnerability/CVE-2023-21716
- However, exploiting these vulnerabilities is much more difficult than before.
  - No memory corruption exploitation has been detected in last 5 years.



Vulnerability Classification

RTF Control Word Parsing Problem	Open XML Tag Parsing Problem	Embedded OLE object Parsing Problem	Office Embedded Flash 0day（* Subset of Embedded OLE）
CVE-2010-3333 CVE-2014-1761 CVE-2016-7193	CVE-2015-1641 CVE-2017-11826	CVE-2012-0158 CVE-2015-2424 MS15-132 CVE-2017-11882 CVE-2018-0798 CVE-2018-0802	CVE-2011-0609 CVE-2011-0611 CVE-2013-0634 CVE-2018-4878 CVE-2018-5002 CVE-2018-15982…
EPS File Parsing Problem	Moniker	Embedded IE 0day（*Subset of Moniker）	Other Office Logic Vulnerabilities
CVE-2015-2545 CVE-2017-0261 CVE-2017-0262	CVE-2017-0199 CVE-2017-8570 CVE-2017-8759	CVE-2018-8174 CVE-2018-8373 CVE-2020-0674 CVE-2020-0968 CVE-2021-40444 CVE-2022-30190 CVE-2023-36884	CVE-2014-4114 CVE-2014-6352 CVE-2015-0097

EPS File Parsing Problem

- EPS was once an image format supported by Office
  - Contains PostScript program code
    - You have the ability to do some memory manipulation
    - Vulnerabilities are more likely to be exploited

- This attack vector is no longer exist today
  - From April 2017, Office have turned off the ability to insert EPS files into Office documents

Resource files Parsing Problem

- Apart from EPS, Office also supports the insertion of various resource files, including 3D models.
- There are still quite many vulnerabilities.
- And history is always similar
  - (June 1, 2023) UPDATE: The ability to insert SketchUp graphics (.skp files) has been temporarily disabled in Office

Vulnerability Classification

RTF Control Word Parsing Problem	Open XML Tag Parsing Problem	Embedded OLE object Parsing Problem	Office Embedded Flash 0day（* Subset of Embedded OLE）
CVE-2010-3333 CVE-2014-1761 CVE-2016-7193	CVE-2015-1641 CVE-2017-11826	CVE-2012-0158 CVE-2015-2424 MS15-132 CVE-2017-11882 CVE-2018-0798 CVE-2018-0802	CVE-2011-0609 CVE-2011-0611 CVE-2013-0634 CVE-2018-4878 CVE-2018-5002 CVE-2018-15982…
EPS File Parsing Problem	Moniker	Embedded IE 0day（*Subset of Moniker）	Other Office Logic Vulnerabilities
CVE-2015-2545 CVE-2017-0261 CVE-2017-0262	CVE-2017-0199 CVE-2017-8570 CVE-2017-8759	CVE-2018-8174 CVE-2018-8373 CVE-2020-0674 CVE-2020-0968 CVE-2021-40444 CVE-2022-30190 CVE-2023-36884	CVE-2014-4114 CVE-2014-6352 CVE-2015-0097

Embedded OLE object Parsing Problem

- CVE-2017-11882 is a classic example
  - Microsoft Equation Editor Vulnerability
    - No mitigation in EQNEDT32.EXE（DEP & ASLR）
    - It was removed from all versions in the January 2018 Public Update

- Flash is actually a subset of Embedded OLE object
  - Due to its significant impact, it was categorized separately
    - There were numerous security vulnerabilities
  - Flash were blocked in Office Monthly Channel starting in June 2018

Embedded OLE object Parsing Problem

- Not only memory corruption bug, but also logical bug.
- Mention by Haifei Li and Bing Sun in Blackhat USA 2015.
  - Attacking Interoperability: An OLE Edition
    - https://www.blackhat.com/docs/us-15/materials/us-15-Li-Attacking-Interoperability-An-OLE-Edition.pdf
- DLL-Preloading Vulnerability
  - Will result in loading a DLL from the current working directory
  - Details will be discussed later

Vulnerability Classification

RTF Control Word Parsing Problem	Open XML Tag Parsing Problem	Embedded OLE object Parsing Problem	Office Embedded Flash 0day（* Subset of Embedded OLE）
CVE-2010-3333 CVE-2014-1761 CVE-2016-7193	CVE-2015-1641 CVE-2017-11826	CVE-2012-0158 CVE-2015-2424 MS15-132 CVE-2017-11882 CVE-2018-0798 CVE-2018-0802	CVE-2011-0609 CVE-2011-0611 CVE-2013-0634 CVE-2018-4878 CVE-2018-5002 CVE-2018-15982…
EPS File Parsing Problem	Moniker	Embedded IE 0day	Other Office Logic Vulnerabilities
CVE-2015-2545 CVE-2017-0261 CVE-2017-0262	CVE-2017-0199 CVE-2017-8570 CVE-2017-8759	CVE-2018-8174 CVE-2018-8373 CVE-2020-0674 CVE-2020-0968 CVE-2021-40444 CVE-2022-30190 CVE-2023-36884	CVE-2014-4114 CVE-2014-6352 CVE-2015-0097

Moniker Problem

- Presented by Haifei Li and Bing Sun in Syscan360 2017
  - Moniker Magic: Running Scripts Directly in Microsoft Office
    - https://www.blackhat.com/docs/us-15/materials/us-15-Li-Attacking-Interoperability-An-OLE-Edition.pdf
- Microsoft banned some CLSIDs in the fix.
  - CVE-2017-0199: htafile via URL Moniker / “Script” Moniker
  - CVE-2017-8570: CompositeMoniker/FileMoniker/NewMoniker/Scriptletfile
  - CVE-2017-8579: SOAPMoniker
- OLE “ StdOleLink ” feature can still run moniker/COM objects

Office Embedded IE 0day

- Evolved from moniker vulnerabilities
  - Use “ StdOleLink ” feature to load the html
  - The parsing module is handled by the IE module
- CVE-2018-8174/8373
  - Attacker Exploited an IE VBScript 0day
    - VBScript was disabled by default in August 2019
- CVE-2020-0674/CVE-2021-40444
  - Attacker Exploited an IE JavaScript 0day
    - JavaScript was disabled by default in January 2023

Current Landscape

RTF Control Word Parsing Problem	Open XML Tag Parsing Problem	Embedded OLE object Parsing Problem	Office Embedded Flash 0day（* Subset of Embedded OLE）
Hard to exploit	Hard to exploit	Let’s go!	End of life
EPS File Parsing Problem	Moniker	Embedded IE 0day （*Subset of Moniker）	Other Office Logic Vulnerabilities
End of life	Still there	End of life	Still there

Finding and Exploiting Vuls in Office

Finding and exploiting vulnerabilities in office

- Previous Related Work
- Vulnerability patterns

Previous Related Work

- Attacking Interoperability: An OLE Edition
  - https://www.blackhat.com/docs/us-15/materials/us-15-Li-Attacking-Interoperability-An-OLE-Edition.pdf
  - by Haifei Li and Bing Sun in Blackhat USA 2015
  - Special thanks to Haifei Li for his years of sharing;

Three vuls patterns in Embedded OLE object

- Memory Corruption in CoCreateInstance
- Memory Corruption in IPersistStorage
- DLL preloading attacks

Type-1: Memory Corruption in CoCreateInstance

- Vulnerability pattern
- Auto discovery
- Case study
- Exploitability analysis

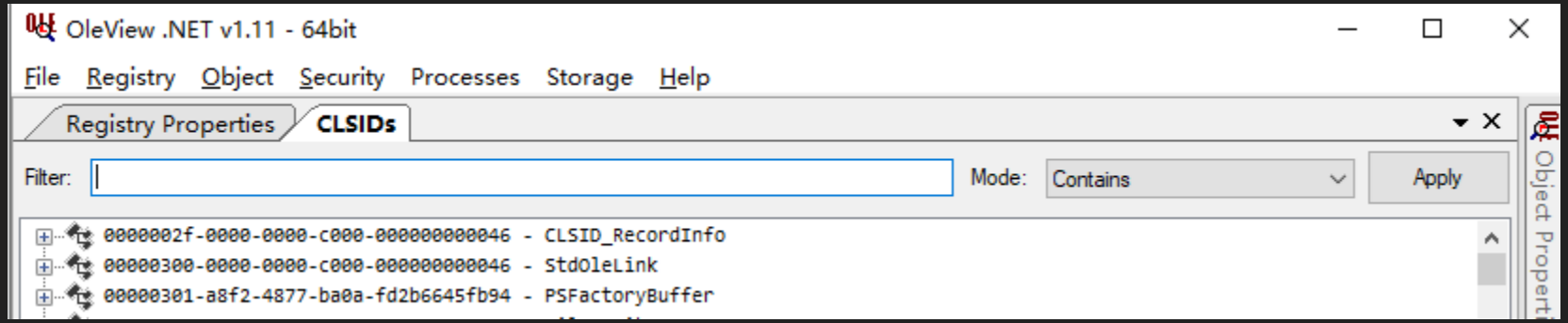
Type-1: Vulnerability pattern

- The CLSID from document can be controlled by attacker.
  - untrusted!
- The object associated with the specified CLSID may be not OLE object
  - Several thousand CLSID but only a few hundred OLE objects
- To determine whether it's an OLE object, Office must first load the object
  - Even the object is not designed to load by Office
  - This can lead to many security issues.

Type-1: Auto discovery

➢ Just let Office load all CLSID-Associated object

1. Collect all CLSIDs in the system
    - We use the OleViewDotNet designed by James Forshaw
        - https://github.com/tyranid/oleviewdotnet

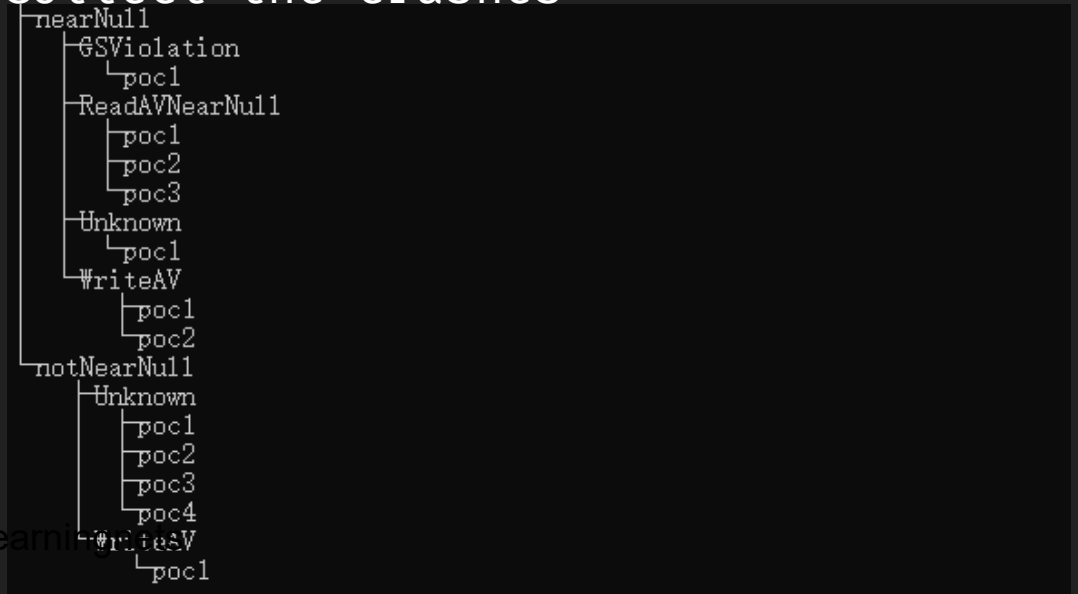


Type-1: Auto discovery

2. Choose the template
    - According to Tavis Ormandy's report in 2015
        - https://bugs.chromium.org/p/project-zero/issues/detail?id=514#c2
    - The trigger will load the "OLE" object without user click with the following RTF document
        - {\rtf1{\object\objemb{*\objclass None}{*\oleclsid \'7b<clsid>\'7d}{*\objdata 01050000010000000100000000000000000000000000000000000000000000000000000000000000000000000000000000}}}

Type-1: Auto discovery

3. Using a “fuzzing” framework
    - Auto open and close
    - Collect the crashes



Type-1: CaseStudy

- Unexpectedly we still discovered many low-hanging fruit vulnerabilities.
- We conducted the tests twice:
  - Win10 in 2021
    - CVE-2022-21878 CVE-2022-21888 CVE-2022-21971 CVE-2022-21992 CVE-2022-21974
  - windows 11 & Windows Server in 2023:
    - CVE-2023-29366 CVE-2023-29367 CVE-2023-35313 CVE-2023-35323 CVE-2023-36704
  - Visual Studio in 2023: CVE-2023-28296
- Perhaps there will be more in 2024?

Type-1: CaseStudy

- ➢ CVE-2022-21971 Windows Runtime Remote Code Execution Vulnerability
  - Root Cause Analysis:
    - In CoCreateInstance, the WapAuthProvider::CreateInstance method is invoked
      - result = operator new(0x78ui64, &std::nothrow);
      - Only partial initialization was performed.
    - In function “prauthproviders!WapAuthProvider::~WapAuthProvider”:
      - v3 = *(void **)(this + 0x50); // Uninitialized pointer
      - LocalFree(v3);
    - Enable full pageheap and observe in windbg:
      - prauthproviders!WapAuthProvider::~WapAuthProvider+0x38:
        - mov rcx,qword ptr [rbx+50h] ds:00000244`9b65cfd0=c0c0c0c0c0c0c0c0
  - Summary:
    - Here existed a vulnerability of Access (Free) of Uninitialized Pointer

Type-1: Exploitability analysis

- The vulnerabilities under this pattern are memory corruption vulnerabilities.
- Requires clever exploitation:
  - 64bit
  - ASLR bypass
  - DEP bypass
  - CFG bypass
- It's possibly exploitable but quite challenging, requiring strong technical skills and a significant amount of time.

Type-2: Memory Corruption in IPersistStorage

- Vulnerability pattern
- Autodiscovery
- Case study
- Exploitability analysis

Type-2: Vulnerability pattern

- The CLSID from document can be controlled by attacker.
  - If it is an OLE object
- Office load the object and parse the data from Storage
  - Can be controlled by attacker. (still untrusted)
- In most cases, Storage is in binary format.
  - There may be potential security risks.
  - Fuzzing!

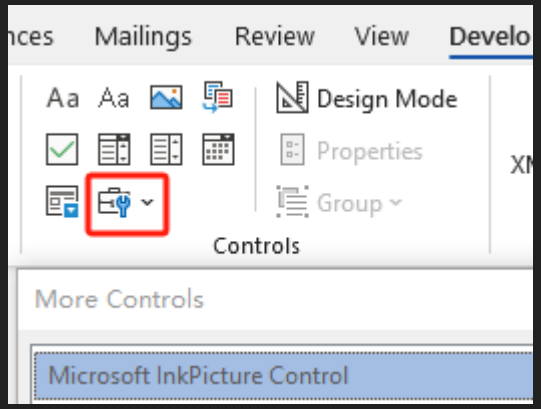
Type-2: Auto discovery

1. Collect all the OLE objects in the system
    - We still use the Oleviewdotnet designed by James Forshaw
    - Get-ComClass | Select-ComClassInterface -Name "IOleObject"> IOleObjectFile.txt
2. Write the wrapper
    - StgCreateStorageEx
    - SHCreateStreamOnFileEx // Data to fuzz
    - CoCreateInstance
    - IPersistStorage->Load // Function to fuzz
    - Release

Type-2: Auto discovery

3. Collect the corpus
    - It is hard to collect corpus from Internet
    - We used the 'ActiveX control' feature to construct the corpus manually

4. Start Fuzzing



Type-2: CaseStudy

- ➢ CVE-2022-23290 Windows Inking COM Elevation of Privilege Vulnerability(?
  - Root Cause Analysis:
    - In IpersistStorage, the CSketchInk::IPersistStreamInit_Load method is invoked
      - HeapAlloc(*(HANDLE *)Default, *((_DWORD *)Default + 2), 0x70)
      - Only partial initialization was performed.
    - In function “InkObj!CSketchInk::FreeStrokeList:
      - v6 = *(void **)(this + 0x10); // Uninitialized pointer
      - HeapFree(*(HANDLE *)Default, *((_DWORD *)Default + 2), v6)
    - Enable full pageheap and observe in windbg:
      - InkObj!CSketchInk::FreeStrokeList+0x3d:（Simplified the logic here）
        - mov rdi,qword ptr [rax+10h] ds:00000158`42fcbfa0=c0c0c0c0c0c0c0c0
  - Summary:
    - Here existed a vulnerability of Access (Free) of Uninitialized Pointer

Type-2: Exploitability analysis

- The vulnerabilities is also memory corruption.
- Requires clever exploitation:
  - 64bit
  - ASLR bypass
  - DEP bypass
  - CFG bypass
- It's possibly exploitable but quite challenging, requiring strong technical skills and a significant amount of time.

Short Summary

- ➢ We did indeed discover some memory corruption vulnerabilities through these two vulnerability patterns.
- ➢ Exploiting these vulnerabilities is possible exploitable but highly challenging.
- ➢ Instead of spending significant time in exploiting these vulnerabilities, we prefer to find some logic bugs.

Type-3: DLL preloading attacks

- Vulnerability pattern
- Autodiscovery
- Case study
- Exploitability analysis

Type-3: Vulnerability pattern

- During the process of loading the object, the LoadLibrary function may sometimes be invoked.
  - `HMODULE handle = LoadLibrary("schannel.dll");`
- There is a risk that if the file is not present, the application may try to load the file from the current working directory.
- Microsoft once specifically addressed this issue in 2017, but it seems they have forgotten about it.
  - https://support.microsoft.com/en-gb/topic/secure-loading-of-libraries-to-prevent-dll-preloading-attacks-d41303ec-0748-9211-f317-2edc819682e1

Type-3: Auto discovery

- Our approach was inspired by WILL DORMANN
  - https://insights.sei.cmu.edu/library/attacking-com-via-word-rtf/
- Use the procmon to observe when testing the CLISD
  - With Filter:
    - Path: “C:\\test”
    - Result contains: not found.
- Sometimes procmon will crash because of out-of-memory
  - We use the openprocmon designed by progmboy instead
    - https://github.com/progmboy/openprocmon

Type-3: Case Study

- We discovered two vulnerabilities:
  - CVE-2023-36898 Tablet Windows User Interface Application Core Remote Code Execution Vulnerability
    - Windows 11 21H2 & 22H2 with default configuration
  - CVE-2023-35343 Windows Geolocation Service Remote Code Execution Vulnerability
    - Windows Server 2019 & 2022 with default configuration

Type-3: Case Study

- CVE-2023-35343 Windows Geolocation Service Remote Code Execution Vulnerability
  - Root Cause Analysis:
    - In CoCreateInstance, the GetFindMyDeviceEnabled method is invoked
      - LibraryW = LoadLibraryW(L"mdmcommon.dll");
      - “mdmcommon.dll” does not exist in Windows Server.
  - Summary:
    - If there is a malicious 'mdmcommon.dll' in the current directory, it could lead to RCE!

Type-3: Exploitability analysis

- Logic bug
  - Can be easily exploited this time!
- require a condition
  - It requires delivering both a malicious document and a malicious DLL in the same Current working directory

Weaponize the vulnerability

- Protected View Mode
- Choose the target
- Clarify the Current directory
- Deliver

Protected View Mode

- With Protected View Mode, basically all the stuff that could bring security or privacy risks are disabled
  - ActiveX
  - OLE
  - Macros
  - Remote resource loading

- Enable if WORD/Excel/PPT open the document with Mark Of The Web

Choose the target

➢ Let's find what target support OLE

Application	Introduction	Protected View Mode	User Interaction	Additional explanation
winword.exe 	Word processor in Office	Yes if MOTW.	1 ~ 2 click (without MOTW)	Defense-in-depth improvement in 1-click way
wordpad.exe 	Word processor in Windows	No. 	1 click (warning)	Deprecated in September 2023 （ removed in win12? ）
MSpub.exe 	Less popular app in Office	No. 	1 click	Defense-in-depth improvement

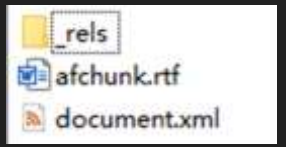
Choose the target: Winword

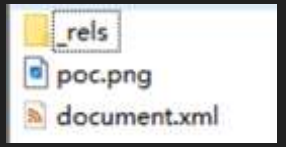
➢ Let's find the OLE activation methods in WinWord

File type	Introduction	OLE activation methods	User Interaction	Additional explanation
.doc	Binary	oleObject1.bin OLESS	2 click	Need click the OLE object
.docx	Open-XML	ActiveX1.xml OLESS	1 click	Pop up warning
.rtf	Rich Text	oleclsid objdata (progid)	1 click	oleclsid: defense-in-depth improvement

Choose the target: Winword

- Attackers prefer using the RTF format.
- But the RTF format is noticeable because it's less commonly used. Attacker always disguise.
  - .rtf -> .doc : just change the extension
  - .rtf -> .docx: Use <altChunk> to embedding RTF (samples:CVE-2023-36884)
    - poc.docx\word\afchunk.rtf
  - further hide:
    - <Default Extension="png" ContentType="application/rtf" />
    - poc.docx\word\poc.png





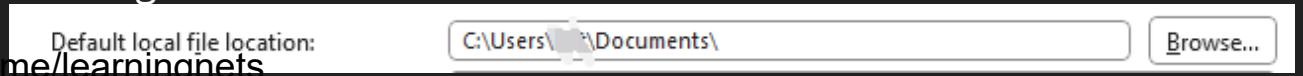
Current Directory

➢ Let's clarify the current directory

Attacker vector	Current directory	Exploitable?
wordpad/Mspub.exe via explorer.exe	where the document is located	Yes 
winword.exe via explorer.exe	1. where the document is located 2. C:\Users\%name%\Documents	1. Yes  2. No
winword.exe via Preview Pane	C:\Windows\system32\	No

Current Directory of Winword

- Clarify why current directory of Winword is C:\Users\%name%\Documents.
  1. The winword.exe will change the Current Directory to C:\Users\%name%\Documents after ~10 seconds.
     - So exploit will fail If the user quit protected mode after ~10 seconds.
  2. Open a new document when winword.exe is running
     - Using the original process
- C:\Users\%name%\Documents is the default local file location in the settings of office.



Current Directory of Winword

- If you don't get it, destroy it :>
- In fact, Protected Mode only protect you first time
  - If user believe the document once, Protected View Mode is no longer available for this document.
- So a crash will help us.
  - Crash will kill the running process.
    - new process without Protected Mode next time.
  - Actually Office designed many DOS vulnerabilities
    - MS: DOS does not meet the bar for servicing in a security update

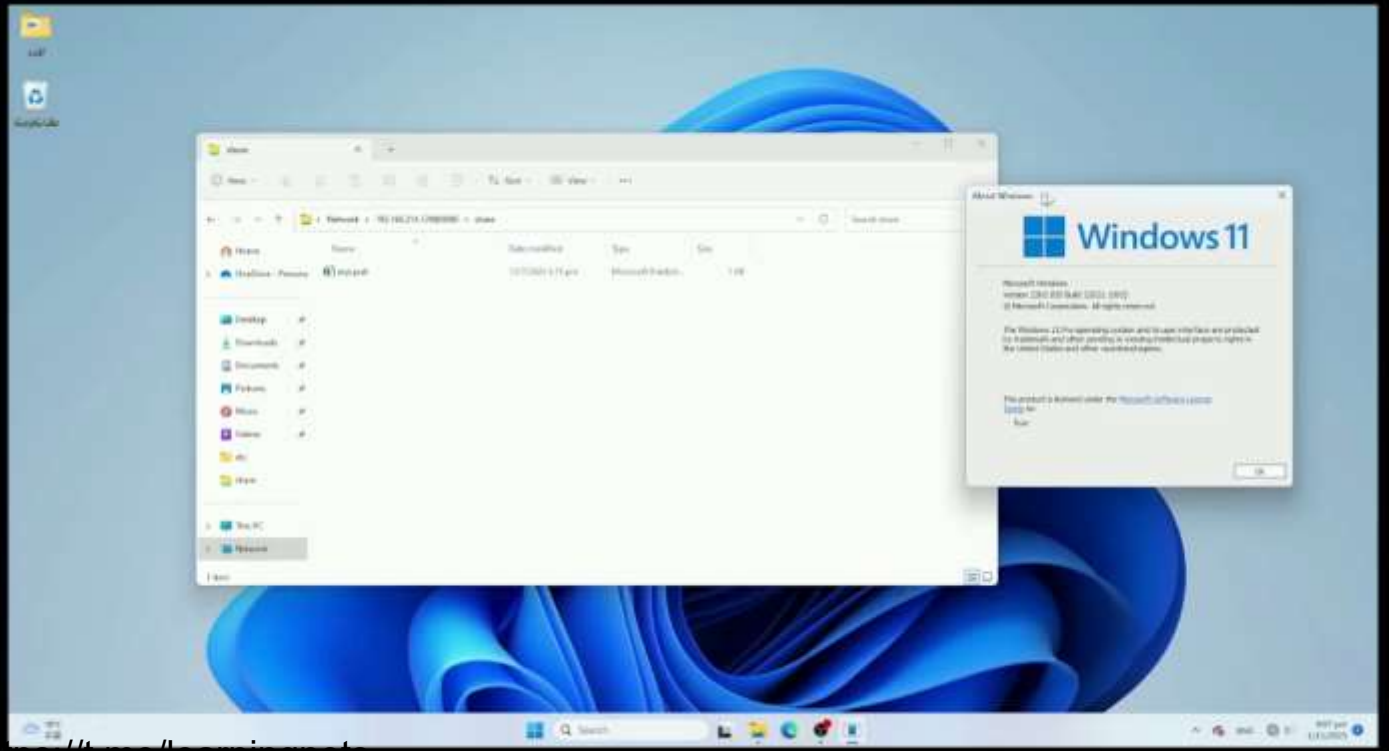
Deliver

➢ Now the exp is ready, Just Deliver!

Vector	Trick	Additional explanation
Zip	“hidden” via explorer.exe (Default)	Some archiver software don't propagate MOTW https://micahbabinski.medium.com/search-ms-webdav-and-chill-99c5b23ac462
Smb/ Webdav	Attackers can hide the DLL by modifying the PROPFIND method.	
ms-search	Guide users to view files on a remote UNC path	Research in trellix wrote an article about this method. https://www.trellix.com/en-us/about/newsroom/stories/research/beyond-file-search-a-novel-method.html

Demo1: Win11 + Wordpad + Zip

Demo2: Win11 + Mspub + webdav



Other applications support OLE

- Handling OLE is complex, vulnerability will definitely appear here again.
  - We did a simple reverse engineering of Office and did not found low hanging fruit.
- However Some office-like applications also support OLE. These applications also have vulnerabilities when processing OLE:
  - Vulnerabilities that have appeared in Office will also appear here
  - The custom features of these applications can also have vulnerabilities.

Demo3: RCE in office-like application

Detect & Defense

- For users:
  - Update the windows version
  - Use the Office in Current channel
    - Implemented defense-in-depth improvements
- For security companies:
  - Detect:
    - Focus on the vulnerable CLSID
  - Defense: Hotpatch
    - Target specific vulnerabilities:
      - switch from LoadLibraryW to LoadLibraryExW(dll,0x800)
    - Target specific Process:
      - Remove Workingdirectory when searching dll

Defense-in-depth improvement

- Office's improvement against OLE:
  - .RTF : can't load CLSID from "\oleclsid" RTF control word directly
    - still can load from Progid in "\objdata" RTF control word
  - .Pub : pop up Security Warning before load CLSID
- Affected Version
  - apply to
    - Current Channel / Monthly Enterprise
    - Semi-Annual Enterprise Channel: Version 2302
  - No apply to
    - Semi-Annual Enterprise Channel: Version 2208
    - Office LTSC 2021 Volume Licensed / 2019 Volume

Summary and Future Work

Summary and Future Work

- We have researched a few patterns among historical vulnerabilities.
  - Still discovered many low hanging fruit vulnerabilities, and some of them were exploitable.
  - There are many memory corruption vulnerabilities
    - challenging to exploit, but not impossible.
- Microsoft has implemented Defense-in-Depth improvement for OLE, although not for all versions.
- However the attack surface for OLE is still present, so OLE object are still dangerous today. It still requires continuous attention from security researchers.

https://bsnet.github.io/ets

Summary and Future Work

- Overview the current landscape, there are still some attacker surfaces that we need to conduct more in-depth research on.
- Office is complex, with numerous features. Deeper logic bugs & other attack surfaces certainly exist and require the collective attention of security researchers in the future.

RTF Control Word Parsing Problem	Open XML Tag Parsing Problem	Embedded OLE object Parsing Problem	Office Embedded Flash 0day
Hard to exploit	Hard to exploit	still dangerous	End of life
EPS File Parsing	Moniker	Embedded IE 0day	Other Logic bug
End of life	Future Work	End of life	Future Work

THANK YOU

Acknowledgement

- We’d like to especially thank Quan Jin, who helped peer-review the presentation
- Special thanks to Haifei Li and Will Dormann for their sharing about Office security

Reference

[1] Microsoft Threat Intelligence. “Analyzing attacks that exploit the CVE-2021-40444 MSHTML vulnerability”. [Online]
https://www.microsoft.com/en-us/security/blog/2021/09/15/analyzing-attacks-that-exploit-the-mshtml-cve-2021-40444-vulnerability/

[2] Qi Li, Quan Jin. ”Needle in A Haystack: Catch Multiple Zero-days Using Sandbox“. [Online]
https://images.seebug.org/archive/Catch_Multiple_Zero-Days_Using_Sandbox-EN.pdf

[3] Haifei Li, Bing Sun. “Attacking Interoperability: An OLE Edition”. [Online]
https://www.blackhat.com/docs/us-15/materials/us-15-Li-Attacking-Interoperability-An-OLE-Edition.pdf

[4] Haifei Li, Bing Sun. “Moniker Magic: Running Scripts Directly in Microsoft Office”. [Online]
https://leo00000.github.io/pdf/Moniker_Magic_final.pdf

[5] Haifei Li. “Analysis of the Attack Surface of Microsoft Office from a User's Perspective”. [Online]
https://sites.google.com/site/zerodayresearch/Analysis_of_the_Attack_Surface_of_Microsoft_Office_from_User_Perspective_final.pdf

[6] Tavis Ormandy. “Issue 514: Microsoft Office / COM Object DLL Planting with els.dll”. [Online]
https://bugs.chromium.org/p/project-zero/issues/detail?id=514&q=label%3AFinder-scvitti&can=1

[7] Will Dormann. “Attacking COM via Word RTF”. [Online]
https://insights.sei.cmu.edu/library/attacking-com-via-word-rtf/