Analyzing Malicious Documents - v1.0

Office Files Versions

Office 97-2003 documents use OLE (object linking and embedding) compound binary file format.

These files have a signature of 0xD0CF11E0A1B11AE1, which appears as "DOCFILE".

These files are still compatible with the latest versions of Office.

OLE file format was replaced in Office 2007 with Office Open XML (OOXML).

OOXML is a ZIP archive that uses XML-based files to represent document structure.

Macros are stored inside OLE2 files and in *vbaProject.bin* in OOXML.

OneNote files can not contain macros without 3rd party extensions. Often contain embedded objects, such as scripts and executables.

PDF files can contain JavaScript to execute code.

RTF documents do not contain macros but can be opened by Word and may be used in exploitation.

Common File Types & Extensions

Word 97-2003	.doc, .dot
Word (OOXML)	.docm, .docx, .dotm, .dotx, .rtf
Excel 97-2003	.xls, .xlt, .xlm
Excel (OOXML)	.xlsx, .xlsm, .xltx, .xltm
PowerPoint 97-2003	.ppt, .pot, .pps, .ppa
PowerPoint (OOXML)	.pptx, .pptm, .potx, .potm, .ppam, .ppsx, .ppsm, .sldx, .sldm, .ppam
OneNote	.one
Publisher	.pub
Portable Document Format (PDF)	.pdf

oledump Use Cases

Python tool designed by Didier Stevens to analyze OLE (Compound Binary Files). Primary file types are Microsoft Office.

Basic usage:

oledump.py [options] file

First command to run on new sample:

oledump maldoc.docm

```
A: word\vbaProject.bin
 A1:         487     'PROJECT'
 A2:          71     'PROJECTwm'
 A3:  M     1053     'VBA/NewMacros'
 A4:  M     6073     'VBA/ThisDocument'
 A5:        3667     'VBA/_VBA_PROJECT'
 A6:        1089     'VBA/__SRP_0'
 A7:          70     'VBA/__SRP_1'
 A8:          84     'VBA/__SRP_2'
 A9:         103     'VBA/__SRP_3'
A10:         575     'VBA/dir'
```

vbaProject.bin indicates OOXLM document with macro code.

First column *A#* indicates the stream number, this value can vary in format.

Second column indicates stream type. Uppercase *M* indicates a macro stream with macro content. These are typically the most interesting.

m (lowercase) indicates a macro stream without macro content. This are often found with documents that contain forms.

Other indicator types include:

- E: corrupt
- !: unusual VBA code
- O: embedded object
- .: storage
- R: root entry

Third column represents stream size.

Fourth column represents stream name.

To inspect macros, use the *-s* argument to identify stream number and *-v* to decompress macro stream:

oledump -s A3 -v maldoc.docm

This will dump macro content to the terminal, consider redirecting with > to a file for further inspection.

Use *a* as the stream index to dump all macro streams.

Use Yara rules to scan a document:

oledump -y yara.rule maldoc.docm

Output will show if Yara rule matched the VBA code.

To use a plugin, use the *-p* argument.

Oledump -p plugin_vba_summary maldoc.docm

Josh Stroschein | The Cyber Yeti @jstrosch

Analyzing Malicious Documents - v1.0

olevba Use Cases

Script to parse OLE and OOXML files to extract and analyze VBA code. Also supports XLM/Excel 4 macros.

olevba [options] file file2 …

First command to run on a sample:

olevba file

Output will include macro code and a summary table w/ helpful indicators.

Summary table will include *Type, Keyword* and *Description*.

Indicators will be highlighted in macro code above summary table; this helps identify suspicious code segments.

If obfuscated strings are suspected, try *--decode*. For obsucated code, try *--reveal*.

Excel 4.0 / XLM

Introduced in 1992 by Microsoft, were disabled by default in 2022 after an uptick in abuse.

Macros are embedded in an Excel sheet, do not use VBA. Require different tools to analyze.

oledump provides the plugin *plugin_biff* for analyzing XLM macros.

XlmMacroDeobfuscator is another purpose-built tool and includes macro emulation.

oledump PLUGIN_BIFF Use Cases

To view sheet visibility and name:

oledump --p plugin_biff -pluginoptions "-x" maldoc.xls

XLMMacroDeobfuscator Use Cases

First command to run on a sample:

Xlmdeobfuscator -f maldoc.xls

Output includes entry point worksheet/cell and emulated code.

onedump Use Cases

Dump tool for OneNote files.

onedump [options] file

First command to run on a sample:

onedump malware.one

```
le: malware.one
 1: 0x00002530 powe …
 2: 0x00003490 .PNG …
 3: 0x00003840 .PNG …
 4: 0x0000f148 .Set …
```

The first column represents the index. The second column the beginning of the stream content. This is often the most interesting column. The remaining columns provide stream information.

In this example, streams 1 and 4 are the most interesting. Stream 1 appears to be PowerShell while stream 4 a batch script.

To extract content from a stream, use the *-d* argument w/ the stream index.

oledump -s 1 -d malware.one

PDFID & PDF-Parser Use Cases

PDFID is a tool to test a PDF file.

PDFParser is a tool to parse PDF files.

First command to run:

pdfid.py file

Look for evidence of JavaScript in /JS or /JavaScript. Then use *pdf-parser.py*.

Inspect objects:

pdf-parser.py file

Index for object will be displayed, can use with *-o* argument. Look for command execution and other suspicious behavior.

Suspicious PDF Strings

/JavasScript, /JS, /AcroForm, /XFA	JavaScript code
/OpenAction, /AA	Auto-execution
/URI	Resource access by URI
/ObjStm	Object stream

oledump Commands

Python tool designed by Didier Stevens to analyze OLE (Compound Binary Files).

oledump.py [options] file

-m, -h	Usage information
--version	Version number
-m	Print manual
-s [NUMBER]	Select item NUMBER for dumping
-d	Perform dump
-x	Performs hex dump
-a	Performs ASCII dump
-A	ASCII dump w/ RLE
-S	String dump
-T	Head & Tail
-v	VBA decompression
-r	Read raw file (use with -v or -p)
-t ENCODING	String translation, ENCODING is utf16, etc
-e	Extract OLE embedded files
-i	Print extra info
-p PLUGIN	Loads PLUGIN
--pluginoptions	Options for the loaded plugin
-q	Only print output from plugins
-y YARA	YARA rule file or directory
-D DECODER	Decoders to load
-M	Print metadata
-V	Verbose output
-C CUT	Cut data
-c	Extra data such as hashes
--password=	Password to use on input file
-j	JSON output

Josh Stroschein | The Cyber Yeti @jstrosch

Analyzing Malicious Documents - v1.0

olevba Commands

Script to parse OLE and OOXML files to extract and analyze VBA code. Also supports XLM/Excel 4 macros.

olevba [options] file file2 …

-h	Usage information
-r	Finds files recursively
-z	ZIP password
-a	Display only analysis, not source code
-c	Display on VBA source code
--decode	Attempt to deobfuscate encoded strings
--reveal	Show VBA source code with deobfuscated string content
--deobf	Attempt to deobfuscate VBA expressions
--show-pcode	Show disassembled P-code
-t	Triage mode, display summary table only
-d	Detailed report

XLMMacroDeobfuscator Commands

Xlmdeobfuscator -f FILE [options]

-h	Help
-f	File path
-x	Extract cells only without emulation
--start-point	Start interpretation from a specific cell
-p	Password for file
--timeout N	Emulation timeout in seconds

PDFID Commands

PDFID is a tool to test a PDF file.

pdfid.py [options] file

--version	Version number
-h	Help
-s	Scan given directory
-a	Display all names
-f	Force scan
-d	Disable JavaScript and auto launch
-p PLUGIN	Load PLUGIN
-v	Verbose mode
-S SELECT	Select expression

PDF-Parser Commands

pdf-parser.py [options] file

-s	String to search
-r REFERENCE	ID of indirect object being referenced
-e ELEMENTS	Type of elements to select
-o OBJECT	ID(s) of indirect object being referenced

onedump Commands

-h	Help
-o	Output to file
-s #	Select item by #
-d	Dump
-x	Hexdump

Misc. Tools & Purpose

Remove password from protected VBA project:

evilclippy -uu file

Extract objects embedded in an RTF file:

rtfobj.py malware.rtf

Inspect contents of OOXML file (ZIP):

zipdump.py file

Extract stream with index from file:

zipdump.py -s 2 -d file

Pretty-print XML from STDIN, consider using | redirection:

xmldump.py pretty

ViperMonkey is a VBA emulation tool.

vmonkey file

msoffcrpyto-tool is used to decrypt/encrypt office files. Will prompt for password, otherwise define password with *-p* argument.

msofficecrpto-tool encrypted decrypted

Josh Stroschein | The Cyber Yeti @jstrosch