Files and Filetypes

REVERSE ENGINEERING

João Paulo Barraca

deti universidade de aveiro
departamento de eletrónica, telecomunicações e informática

Files

- Files are containers that are parsed according to a schema
  - Parsing implies knowing the file content
- How to select the adequate parser?
  - Using the file extension
  - Using magic headers
  - Using rules provided by configuration
  - Previous knowledge
- What if the parser is wrong?

File extensions

- File extensions are words appended to the filename, after a dot

`lecture.pptx`

- File extensions are a basic mechanism to know how to handle a file
  - Operating systems uses extensions to select the correct process
  - Applications use it to filter which files are adequate (.e.g images). Mostly an usability aspect
  - Humans use extensions to differentiate files

- Popular file extensions:
  - zip, rar, bz2, gz, 7z: compressed files
  - exe, dll, so, com: executable files
  - jpg, tiff, bmp, fits, png: images

File extensions

- Knowing the file extension is important to apply the correct analysis process
  - Analyzing a JPG is different from analyzing an EXE, or even a PNG

File extensions

Extensions are misleading!

- Windows hides extension of known file types
  - **`Sample.pptx`** becomes only **`Sample`**

- Executable files may have an embedded icon
  - Freely defined by the developer
  - Explorer will show that icon

- A file named **`Sample.pptx.exe`** will be shown as **`Sample.pptx`**
  - Users recognize the extension and may think the file is safe

- In a RE task, consider that a file may have bogus extensions

File Signature

Also known as Magic Bytes/Header

- Most files can also be recognized by a magic value in the file start/end
  - Manipulating headers can lead to incorrect detection and maybe processing
  - Some OS use the magic headers instead of the file extension
  - Also known as File Signatures

- Some magic values:
  - Office Documents: `D0 CF 11 E0`
  - ELF: `7F E L F`
  - JPG: `FF D8`
  - PNG: `89 P N G 0D 0A 1A 0A`
  - Java class: `CA FE BA BE`

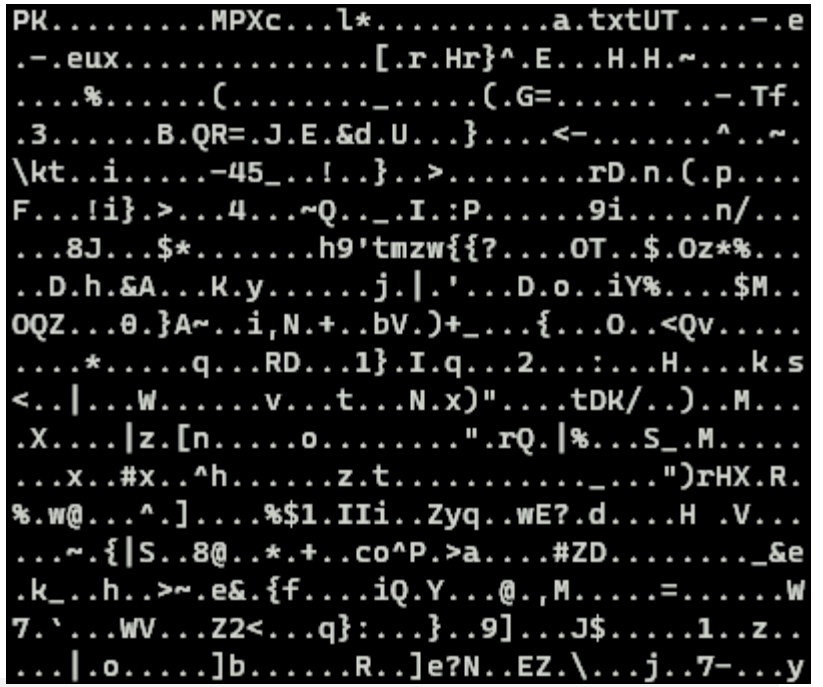
File Signature

Sometimes, magic headers are reused

- PK.. (`50 4B 03 04`) is the magic for ZIP files

```
└─$ file 8\ -\ Obfuscation.pptx
8 - Obfuscation.pptx: Microsoft PowerPoint 2007+
```


```
└─$ file sample.zip
sample.zip: Zip archive data, at least v2.0 to extract
```



File Signature

Sometimes, magic headers are reused

- Actually, pptx are zip files

```
 $ unzip -l 8\ -\ Obfuscation.pptx
Archive:  8 - Obfuscation.pptx
  Length      Date    Time    Name
---------  ---------- -----   ----
     5179  1980-01-01 00:00   ppt/presentation.xml
    12041  1980-01-01 00:00   customXml/item1.xml
     1203  1980-01-01 00:00   customXml/itemProps1.xml
      219  1980-01-01 00:00   customXml/item2.xml
      335  1980-01-01 00:00   customXml/itemProps2.xml
      394  1980-01-01 00:00   customXml/item3.xml
      606  1980-01-01 00:00   customXml/itemProps3.xml
    33895  1980-01-01 00:00   ppt/slideMasters/slideMaster1.xml
     2477  1980-01-01 00:00   ppt/slides/slide1.xml
     4665  1980-01-01 00:00   ppt/slides/slide2.xml
     4384  1980-01-01 00:00   ppt/slides/slide3.xml
     4003  1980-01-01 00:00   ppt/slides/slide4.xml
     4719  1980-01-01 00:00   ppt/slides/slide5.xml
```

```
PK.........!.x.......;.......ppt/presentati
on.xml...n.8.....;..-............@P...Jt"T'.
t.t...$-.CS(z.w..........x.....W>..k.>..|..E
WV.......2....%.../.w..O......~....I5.SaZ.^K
.E.~...x...7].[....aR....r^?T...q.%a.......3
......w..Q.....6>.K..)Z.;.^..%.^...Mp..Z)...
...u.7......B^...r.cDS...*v.B.Q....m87..$..z
w...1..........[.kr4?O........)..l...\..! ..
....#'pv..).Q....r..pu..=..n...C{...u....R.u
..N0.]z....>k..~..x....]~i.u._..a._..a._....
......,........ ?...a~.....G\~..~d..5.f...Kf
.Y../....R....r..../....?....r.8u......?.A..
.G"k}..AV|... ...~.....G"..:.H...u...:..~$..
+........^.:....o..b..R....K?..L.1.Mi..M...#
JO.J.g.Z>.7...5_.2...q...<.^.t........C....j
```

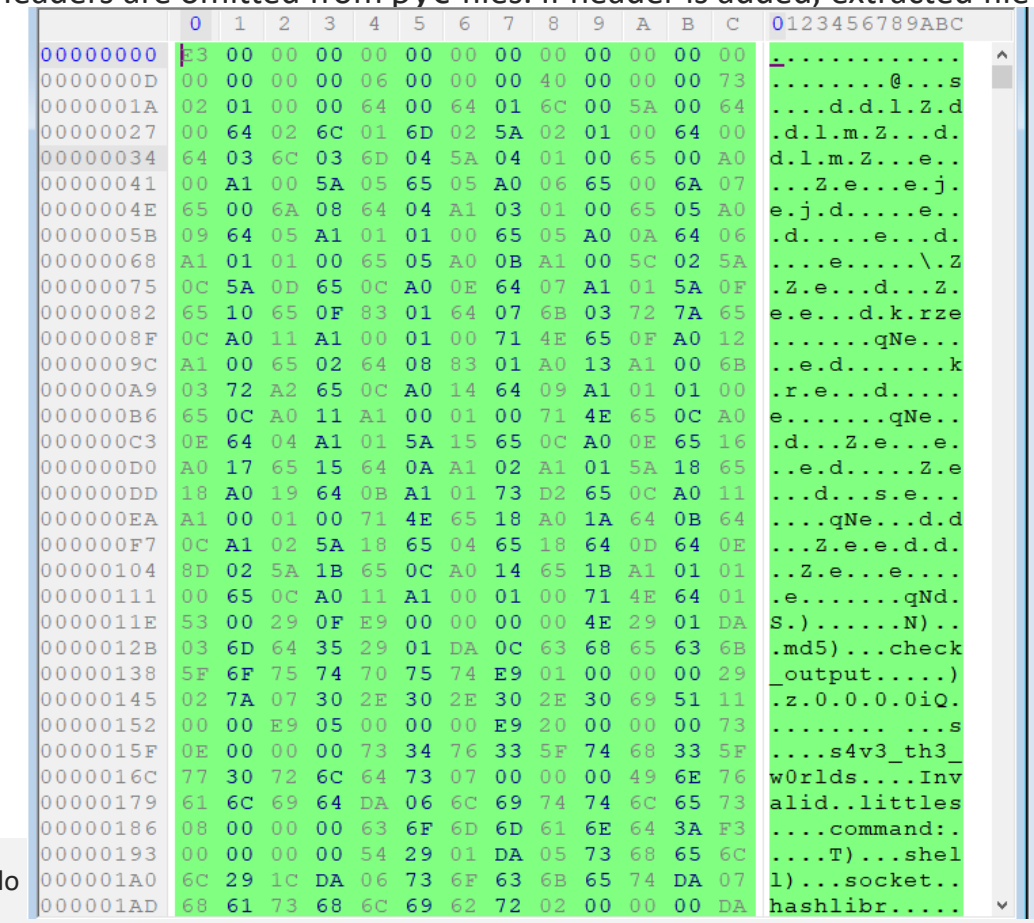
File Signature

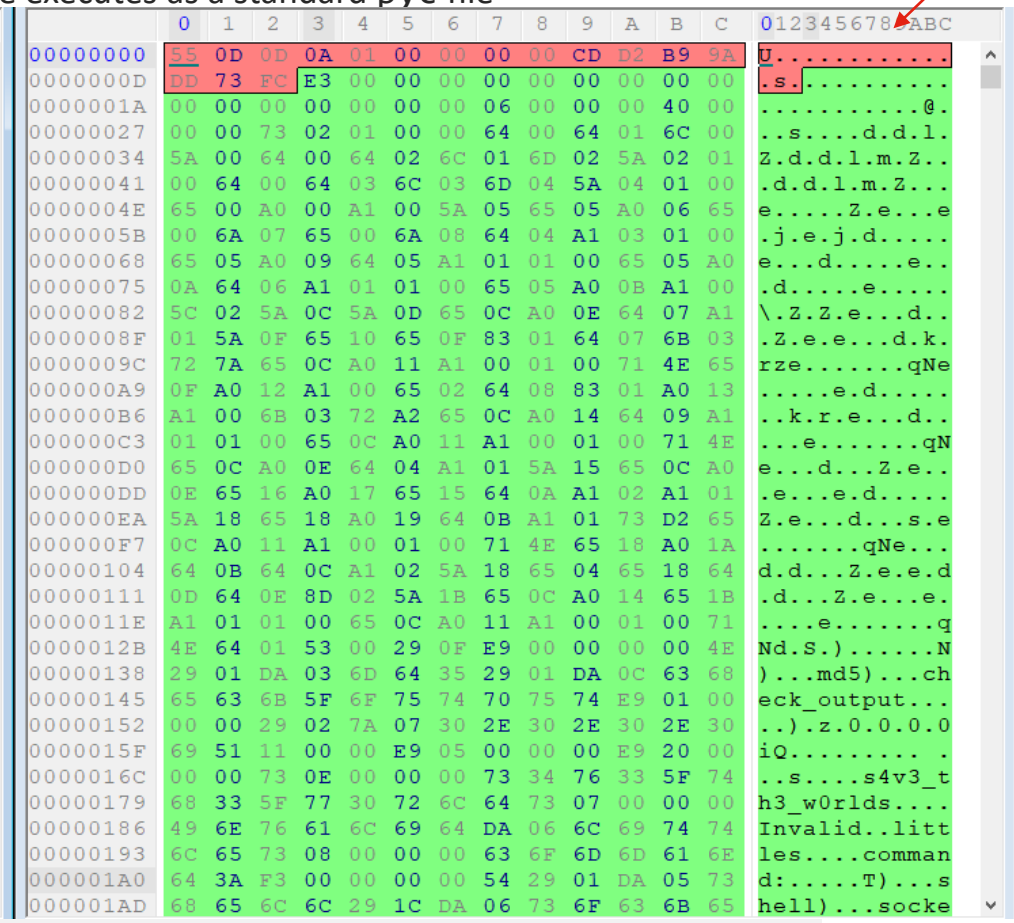
Magic Headers can be manipulated if the content is known

- `PyInstaller` allows conversion of a Python application to an executable application
  - It packs the `pyc` files into a container. Container is extracted on runtime and compiled python code is executed
  - Headers are omitted from pyc files. If header is added, extracted file executes as a standard pyc file

Added header

Extracted



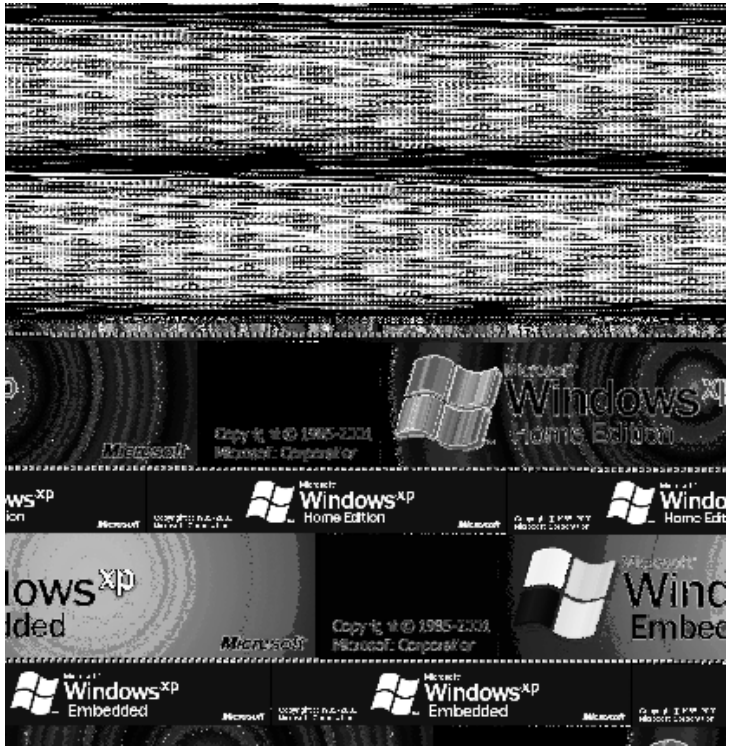


Reconstructed

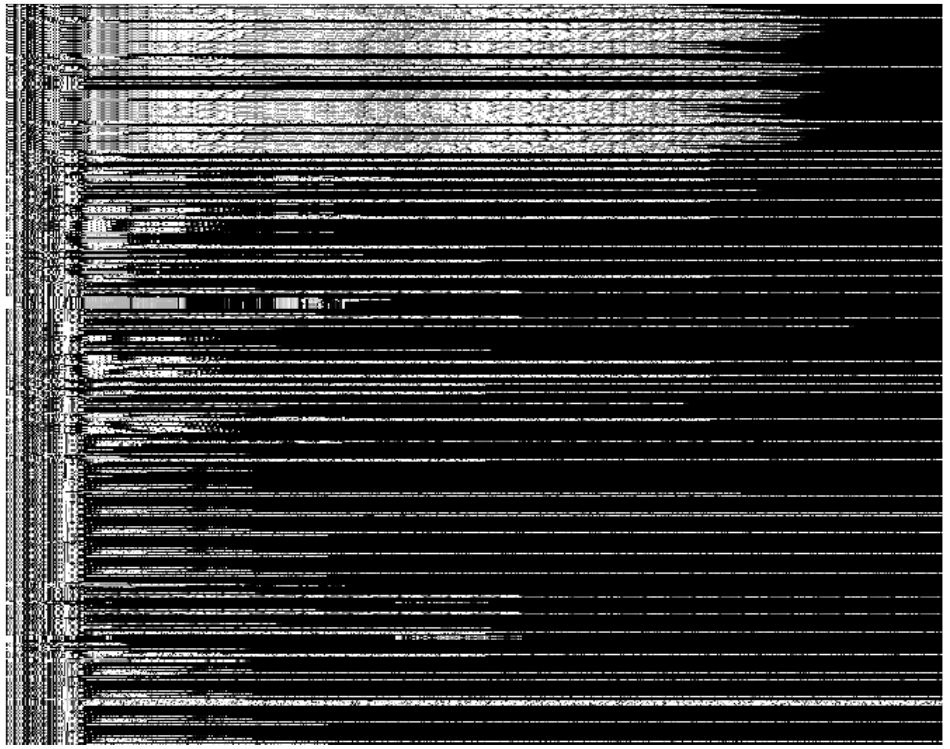
File Signature

Magic Headers can be manipulated if the content is known

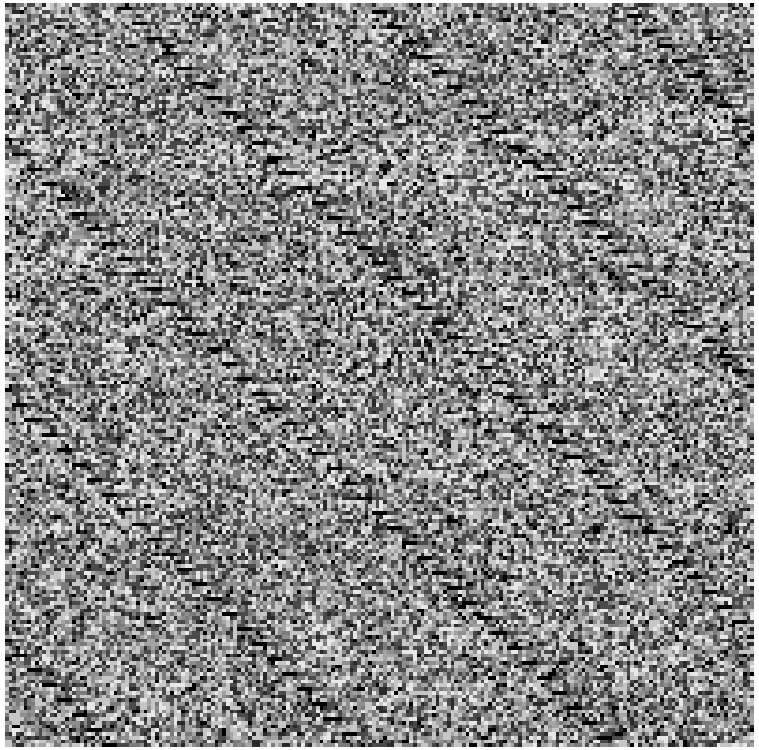
- Direct Visualization may help
  - Direct byte visualization, Mapping to an image, Entropy Analysis, Tuples



shell32.dll



Network traffic



Compressed data

Greg Conti, Sergei Bratus, “Voyage of the Reverser A Visual Study of Binary Species”

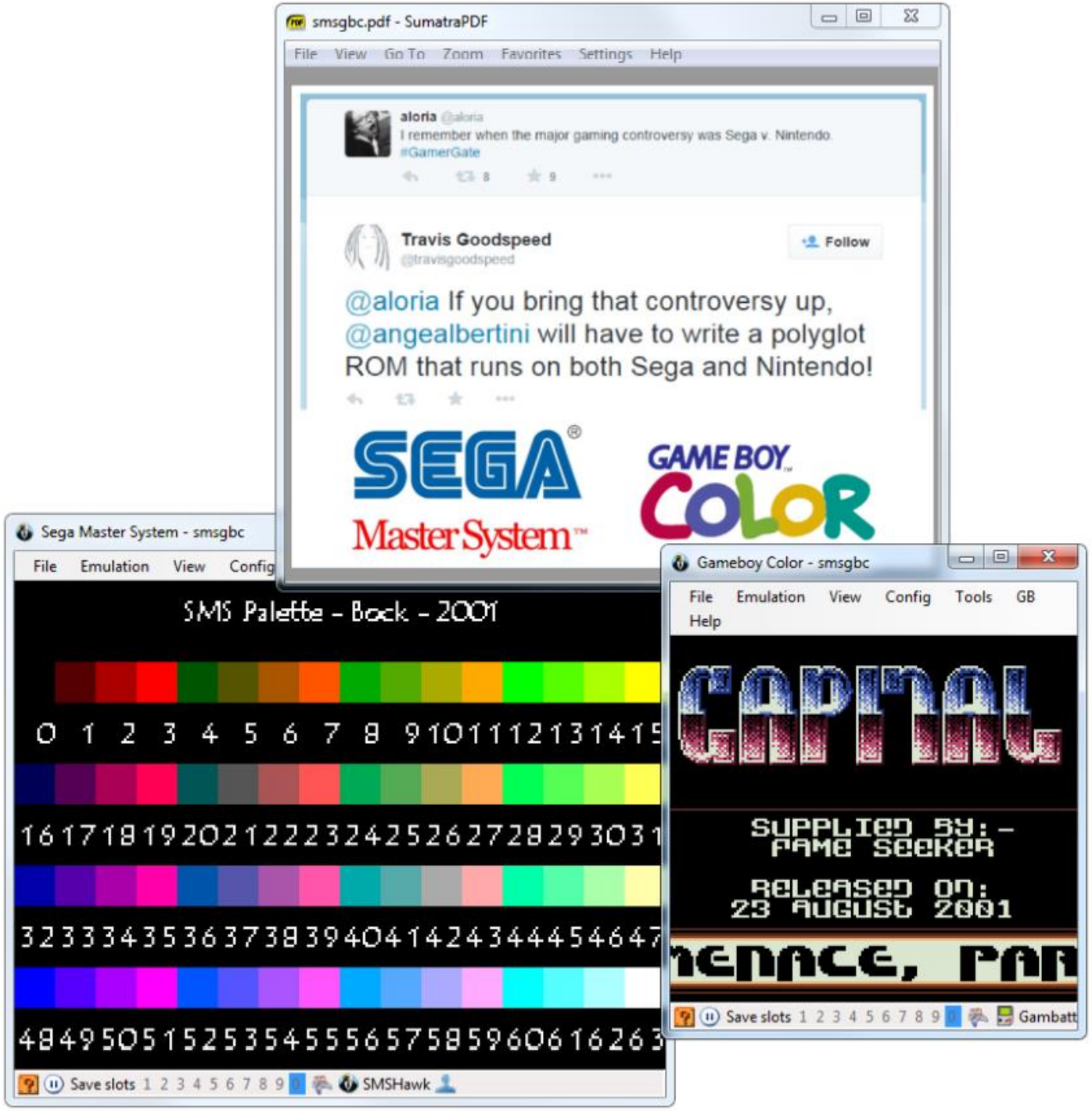
Content Type Obfuscation

Polyglots

A file that has different types simultaneously, which may bypass filters and avoid security counter-measures.

pocorgtfo19.pdf (alchemistowl.org)

Technical Note: This file, pocorgtfo19.pdf, ***is valid as a PDF*** *document,* ***a ZIP archive****, and* ***a HTML page****. It is also available as a* ***Windows PE executable****, a* ***PNG image*** *and an* ***MP4 video****, all of which have the same MD5 as this PDF*


smsgbc.pdf - SumatraPDF
File View Go To Zoom Favorites Settings Help
aloria @aloria
I remember when the major gaming controversy was Sega v. Nintendo.
#GamerGate
Travis Goodspeed
@travisgoodspeed
Follow
@aloria If you bring that controversy up, @angealbertini will have to write a polyglot ROM that runs on both Sega and Nintendo!
SEGA
Master System
GAME BOY COLOR
Sega Master System - smsgbc
File Emulation View Config
SMS Palette - Back - 2001
Save slots 1 2 3 4 5 6 7 8 9
SMSHawk
Gameboy Color - smsgbc
File Emulation View Config Tools GB Help
CAPTAIN
SUPPLIED BY:-
PAME SEEKER
RELEASED ON:
23 AUGUST 2001
Save slots 1 2 3 4 5 6 7 8 9
Gambatt

Content Type Obfuscation - Polyglots

Types

- **Simple Polyglot file**: file **has** different types, accessed depending on how it is handled

- **Ambiguous file**: is one that **is interpreted** differently depending on the parser. One parser may crash or fail to process it, while other may return a valid file.

- **Chimera file**: file has some data that is interpreted as different types

Content Type Obfuscation - Polyglots

Use in Malware

https://nvd.nist.gov/vuln/detail/CVE-2009-1862

…allows remote attackers to **execute arbitrary code** or cause a **denial of service** (memory corruption) via (1) a **crafted Flash application in a .pdf file** or (2) **a crafted .swf file**, related to authplay.dll, as exploited in the wild in July 2009.

Content Type Obfuscation - Polyglots

Strategies

- Stacks: Data is appended to the file
- Cavities: Uses blank (non used space) in the file
- Parasites: Uses comments or metadata fields that allow content to be written
- Zippers: mutual comments

Content Type Obfuscation - Polyglots

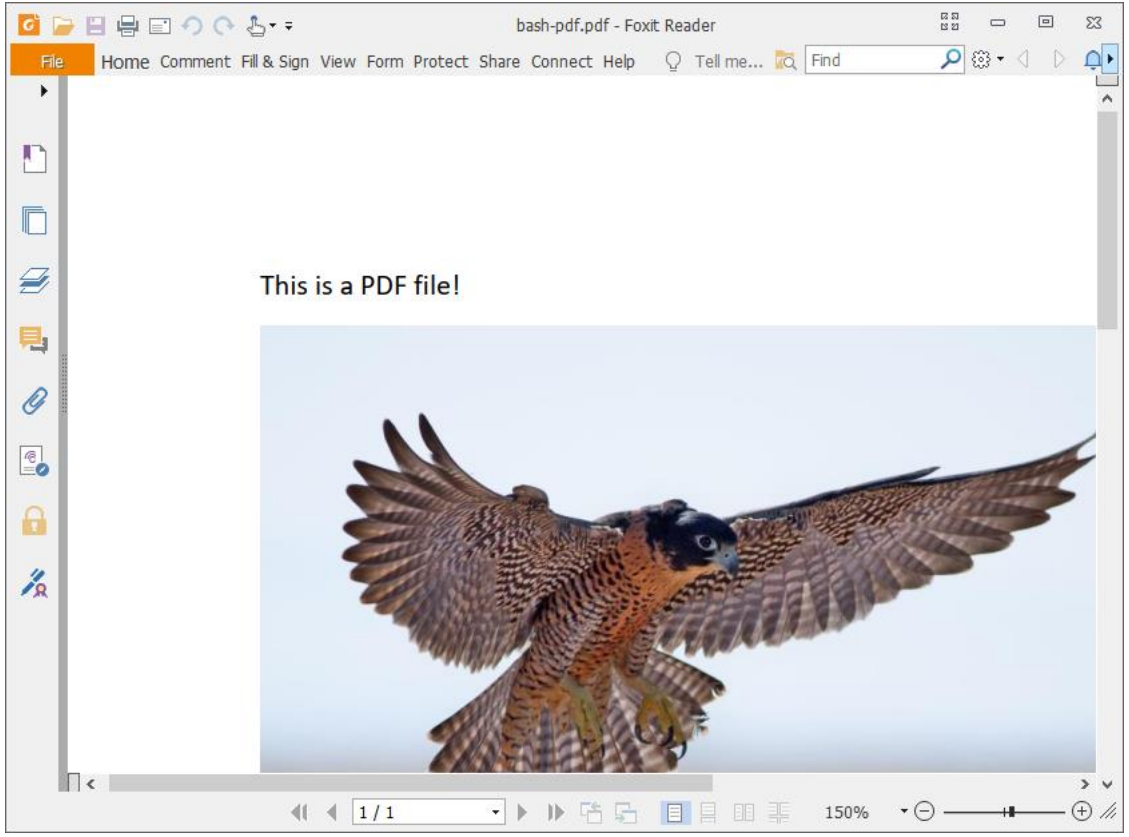
Empty Space

- Files sometimes allow empty or unused space
  - Before, in the middle or after actual content (appended)
  - Most common in Block formats (ISO and ROM dumps, TAR archives)
    - NAND dumps, ROM dumps, ISOs are directly mapped to sectors
  - Some formats allow arbitrary bytes before file start (e.g. PDF)
    - PDFs are processed from the end
- “Empty space” can be abused to inject crafted content

A simple bash-pdf polyglot

bash-pdf.pdf

```
$ file bash-pdf.pdf
bash-pdf.pdf: POSIX shell script executable (binary data)
$ ./bash-pdf.pdf
Hello World
```




```
1   #!/bin/bash
2   echo "Hello World"; exit
3   %PDF-1.7
4   %µµµµ
5   1 0 obj
6   <</Type/Catalog/Pages 2 0 R/Lang(en-US) /StructTreeRoot 11 0 R/MarkInfo<</Marked true>>/Metadata 23 0 R/ViewerPreferences 24 0 R>>
7   endobj
8   2 0 obj
9   <</Type/Pages/Count 1/Kids[ 3 0 R] >>
10  endobj
11  3 0 obj
12  <</Type/Page/Parent 2 0 R/Resources<</Font<</F1 5 0 R>>/ExtGState<</GS7 7 0 R/GS8 8 0 R>>/XObject<</Image9 9 0 R>>/ProcSet[/PDF/Text/ImageB/ImageC/ImageI]
    >>/MediaBox[ 0 0 612 792] /Contents 4 0 R/Group<</Type/Group/S/Transparency/CS/DeviceRGB>>/Tabs/S/StructParents 0>>
13  endobj
14  4 0 obj
15  <</Filter/FlateDecode/Length 245>>
16  stream
```

A simple bash-pdf polyglot

Why?

- PDF is a collection of objects
  - Objects are dictionaries of properties with a named type
  - Called “CosObjects” or Carousel Object System
  - Simply added to file. New revisions will create new objects that are appended
  - A PDF can have unused object
  - Objects can contain executable code (the code is not executed by the pdf reader!)
    - Objects can contain anything!
    - Well…. There is the LAUNCH action, and Javascript is a valid object type…

A simple bash-pdf polyglot

A simple object

```
1 0 obj
<</length 100>>
stream

…100 bytes..



endstream
endobj
```

A simple bash-pdf polyglot

Two objects

```
1 0 obj
<</length 100>>
stream
…100 bytes..
endstream
Endobj
2 0 obj
<</length 100>>
stream
…100 bytes..
endstream
endobj
```

A simple bash-pdf polyglot

Two objects and something else that is not parsed

```
1 0 obj
<</length 100>>
stream
…100 bytes..
endstream
Endobj
I should not be here, but who cares. And I could be anywhere
2 0 obj
<</length 100>>
stream
…100 bytes..
endstream
endobj
```

A simple bash-pdf polyglot

The XREF Table

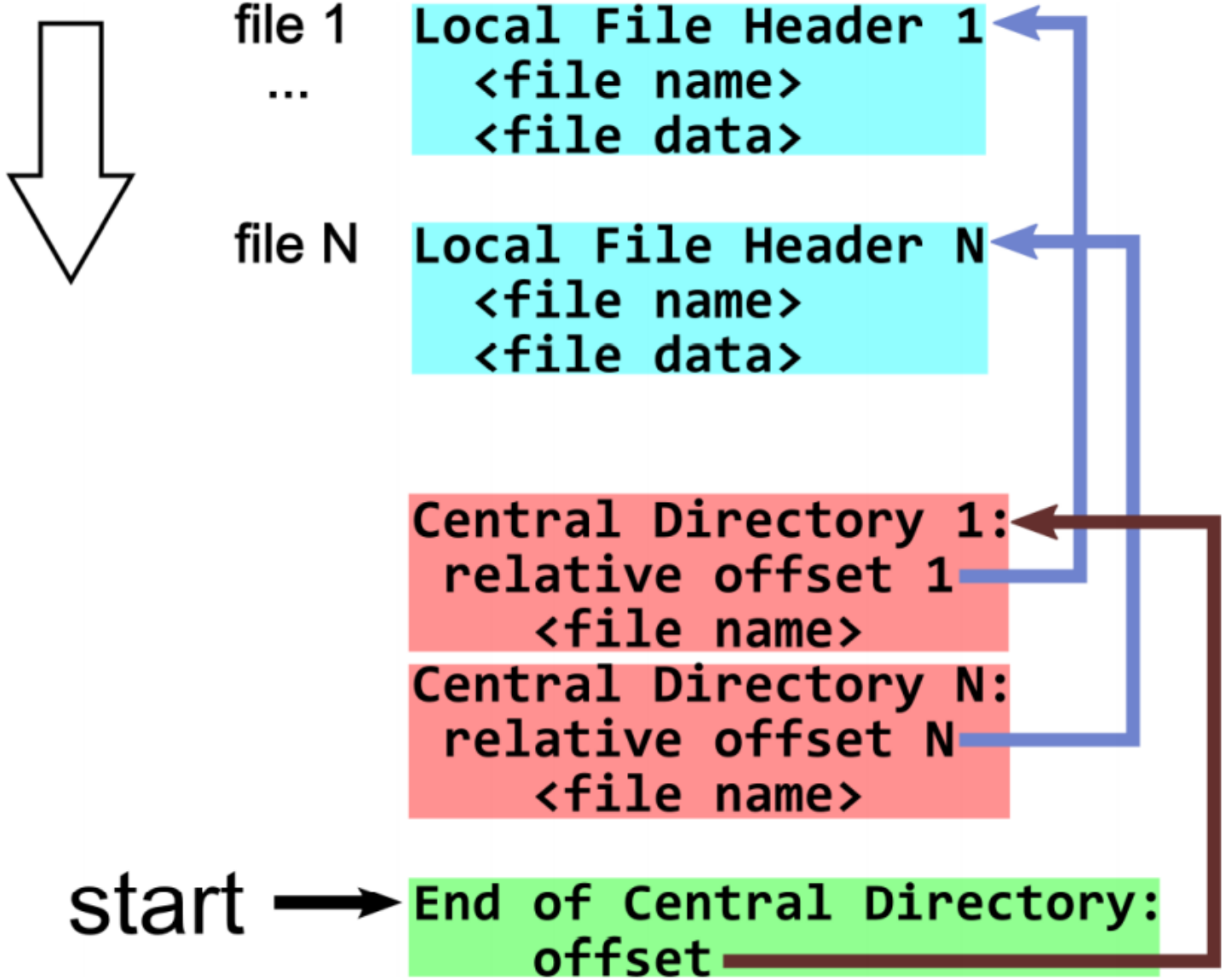
- PDF have a table with the offset of every object
  - In the end!
  - Reader skips to the end of the file, reads the table and parses the objects
    - That's one reason why it ignores garbage between objects

- XREF table also defines where the file magic (`%PDF-1.5\n\n`) is
  - There may be some bytes before the magic
  - Actually, 1024 random bytes are allowed

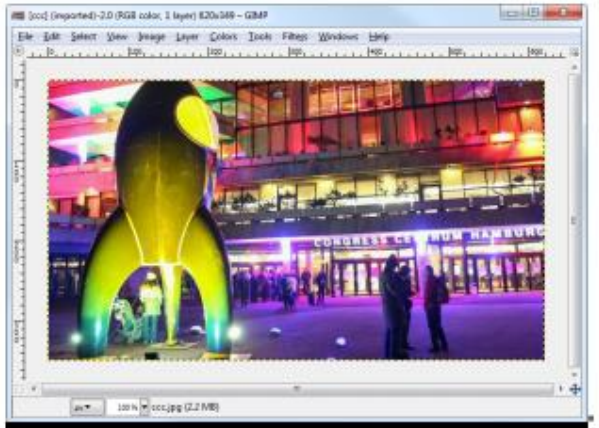
```
1   xref
2   0 26
3   0000000011 65535 f
4   0000000017 00000 n
5   0000000166 00000 n
6   0000000222 00000 n
7   0000000511 00000 n
8   0000000830 00000 n
9   0000000998 00000 n
10  0000001237 00000 n
11  0000001290 00000 n
12  0000001343 00000 n
13  0000055720 00000 n
14  0000000012 65535 f
15  0000000013 65535 f
16  0000000014 65535 f
17  0000000015 65535 f
18  0000000016 65535 f
19  0000000017 65535 f
20  0000000018 65535 f
21  0000000019 65535 f
22  0000000020 65535 f
23  0000000000 65535 f
24  0000056466 00000 n
25  0000056683 00000 n
26  0000083140 00000 n
27  0000086318 00000 n
28  0000086363 00000 n
29  trailer
30  <</Size 26/Root 1 0 R/Info 10 0 R/ID[<85F88F67066D2E4AAB78E636585E887B><85F88F67066D2E4AAB78E636585E887B>] >>
31  startxref
32  86664
33  %%EOF
34  xref
35  0 0
36  trailer
37  <</Size 26/Root 1 0 R/Info 10 0 R/ID[<85F88F67066D2E4AAB78E636585E887B><85F88F67066D2E4AAB78E636585E887B>] /Prev 86664/XRefStm 86363>>
38  startxref
39  87341
40  %%EOF
```

Offsets of object locations

ZIP


file 1
...
Local File Header 1
<file name>
<file data>
file N
Local File Header N
<file name>
<file data>
Central Directory 1:
relative offset 1
<file name>
Central Directory N:
relative offset N
<file name>
start
End of Central Directory:
offset

JPG

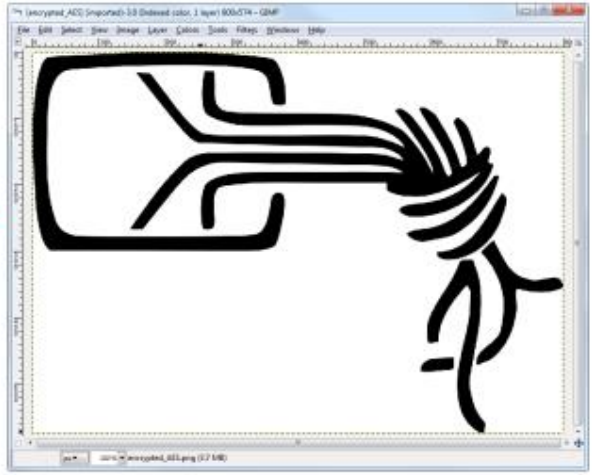


JAR
(ZIP + CLASS)

```
>java -jar ccc.jpg
Hello World! [Java]

>
```

AES_{K_1}



PNG

AES_{K_2}

FLV

3DES

PDF

Content Type Obfuscation - Polyglots

Practical application

- Malware makes use of polyglots are means to circumvent filters
  - A Packet/Email/Web application firewall will block executables, but will it block JPGs?
    - If it does, can it be done with a low rate of false positives?

- General process involves download a polyglot and a decoder
  - Polyglot contains malicious code
  - Decode is implemented in a less suspicious manner (e.g., Javascript)

- From a Reversing Perspective: how much effort will we spend analyzing a JPG?
  - Automated tools such as `binwalk`, `TrId` and file can help (but are limited)