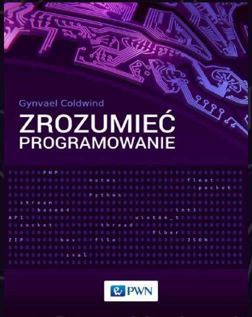
Ten thousand security pitfalls:
The ZIP file format.

Gynvael Coldwind
@ Technische Hochschule Ingolstadt, 2018

About your presenter



Gynvael Coldwind

ZROZUMIEĆ PROGRAMOWANIE

PWN

Redakcja naukowa
Mateusz Jurczyk i Gynvael Coldwind

PRAKTYCZNA INŻYNIERIA WSTECZNA

METODY, TECHNIKI I NARZĘDZIA

PWN

Tomasz Bukowski • Grzegorz Antoniak • Tomasz Kwiecień • Mateusz Krzywicki • Marcin Hartung
Gynvael Coldwind • Hasherezade • Maciej Kotowicz • Michał Kowalczyk
Robert Święcki • Piotr Bania • Mateusz Jurczyk

DRAGON SECTOR

(among other things)

All opinions expressed during this presentation are mine and mine alone, and not those of my barber, my accountant or my employer.

What's on the menu

Also featuring:
Steganography

1. What's ZIP used for again?
2. What can be stored in a ZIP?
   a. Also, file names
3. ZIP format 101 and format repair
4. Legacy ZIP encryption
5. ZIP format and multiple personalities
6. ZIP encryption and CRC32
7. Miscellaneous, i.e. all the things not mentioned so far.

Or actually, hacking a "secure cloud disk" website.

EDITORIAL NOTE

Everything in this color is a quote from the official ZIP specification by PKWARE Inc.

The specification is commonly known as **APPNOTE.TXT**

https://pkware.cachefly.net/webdocs/casestudies/APPNOTE.TXT

Cyber Secure CloudDisk

Where is ZIP used?

.zip files, obviously

Default ZIP file icon from
Microsoft Windows 10's Explorer

And also...

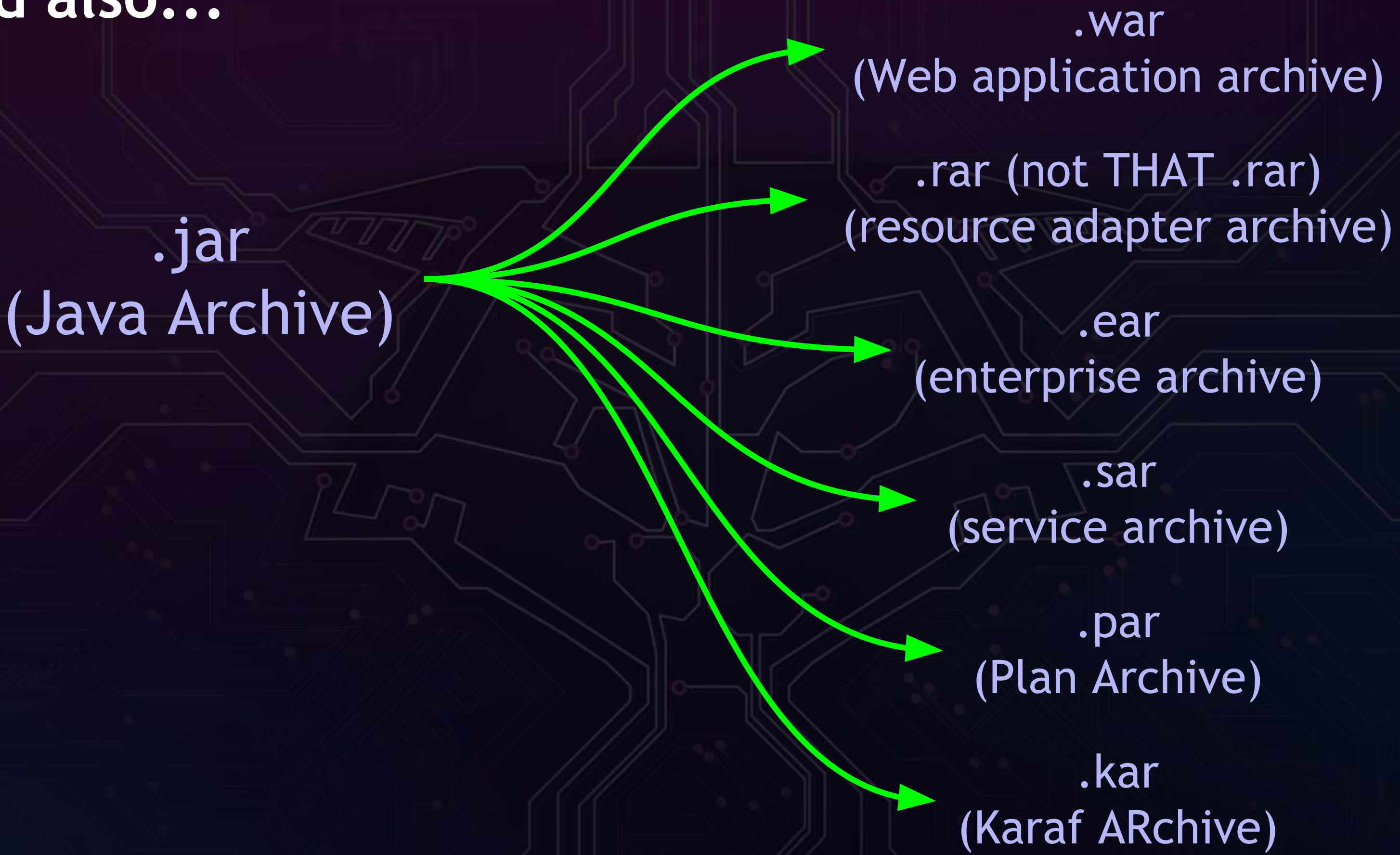
.odt, .odp, .ods, ...
(OpenDocument)

Open Packaging Conventions:
.3mf, .dwfx, .cddx, .familyx, .fdix, .appv, .semblio, .vsix, .vsdx, .appx, .appxbundle, .cspkg, .xps, .nupkg, .oxps, .jtx, .slx, .smpk, .scdoc,

and **Offixe Open XML formats:**
.docx, .pptx, .xlsx

https://en.wikipedia.org/wiki/Open_Packaging_Conventions

And also...



.jar
(Java Archive)

.war
(Web application archive)

.rar (not THAT .rar)
(resource adapter archive)

.ear
(enterprise archive)

.sar
(service archive)

.par
(Plan Archive)

.kar
(Karaf ARchive)

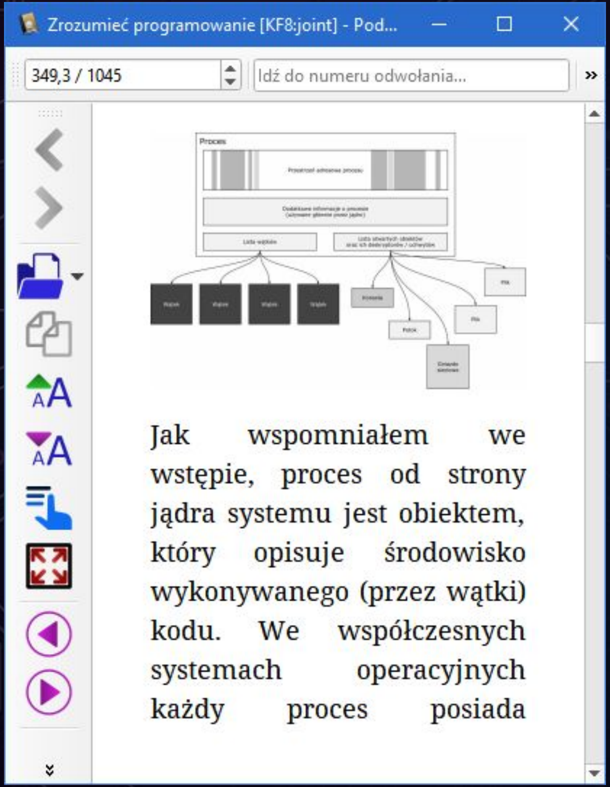
https://en.wikipedia.org/wiki/JAR_(file_format)

And also...

.apk (Android Application Package)

Icon from Android SDK

.epub (Electronic Publication)


Zrozumieć programowanie [KF8:joint] - Pod...
349,3 / 1045
Idź do numeru odwołania...

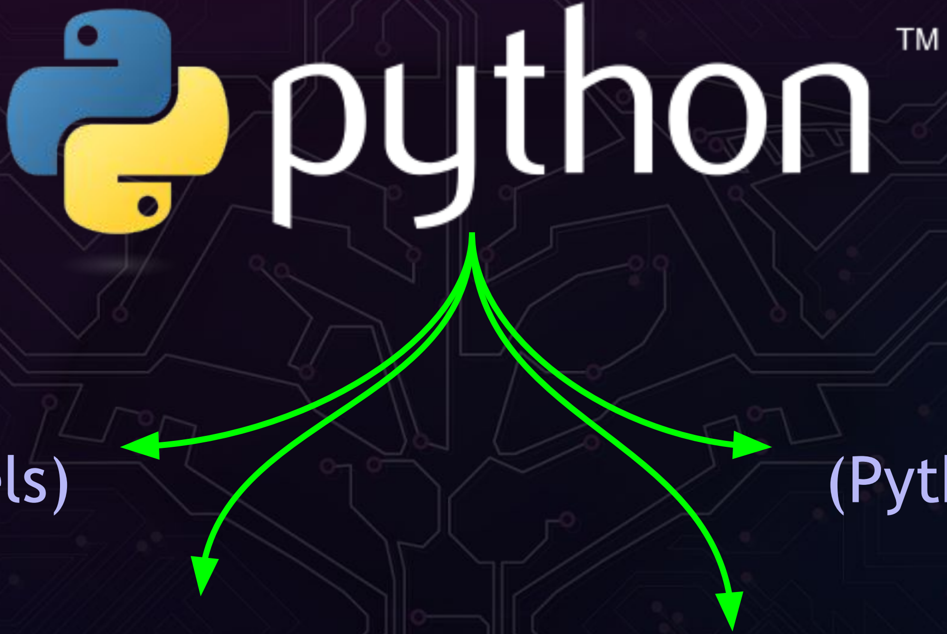
Jak wspomniałem we wstępie, proces od strony jądra systemu jest obiektem, który opisuje środowisko wykonywanego (przez wątki) kodu. We współczesnych systemach operacyjnych każdy proces posiada

calibre

And also… (script to scan drives)

```
import os
import sys

IGNORE_LIST = {
   ".zip", ".docx", ".odt", ".epub", ".jar", ".xlsx",
".pyz",
   ".pptx", ".odp",
}

def process_file(fname):
  try:
    with open(fname, "rb") as f:
      d = f.read(4)
  except WindowsError:
    return False  # No access probably, don't care.
  except IOError:
    return False  # No access probably, don't care.
  if d.startswith("PK\3\4"):
    return True
  return False


def scan_dir(path):
  try:
    entries = os.listdir(path)
  except WindowsError:
    return  # No access probably, don't care.
  for fname in entries:
    name, ext = os.path.splitext(fname)
    if ext.lower() in IGNORE_LIST:
      continue

    if ext == '':
      ext = '_'

    full_path = path + "\\" + fname

    if os.path.isfile(full_path):
      ret = process_file(full_path)

      if not ret:
        continue

      print "%s: %s" % (ext, full_path)

      with open("scan_res/%s" % ext, "a") as f:
        f.write("%s\n" % full_path)
      continue

    if os.path.isdir(full_path):
      scan_dir(full_path)
      continue


if len(sys.argv) != 2:
  sys.exit("usage: scan_disk.py <start_dir>")

scan_dir(sys.argv[1])
```

And also...



.whl
(Python Wheels)

.egg
(Python Egg)

zipimport

.npz
(Python NumPy)

And also...

(avrdbg/bundles/*.bndl)

Anki

(backup-*.apkg)

gog com

(*.hashdb)

(*.btapp, *.lng)

Camtasia®

(.cmmbtn, .cmmtpl, .libzip)

An

(*.fla, *.swc)

And also...

Quake 3 (*.pkg)

02:27.17, 25 March 2010
IP: 127.0.0.1
CPU usage
0..........50..........100
International Stock Market System
Uplink International Bank
Uplink Public Access Server
Uplink Test Machine
International Social Security Database
InterNIC
Gateway
Global Criminal Database
Uplink Internal Services System
International Academic Database
Central Medical Database
Cancel
Connect

Establish the connection to your current remote host

Uplink (*.dat files)

And also...

Torment (*.ttn savegame files)

Pillars of Eternity (*.savegame)

And also...

.aar (Axis Archive / Android Archive Library)
.appx (Microsoft General MIDI DLS)
.bau (OpenOffice's... something?)
.cache (Microsoft extension*.*.cache?)
.dat (Intel VTune Amplifier resources)
.dpk (YAMAHA... something?)
.dsf (DeDe Symbol Files)
.eftx (Microsoft Office Document Themes Effects)
.fcstd (KiCad 3D shapes)
.hdf, .ise (now really sure, sth hardware related)
.htb (wx wxHTML help format)

And also...

.jisp (Psi icon set)
.little (Thunderbird/Firefox startup cache)
.lsz (LiteStep themes/configuration)
.mshc (Microsoft Help Container File)
.mwb (MySQL Workbench Model)
.nupkg (NuGet packages for .NET)
.ora (OpenRaster, used e.g. by MyPaint)
.otp (OpenOffice templates)
.otx (OpenOffice dictionary)
.pez (Prezi Presentation)
.phar (PHP application package)

And also…

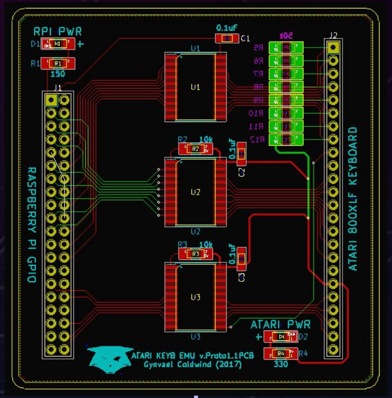
.raz, .saz (Fiddler request history)
.rjt (RealPlayer template?)
.sbsx (PowerPoint shapes)
.snagacc (SnagIt plugin)
.sob (OpenOffice something...)
.sublime-package (Sublime Package, obviously)
.sxw (SUN XML Writer)
.thmx (Microsoft Office document themes)
.vs (RealPlayer UI files?)
.vsb (AIDA64 sidebar gadget)
.wmz (Windows Media Player skins)

And also...

.wsz (LiteStep themes?)
.xmind (xmind documents)
.xmt (xmind template)
.xpi (Firefox Cross-Platform Installer Module)
.xps (XML Paper Specification)
.zxp (PalleteApp extension)

The list is not exhaustive.

Furthermore

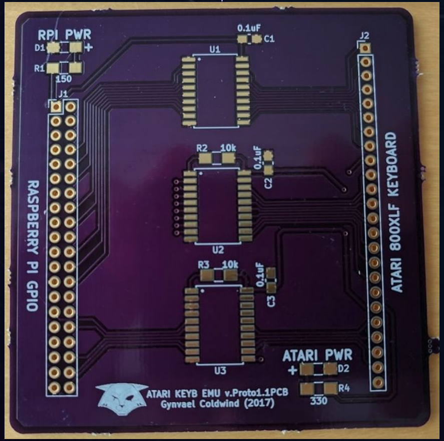


It's used when:

- Uploading GERBER files to your PCB manufacturer
  - Or more general: uploading a bundle of files somewhere
  - Or downloading a bundle of files

- Don't forget about combining ZIPs with other file formats, e.g. EXE (SFX)

- And well, sending files to your friends too.



What can a ZIP store?

What can be stored inside a ZIP archive?

Files

Directories

What can be stored inside a ZIP archive?

Files Directories

Technically identical with one **minor** difference

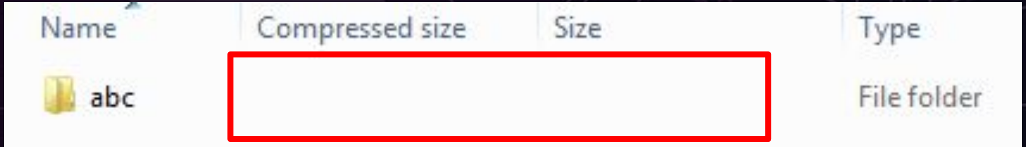
4.4.15 external file attributes: (4 bytes)

The mapping of the external attributes is
host-system dependent (see 'version made by'). For
MS-DOS, the low order byte is the **MS-DOS directory
attribute byte.** If input came from standard input, this
field is set to zero.

What can be stored inside a ZIP archive?

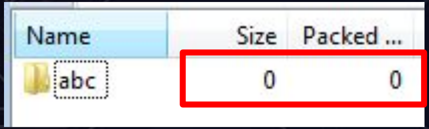
Steganograph y

Files Directories



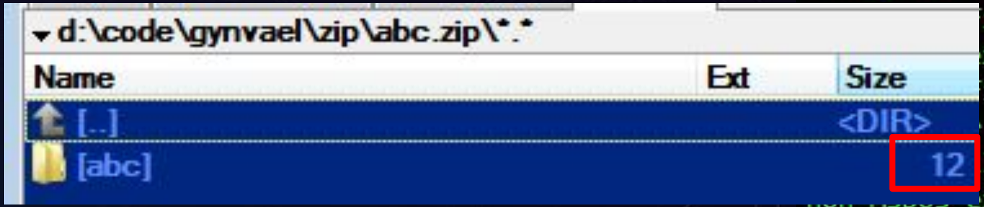
Name	Compressed size	Size	Type
abc			File folder

Windows Explorer



Name	Size	Packed ...
abc	0	0

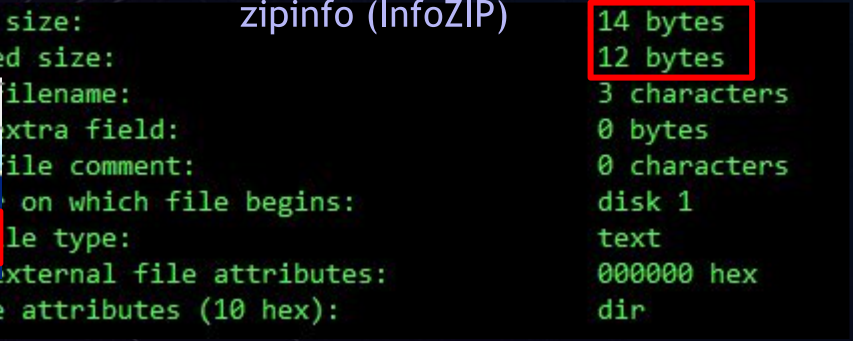
7-zip



d:\code\gynvael\zip\abc.zip\*.*

Name	Ext	Size
[..]		<DIR>
[abc]		12

Total Commander



zipinfo (InfoZIP)

```
compressed size:                                   14 bytes
uncompressed size:                                 12 bytes
          ilename:                                 3 characters
          xtra field:                              0 bytes
          ile comment:                             0 characters
           on which file begins:                   disk 1
          le type:                                 text
          xternal file attributes:                 000000 hex
MS-DOS file attributes (10 hex):                   dir
```

What can be stored inside a ZIP archive?

Files Directories

Technically identical with one **minor** difference

4.4.15 external file attributes: (4 bytes)

> **The mapping of the external attributes is host-system dependent** (see 'version made by'). For MS-DOS, the low order byte is the MS-DOS directory attribute byte. If input came from standard input, this field is set to zero.

What can be stored inside a ZIP archive?

Files

Directories

Symlinks

Cyber Secure CloudDisk

File names in ZIP

Stored in several locations per entry:

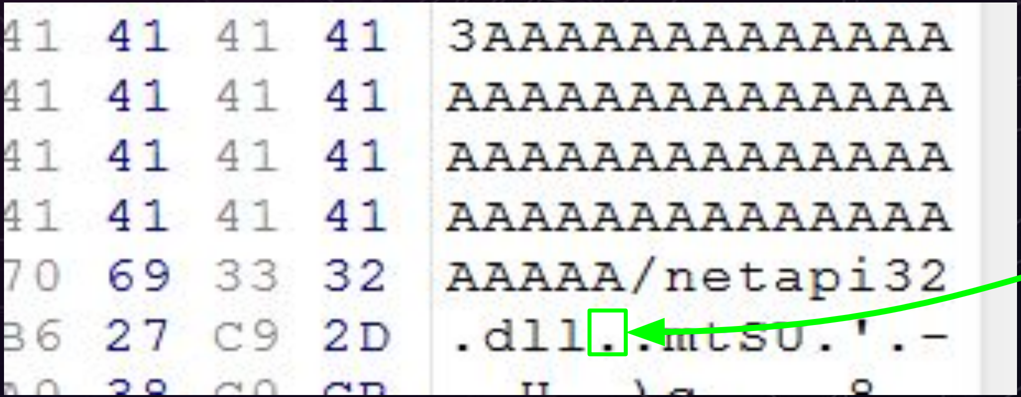
- Local File Header
- Central Directory Header
- Extra: Info-ZIP Unicode Path Extra Field

Which one to use (trust)?

Technically it's possible to create any number of separate Extra entries per file in both LFH and CDH

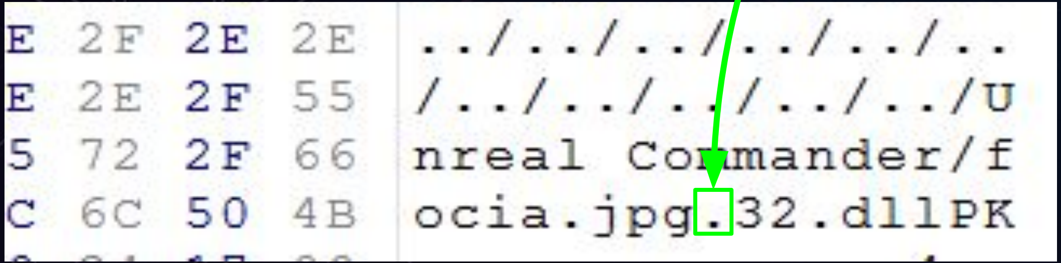
File names in ZIP

Unreal Commander exploit for bug reported in 2007



```
41  41  41  41   3AAAAAAAAAAAAAAA
41  41  41  41   AAAAAAAAAAAAAAAA
41  41  41  41   AAAAAAAAAAAAAAAA
41  41  41  41   AAAAAAAAAAAAAAAA
70  69  33  32   AAAAA/netapi32
B6  27  C9  2D   .dll..mtSU.'.-
```

nul byte



```
E  2F  2E  2E   ../../../../../..
E  2E  2F  55   /../../../../../U
5  72  2F  66   nreal Commander/f
C  6C  50  4B   ocia.jpg.32.dllPK
```

File names in ZIP

Other problems with names (just enumerating ideas):

- Files with the same name

File names in ZIP

Other problems with names (just enumerating ideas):

- Files with the same name
- lower-upper case (i.e. Windows/Unix)

File names in ZIP

Other problems with names (just enumerating ideas):

- Files with the same name
- lower-upper case (i.e. Windows/Unix)
- NTFS ADS :$data

File names in ZIP

Other problems with names (just enumerating ideas):

- Files with the same name
- lower-upper case (i.e. Windows/Unix)
- NTFS ADS :$data
- Network Share names (\\127.0.0.1\C$\...)

https://googleprojectzero.blogspot.de/2016/02/the-definitive-guide-on-win32-to-nt.html

File names in ZIP

Other problems with names (just enumerating ideas):

- Files with the same name
- lower-upper case (i.e. Windows/Unix)
- NTFS ADS :$data
- Network Share names (\\127.0.0.1\C$\...)
- Very Long file names (not well known?)

http://www.icewall.pl/?p=467

File names in ZIP

Other problems with names (just enumerating ideas):

- Files with the same name
- lower-upper case (i.e. Windows/Unix)
- NTFS ADS :$data
- Network Share names (\\127.0.0.1\C$\...)
- Very Long file names (not well known?)
- Encoding issues (UTF-8 vs OS vs IBM 437)

File names in ZIP

Other problems with names (just enumerating ideas):

- Files with the same name
- lower-upper case (i.e. Windows/Unix)
- NTFS ADS :$data
- Network Share names (\\127.0.0.1\C$\...)
- Very Long file names (not well known?)
- Encoding issues (UTF-8 vs OS vs IBM 437)
- XSS in the <script>filename</script>

Or SQL Injection, in the end the file name is just a text field.

File names in ZIP

Other problems with names (just enumerating ideas):

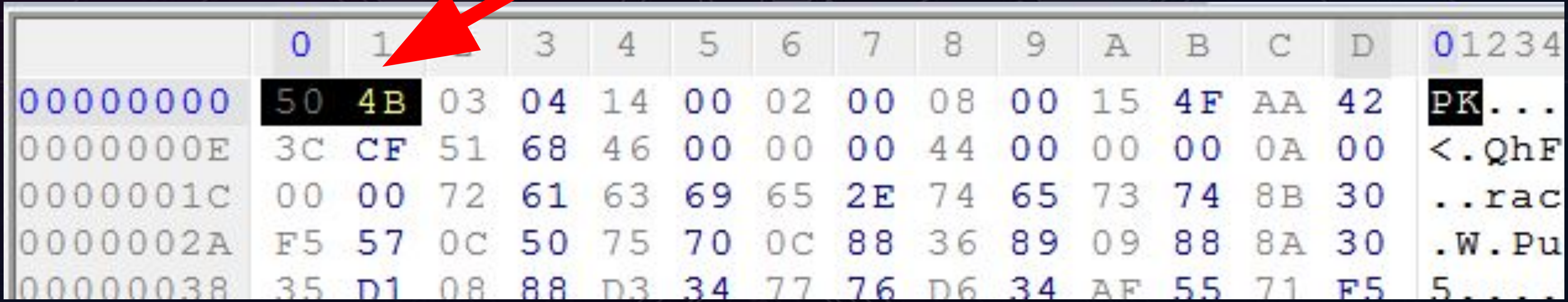
- Files with the same name
- lower-upper case (i.e. Windows/Unix)
- NTFS ADS :$data
- Network Share names (\\127.0.0.1\C$\...)
- Very Long file names (not well known?)
- Encoding issues (UTF-8 vs OS vs IBM 437)
- XSS in the <script>filename</script>

Cyber Secure CloudDisk

ZIP Format 101 & Recovering ZIPs

ZIP Magic: PK

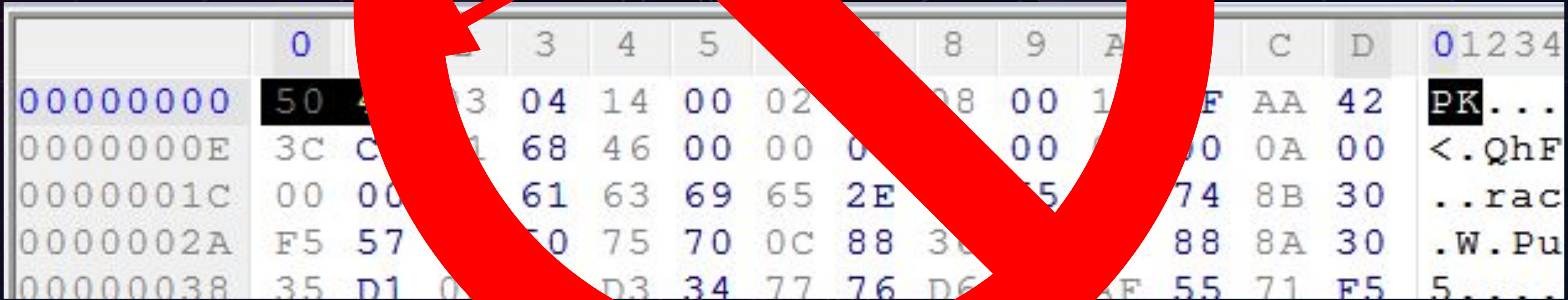
As you know, a ZIP file starts with letters "PK".



```
           0  1  2  3  4  5  6  7  8  9  A  B  C  D   01234
00000000  50 4B 03 04 14 00 02 00 08 00 15 4F AA 42   PK...
0000000E  3C CF 51 68 46 00 00 00 44 00 00 00 0A 00   <.QhF
0000001C  00 00 72 61 63 69 65 2E 74 65 73 74 8B 30   ..rac
0000002A  F5 57 0C 50 75 70 0C 88 36 89 09 88 8A 30   .W.Pu
00000038  35 D1 08 88 D3 34 77 76 D6 34 AF 55 71 F5   5....
```

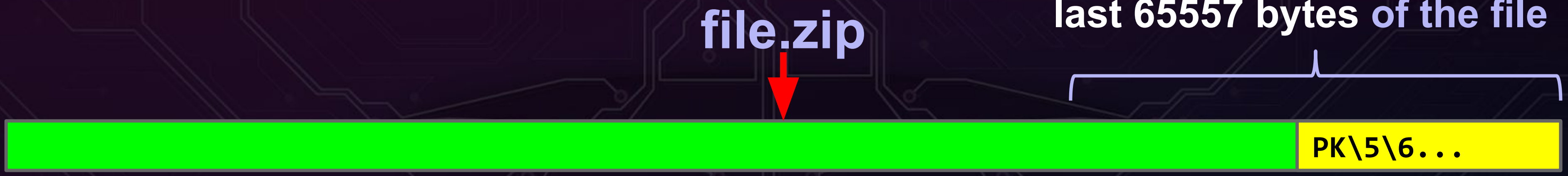
ZIP Magic: PK

As you know, … file starts wi… letters "PK".

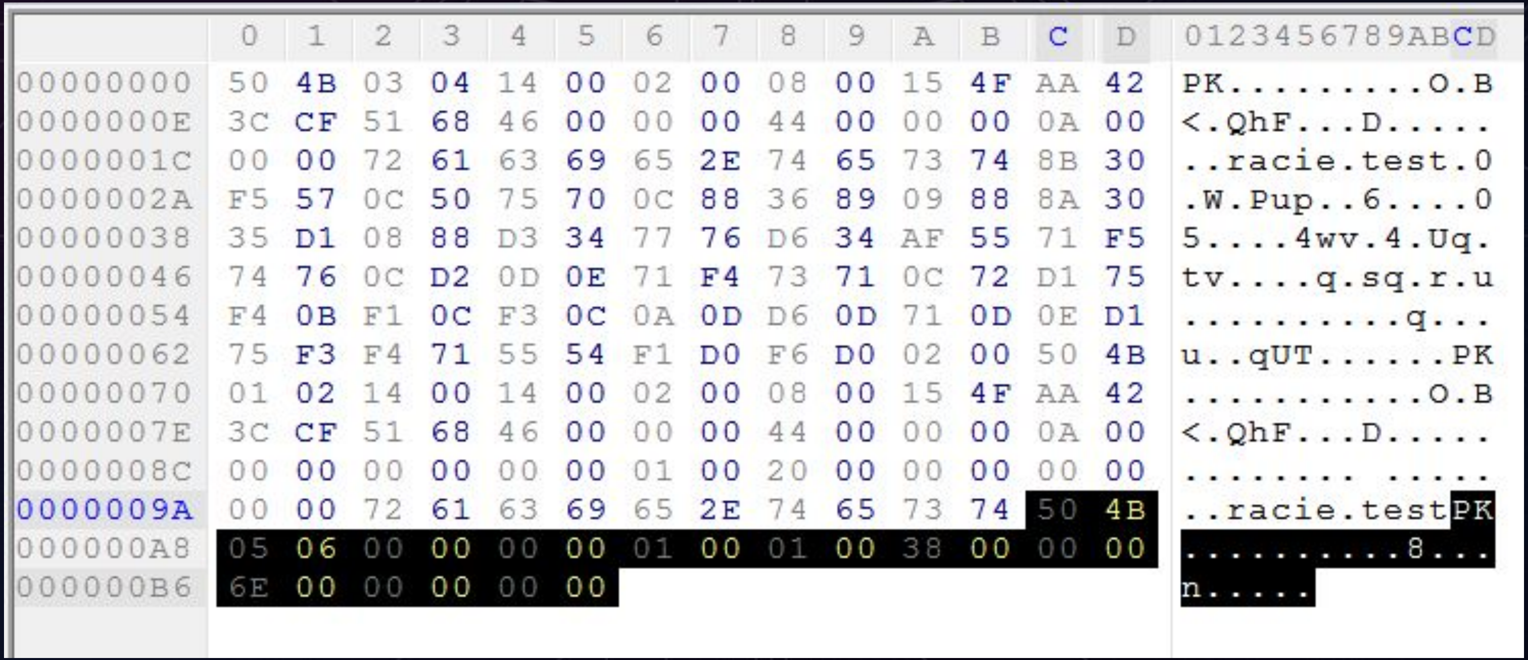


```
              0         3  4  5        8  9  A     C  D    01234
00000000   50 4      3 04 14 00 02     8 00 1    F AA 42   PK...
0000000E   3C C      1 68 46 00 00 0     00      0 0A 00   <.QhF
0000001C   00 00       61 63 69 65 2E    5      74 8B 30   ..rac
0000002A   F5 57     0 75 70 0C 88 3           88 8A 30   .W.Pu
00000038   35 D1 0      D3 34 77 76 D6       F 55 71 F5   5....
```

Let's try that again...



look for the "header" in the last 65557 bytes of the file



```
          0  1  2  3  4  5  6  7  8  9  A  B  C  D   0123456789ABCD
00000000  50 4B 03 04 14 00 02 00 08 00 15 4F AA 42  PK.........O.B
0000000E  3C CF 51 68 46 00 00 00 44 00 00 00 0A 00  <.QhF...D.....
0000001C  00 00 72 61 63 69 65 2E 74 65 73 74 8B 30  ..racie.test.0
0000002A  F5 57 0C 50 75 70 0C 88 36 89 09 88 8A 30  .W.Pup..6....0
00000038  35 D1 08 88 D3 34 77 76 D6 34 AF 55 71 F5  5....4wv.4.Uq.
00000046  74 76 0C D2 0D 0E 71 F4 73 71 0C 72 D1 75  tv....q.sq.r.u
00000054  F4 0B F1 0C F3 0C 0A 0D D6 0D 71 0D 0E D1  ..........q...
00000062  75 F3 F4 71 55 54 F1 D0 F6 D0 02 00 50 4B  u..qUT......PK
00000070  01 02 14 00 14 00 02 00 08 00 15 4F AA 42  ............O.B
0000007E  3C CF 51 68 46 00 00 00 44 00 00 00 0A 00  <.QhF...D.....
0000008C  00 00 00 00 00 00 01 00 20 00 00 00 00 00  ........ .....
0000009A  00 00 72 61 63 69 65 2E 74 65 73 74 50 4B  ..racie.testPK
000000A8  05 06 00 00 00 00 01 00 01 00 38 00 00 00  ..........8...
000000B6  6E 00 00 00 00 00                          n.....
```

Proper parsing must start from the end

```
4.3.16  End of central directory record:

    end of central dir signature    4 bytes  (0x06054b50)
    [...]
    total number of entries in
    the central directory           2 bytes
    size of the central directory   4 bytes
    offset of start of central
    [...]
    .ZIP file comment length        2 bytes
    .ZIP file comment       (variable size)
```

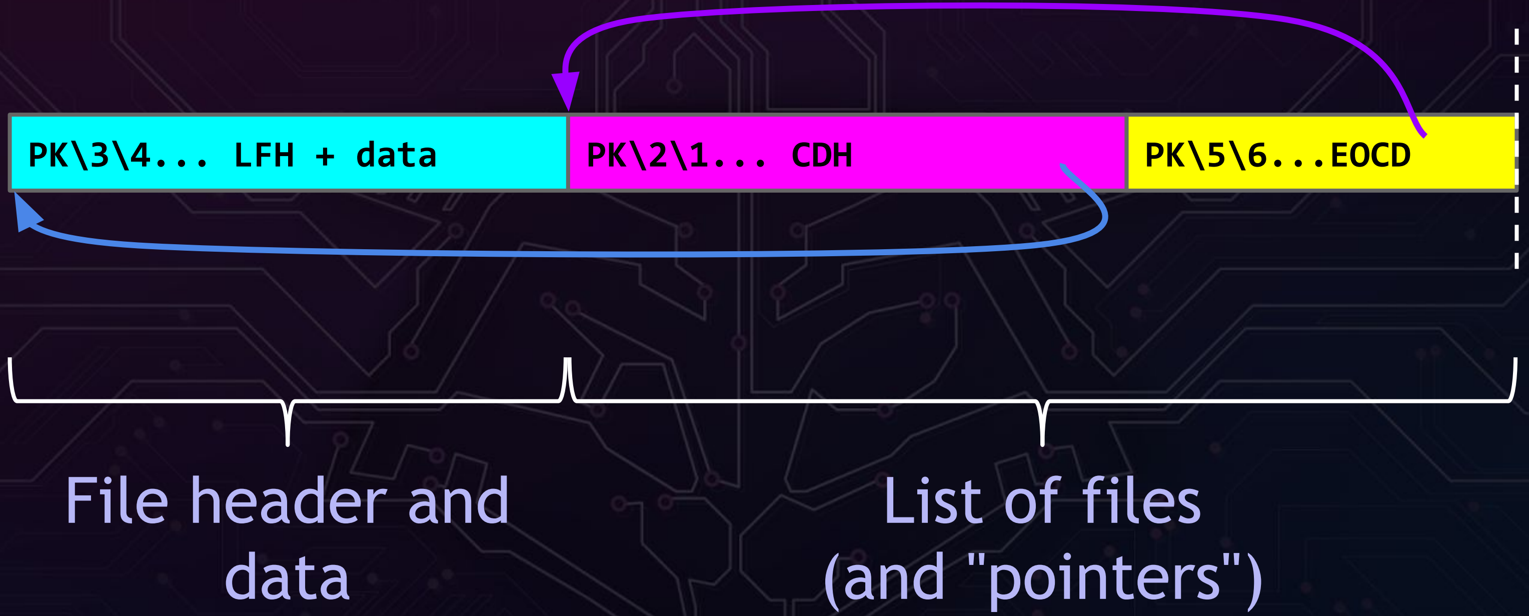
22 bytes

$0000-$FFFF
0-65535

In total: from 22 to 65557 bytes

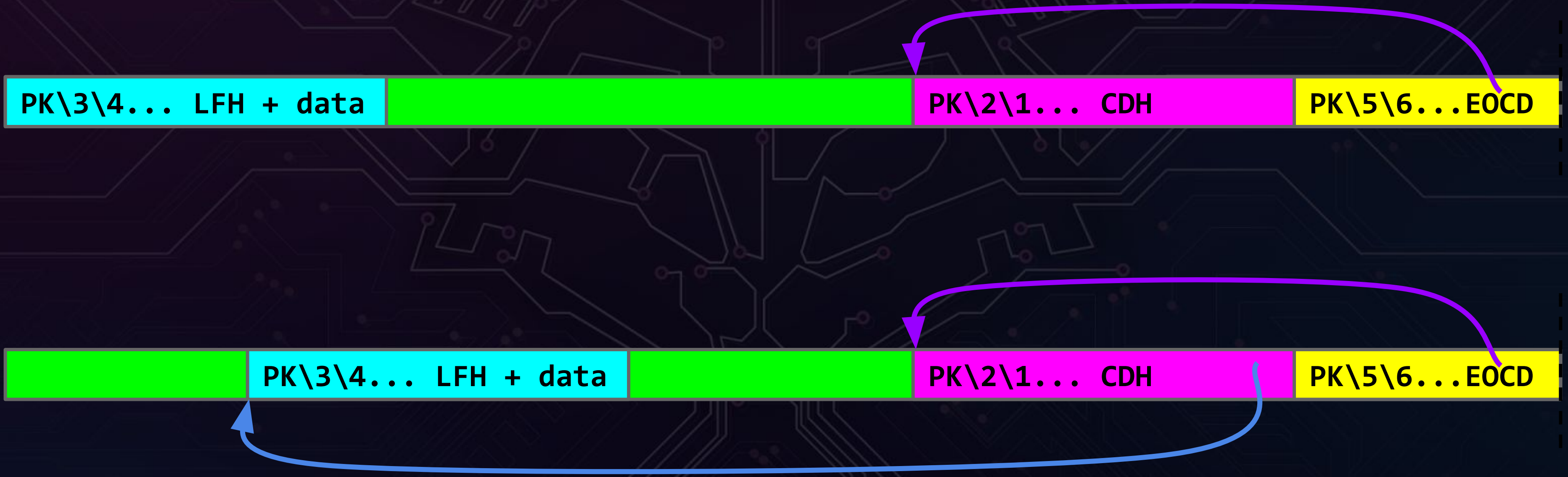
(so, the PK\5\6 sig will be "somewhere" between EOF-65557 do EOF-22)

An overview of a single-file ZIP

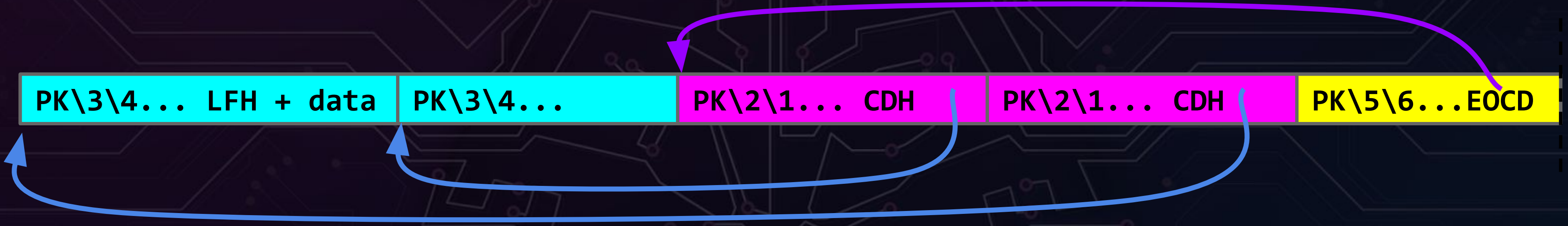


Each file has two "headers":
Local one, right next to data - **Local File Header**
And the more verbose entry in the list of files - **Central Directory Header**

Please note that it's a "pointer"-based format



More files in a ZIP



Central Directory Header (CDH)

PK\2\1... CDH

```
central file header signature   4 bytes  (0x02014b50)
version made by                 2 bytes
version needed to extract       2 bytes
general purpose bit flag        2 bytes
compression method              2 bytes
last mod file time              2 bytes
last mod file date              2 bytes
crc-32                          4 bytes
compressed size                 4 bytes
uncompressed size               4 bytes
file name length                2 bytes
extra field length              2 bytes
file comment length             2 bytes
disk number start               2 bytes
internal file attributes        2 bytes
external file attributes        4 bytes
relative offset of local header 4 bytes

file name (variable size)
extra field (variable size)
file comment (variable size)
```

these are redundant between LFH and CDH (xslx)

What if more CDHs point to the same LFH?

PK\3\4... LFH + data

Local File Header (LFH)

PK\3\4... LFH + data

local file header signature	4 bytes	(0x04034b50)
version needed to extract	2 bytes	
general purpose bit flag	2 bytes	
compression method	2 bytes	
last mod file time	2 bytes	
last mod file date	2 bytes	
crc-32	4 bytes	
compressed size	4 bytes	
uncompressed size	4 bytes	
file name length	2 bytes	
extra field length	2 bytes	

file name (variable size)
extra field (variable size)
file data (variable size)

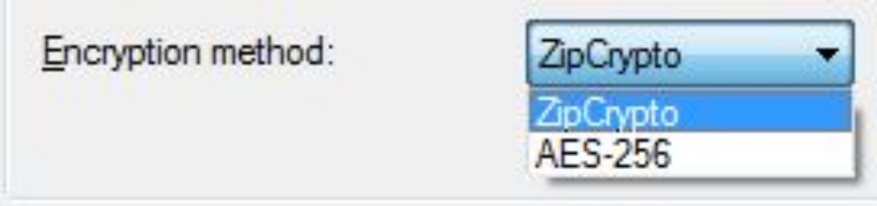
How to repair a ZIP?

- Try some programs like Info-ZIP's zip -F and -FF
- Manually - copy correct-looking data between LFH and CDH
- There is also the "CTF brute force approach"

Cyber Secure CloudDisk

Legacy ZIP encryption

Legacy ZipCrypto



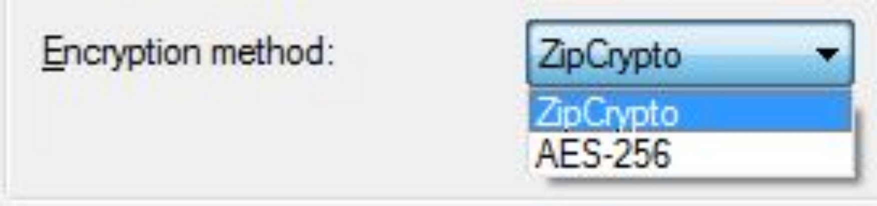
Encryption method: ZipCrypto

7-zip

Also other methods may be available (e.g. RC4)

- Still used by some tools by default
- Really good backward compatibility
- Technically a byte-based stream cipher

Legacy ZipCrypto



Encryption method: ZipCrypto

7-zip

Also other methods may be available (e.g. RC4)

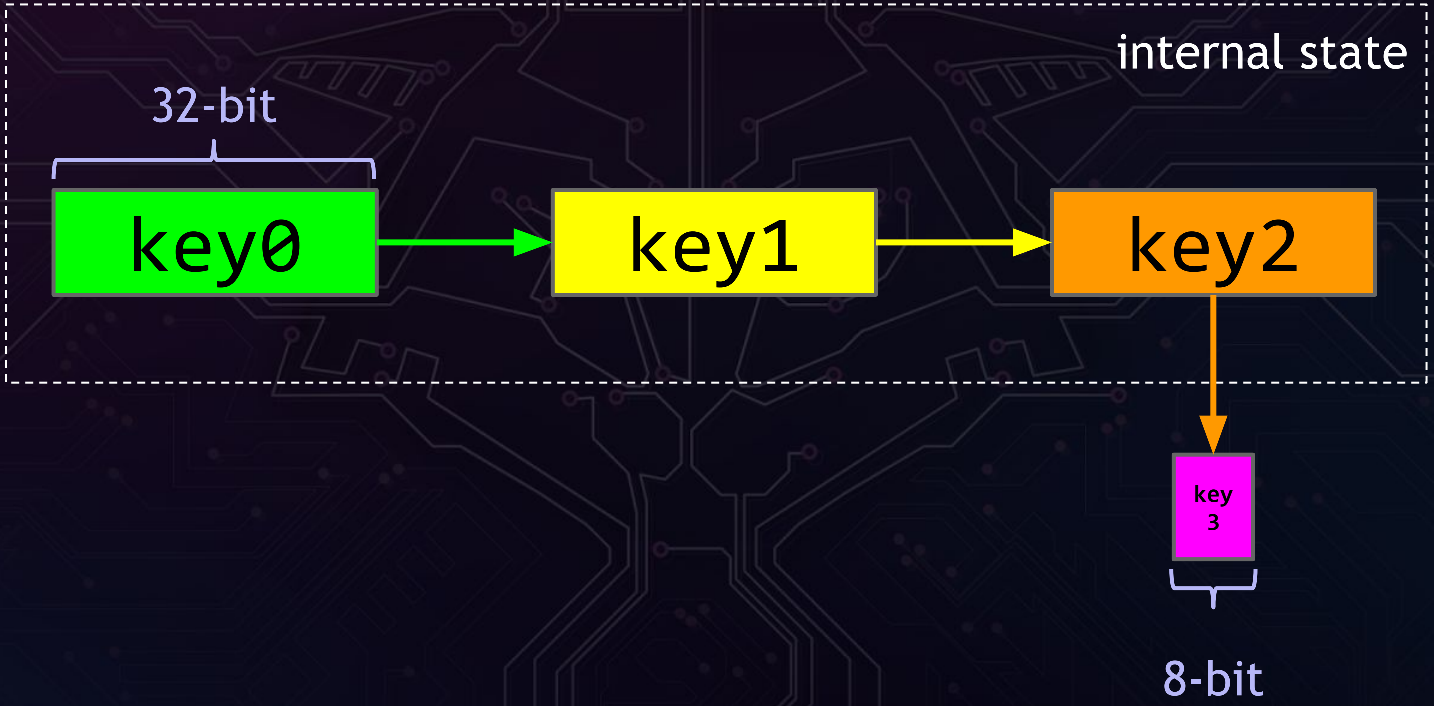
- Still used by some tools by default
- Really good backward compatibility
- Technically a byte-based stream cipher
- A "known plaintext" attack known from 1994
    - With further improvements in 2002

Legacy ZipCrypto - papers to read

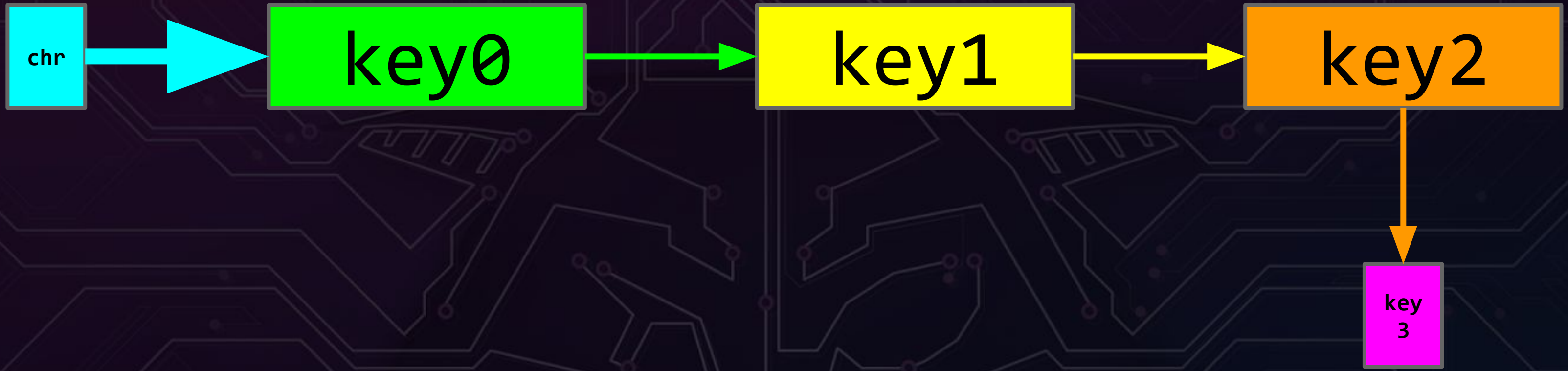
"A Known Plaintext Attack on the PKZIP Stream Cipher" (1994)
by Eli Biham and Paul C. Kocher

"ZIP Attacks with Reduced Known Plaintext" (2002)
by Michael Stay

Legacy ZipCrypto - a 96-bit byte-oriented stream cipher



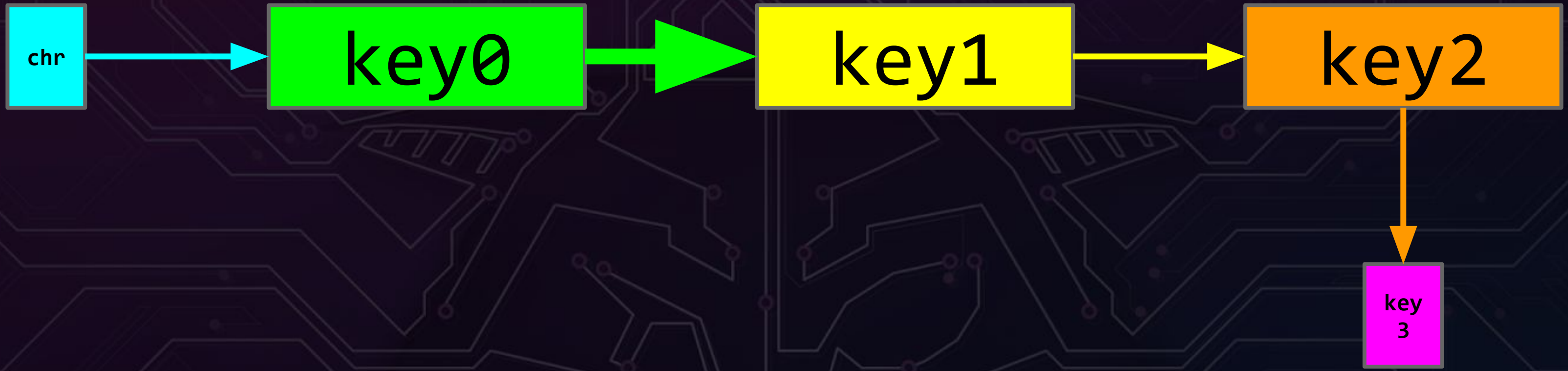
Legacy ZipCrypto - updating the key after each byte enc



(from the first paper)

```
key0 = crc32(key0, chr)
```

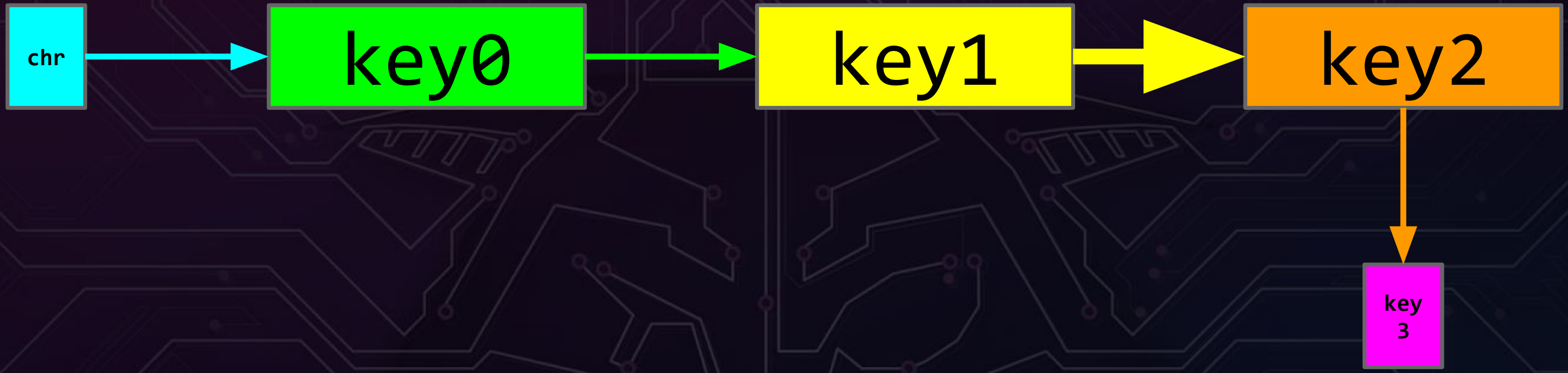
Legacy ZipCrypto - updating the key after each byte enc



(from the first paper)

```
key0 = crc32(key0, chr)
key1 = (key1 + LSB(key0)) * 134775813 + 1
```

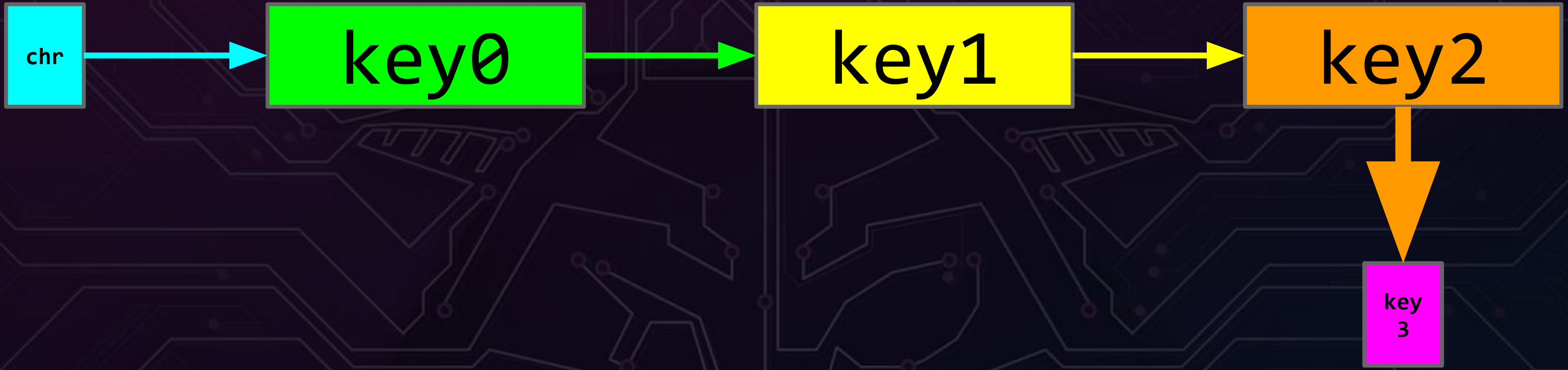
Legacy ZipCrypto - updating the key after each byte enc



(from the first paper)

```
key0 = crc32(key0, chr)
key1 = (key1 + LSB(key0)) * 134775813 + 1
key2 = crc32(key2, MSB(key1))
```

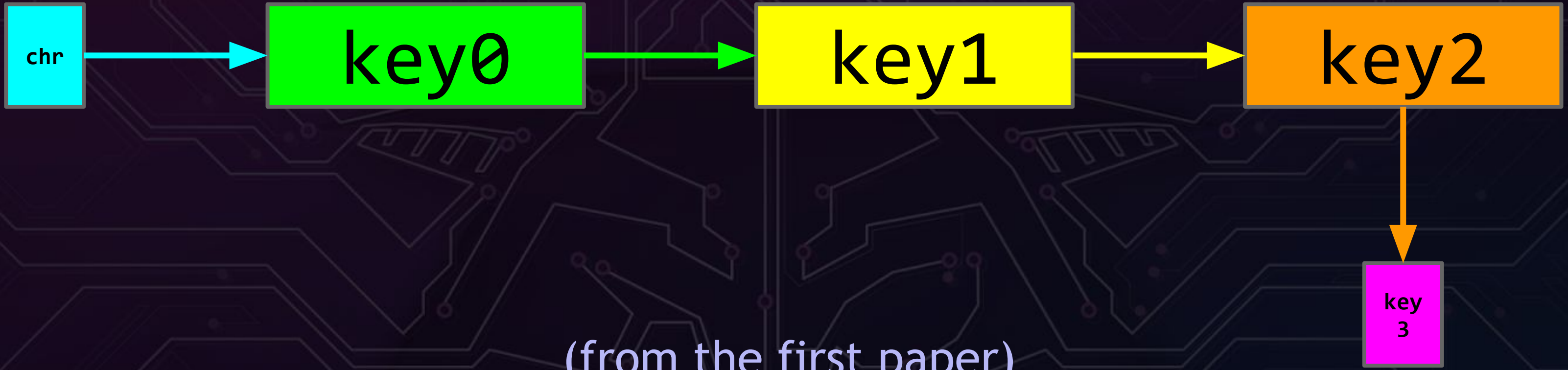
Legacy ZipCrypto - updating the key after each byte enc



(from the first paper)

```
key0 = crc32(key0, chr)
key1 = (key1 + LSB(key0)) * 134775813 + 1
key2 = crc32(key2, MSB(key1))
temp = key2 | 3  (16 bottom bits)
key3 = LSB((temp * (temp ^ 1)) >> 8)
```

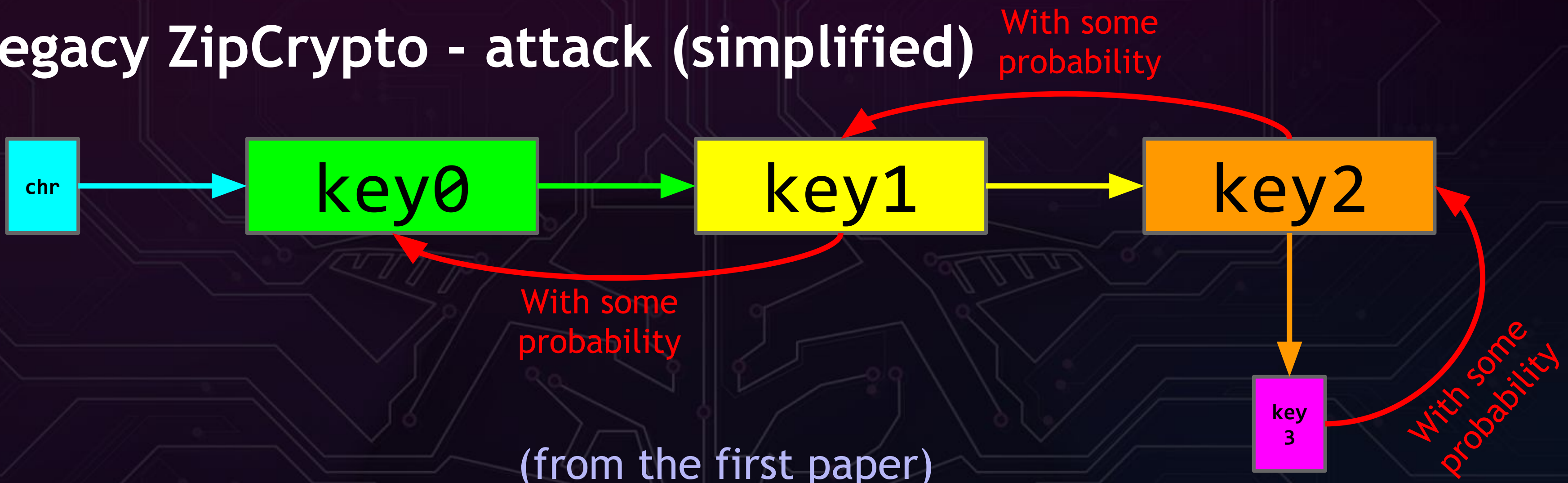
Legacy ZipCrypto - encryption



(from the first paper)

$$C \leftarrow \mathtt{chr} \oplus \mathtt{key3}$$

Legacy ZipCrypto - attack (simplified)



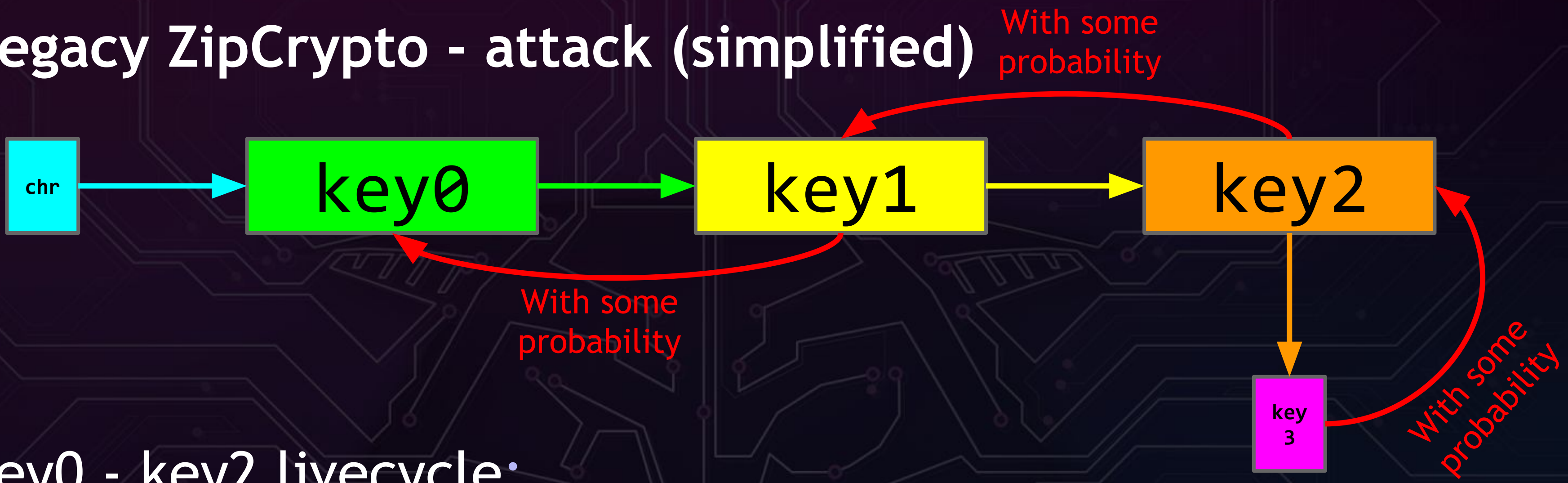
(from the first paper)

$$C \leftarrow \mathtt{chr} \oplus \mathtt{key3}$$

(if we know plaintext, then...)

$$\mathtt{key3} == C \oplus \mathtt{chr}$$

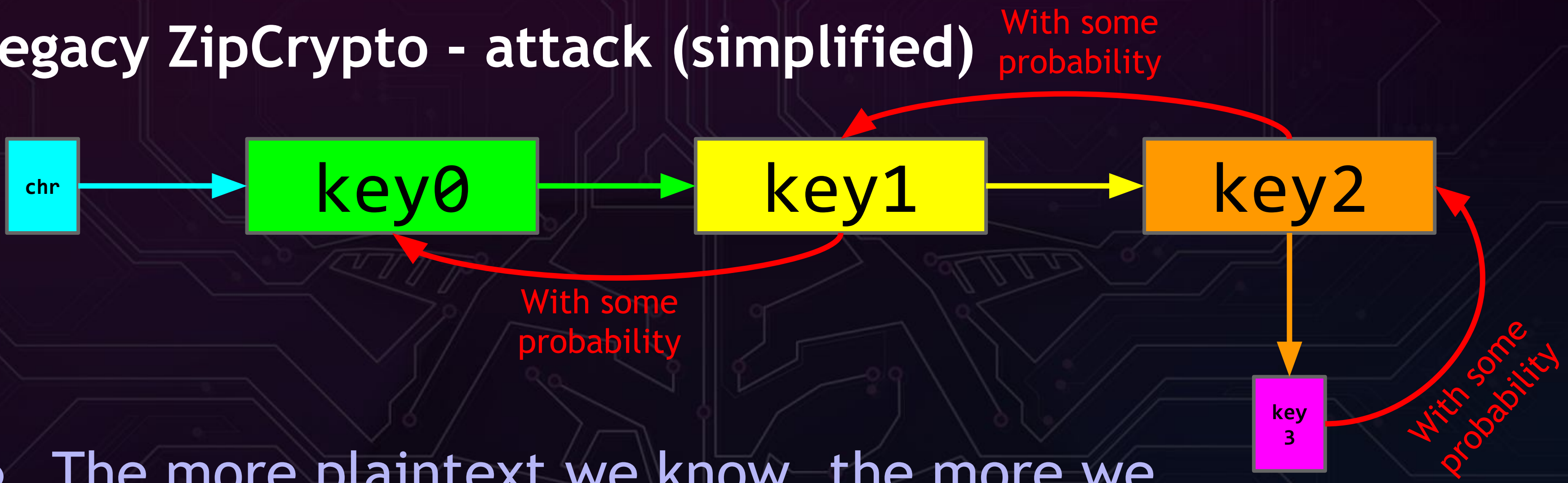
Legacy ZipCrypto - attack (simplified)



key0 - key2 livecycle:

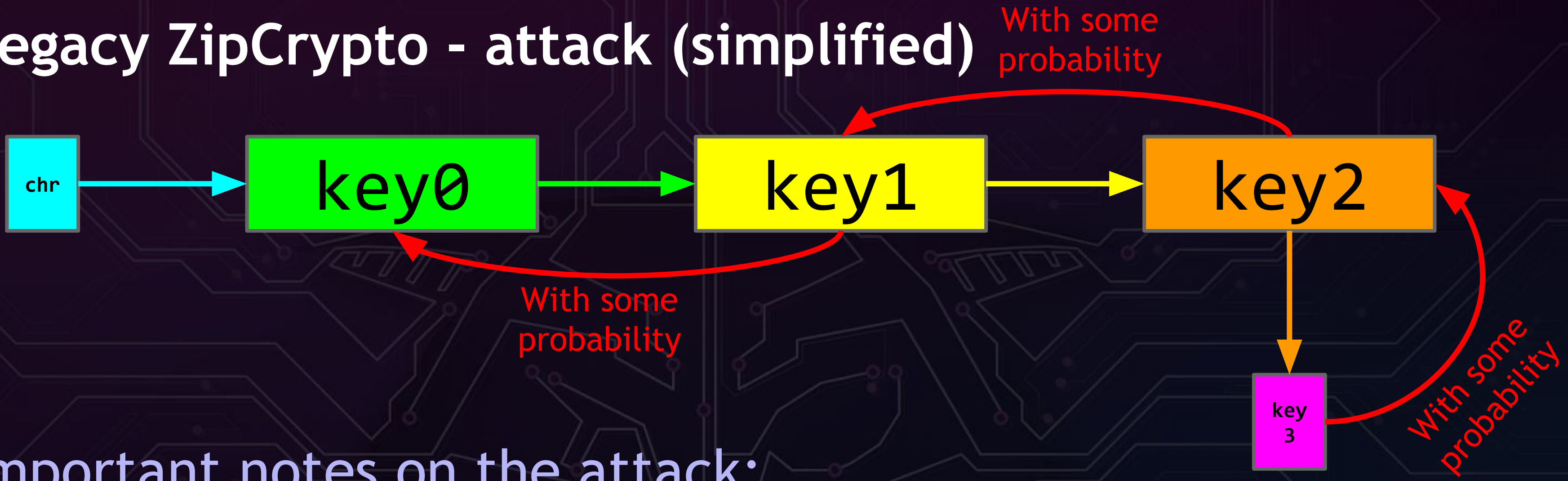
1. Init with constants (0x12345678, ...)
2. Update with password (discard output, keep state)
3. Update with 12 bytes of "random data"
4. Update with data to encrypt

Legacy ZipCrypto - attack (simplified)



- The more plaintext we know, the more we can reason about the state of key0 - key2.
- If we unroll to initial state (after password & "random" data is fed), we can decrypt everything.
- Bonus: In some cases we can even get the password.

Legacy ZipCrypto - attack (simplified)



Important notes on the attack:

- Minimum of 13 bytes of known **compressed plaintext**
- **"compressed"** is the keyword here(different apps generate different output when compressing)
- It takes a few minutes to run

Cyber Secure CloudDisk

https://www.unix-ag.uni-kl.de/~conrad/krypto/pkcrack.html

ZIP format and multiple personalities

Proper parsing must start from the end

Let's look at this slide again

4.3.16 End of central directory record:

```
end of central dir signature    4 bytes  (0x06054b50)
[...]
total number of entries in
the central directory           2 bytes
size of the central directory   4 bytes
offset of start of central
[...]
.ZIP file comment length        2 bytes
.ZIP file comment        (variable size)
```

22 bytes

$0000-$FFFF
0-65535

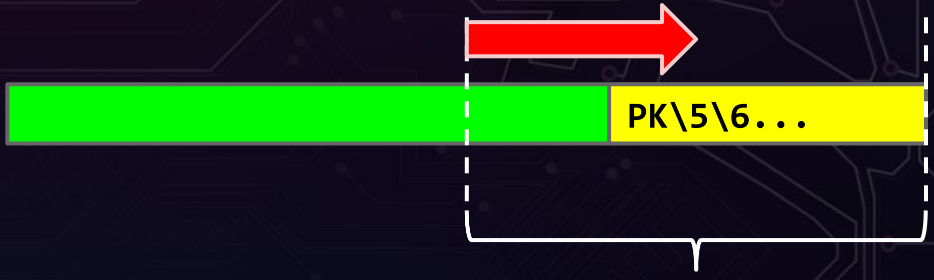
In total: from 22 to 65557 bytes

(so, the PK\5\6 sig will be "somewhere" between EOF-65557 do EOF-22)

So... how do we search for the right comment size?

"Start First"

Start left most at **`EOF-65557`**, and then decrease the comment size one by one.

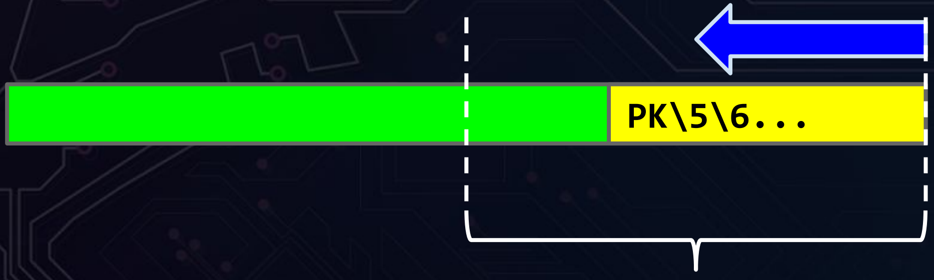


range of possibilities

"End First"

(well, usually there are no comments)

Start at the end at **`EOF-22`**, and then increase comment size one by one.



range of possibilities

Why do we care about EOCDH at all?

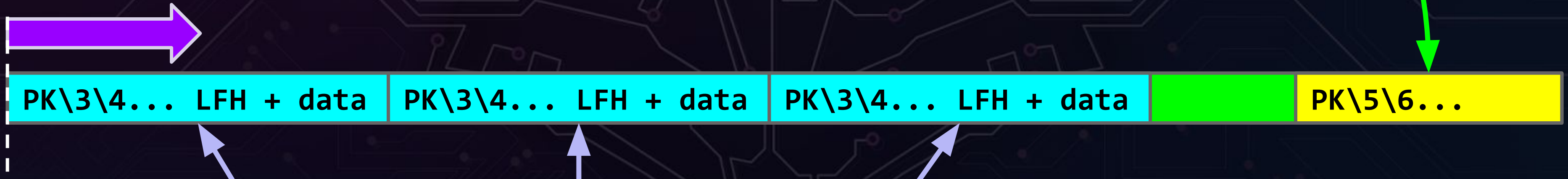
(who needs this anyway)

"stream"

EOCDH is redundant, let's ignore it and parse only Local File Headers going from offset 0 in the file

(usually this is faster)

(99.9% of ZIPs can be successfully parsed like this)



(individual files in the archive)

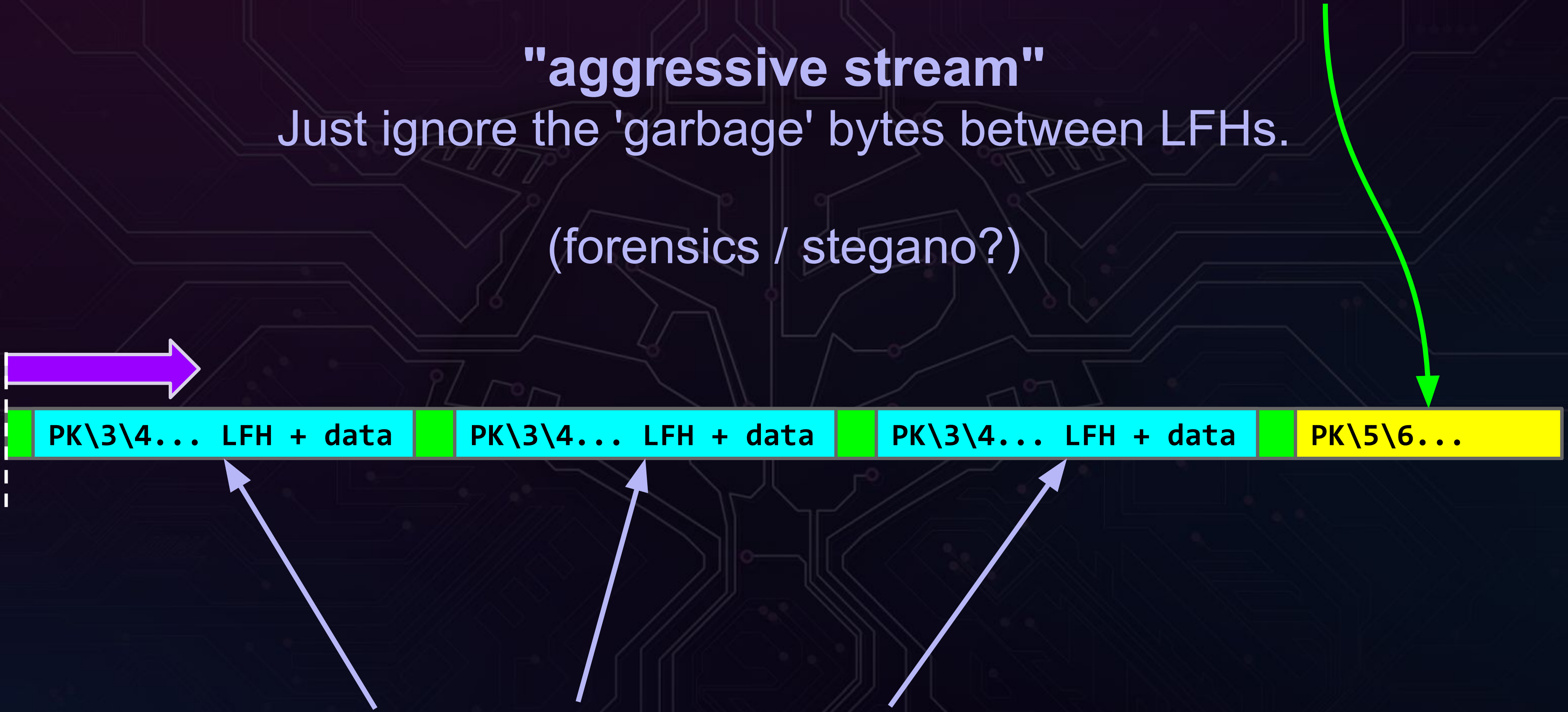
Why do we care about EOCDH at all?

(who needs this anyway)

"aggressive stream"

Just ignore the 'garbage' bytes between LFHs.

(forensics / stegano?)



PK\3\4... LFH + data

PK\3\4... LFH + data

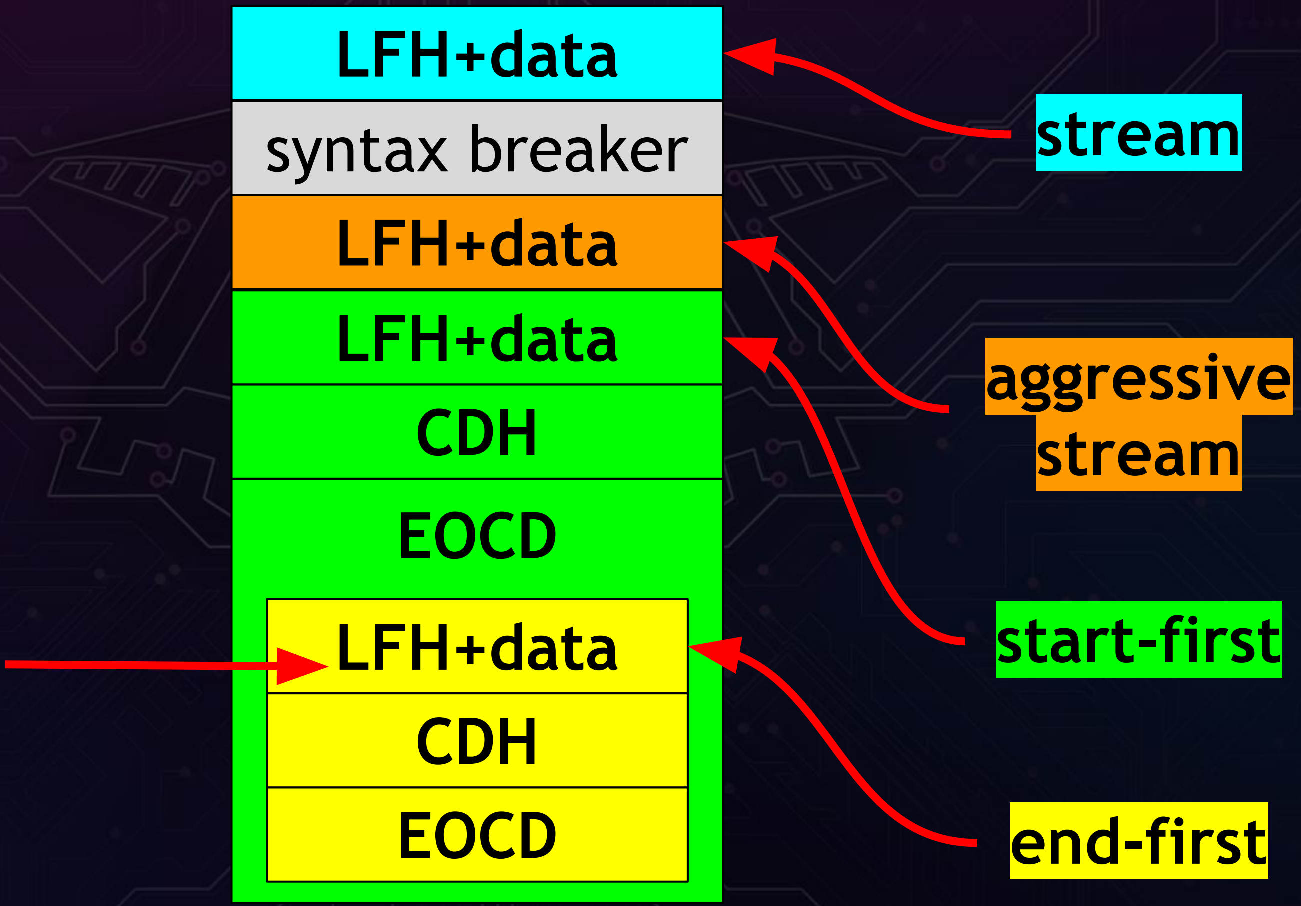
PK\3\4... LFH + data

PK\5\6...

(individual files in the archive)

Let's see how this works in practice - abstract.zip

Architecture of abstract.zip



LFH+data

syntax breaker

LFH+data

LFH+data

CDH

EOCD

LFH+data

CDH

EOCD

stream

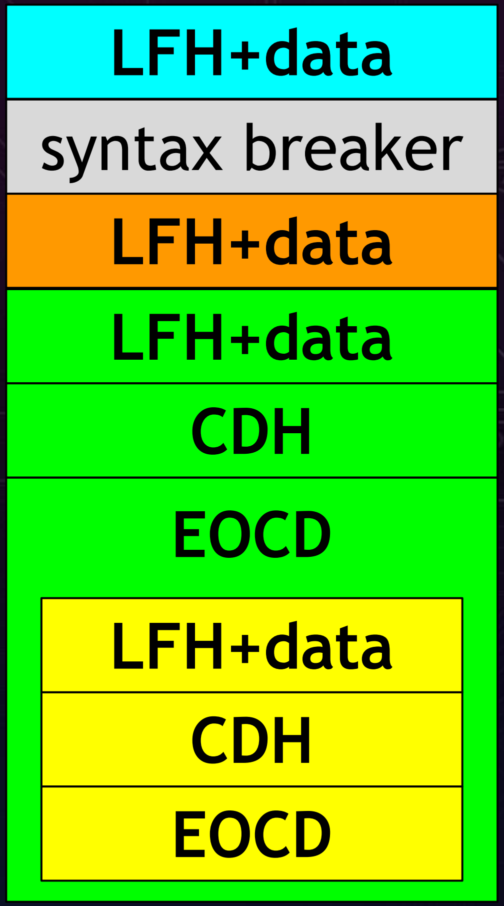
aggressive stream

start-first

end-first

yellow part is technically a comment for the green archive

Testing abstract.zip



LFH+data

syntax breaker

LFH+data

LFH+data

CDH

EOCD

LFH+data

CDH

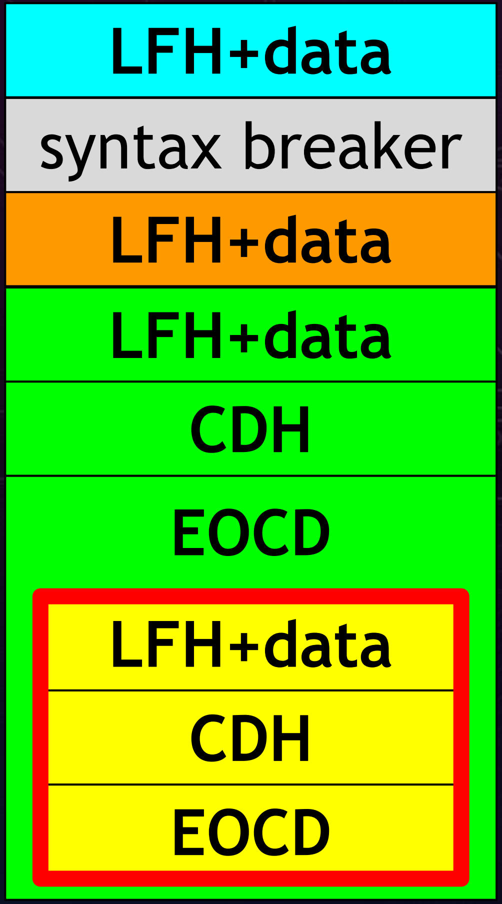
EOCD

Kudos for help in testing this:

- Mulander
- Felix Groebert
- Salvation
- j00ru

Note: data might be a little stale (2013)

EndFirst style



LFH+data

syntax breaker

LFH+data

LFH+data

CDH

EOCD

LFH+data

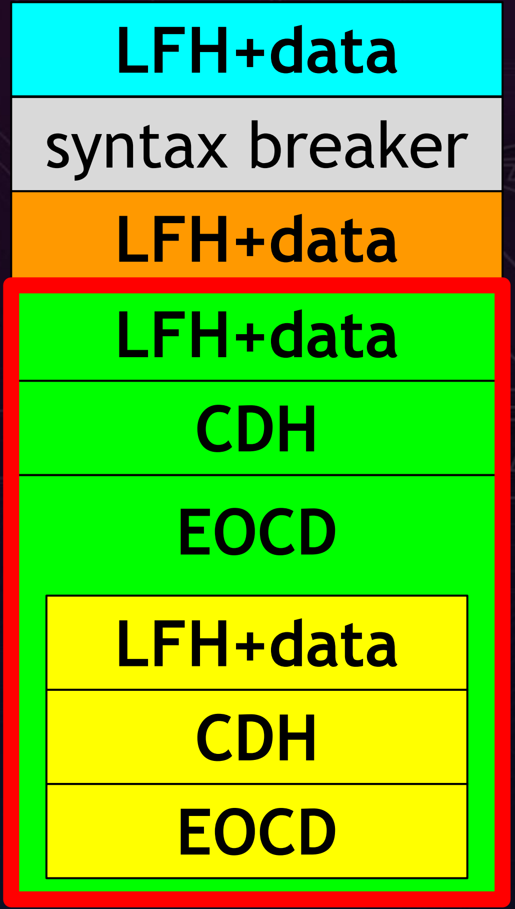
CDH

EOCD

All of these

Total Commander 8.01
UnZip 6.00 (Debian)
Midnight Commander
Windows 7 Explorer
ALZip
KGB Archiver
7-zip
b1.org
Python zipfile
JSZip
C# DotNetZip
perl Archive::Zip
Jeffrey's Exif Viewer
WOBZIP
GNOME File Roller
WinRAR
OSX UnZip
zip.vim v25
Emacs Zip-Archive mode
Ada Zip-Ada v45
Go archive/zip
Pharo smalltalk 2.0 ZipArchive
Ubuntu less
Java ZipFile

StartFirst style



LFH+data

syntax breaker

LFH+data

LFH+data

CDH

EOCD

LFH+data

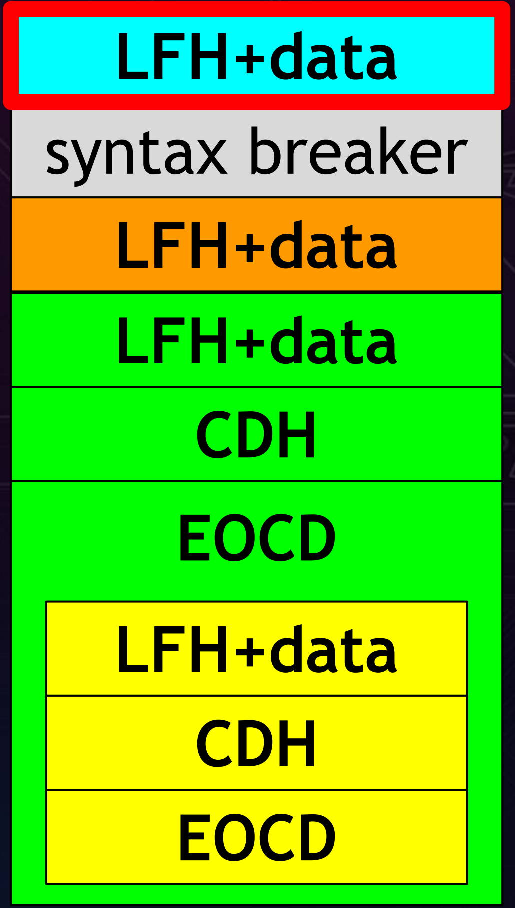
CDH

EOCD

Only these

```
PHP ZipArchive
PHP zip_open ...
PHP zip:// wrapper
tcl + tclvfs + tclunzip
```

Stream style



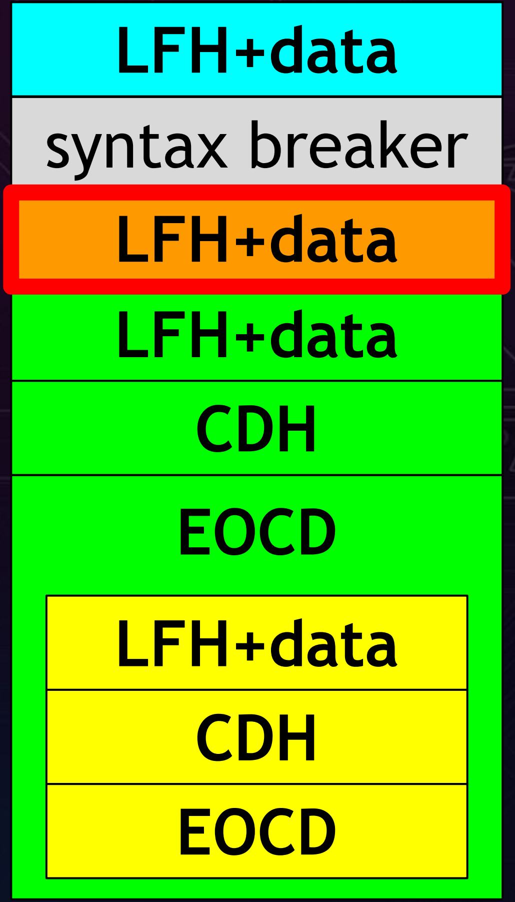
Only these

Note that Java
is both here and
in the 'EndFirst' list

```
Ruby rubyzip2
Java ZipArchiveInputStream
java.util.zip.ZipInputStream
```

Aggressive Stream style

All files should be found



LFH+data

syntax breaker

LFH+data

LFH+data

CDH

EOCD

LFH+data

CDH

EOCD

binwalk

Is this a problem?

- Usually no.
- However, if multiple libraries/apps are used, consistency is key.

Think:

1. Content verification done using one approach
2. Actual unpack done using another approach

Is this a problem?

- Usually no.
- However, if multiple libraries/apps are used, consistency is key.

Warning:
There might be other parsing inconsistencies between libraries!

Ideally use a single library.

Think:
1. Content verification done using one approach
2. Actual unpack done using another approach

Also, AV (warning: stale data, 2013)

EICAR test results (using VT):

- Most use End-First approach
- Some use the Aggressive Stream way
- These use the Stream method:
  - VBA32
  - NANO-Antivirus
  - Norman
  - F-Prot
  - Agnitum
  - Commtouch

Cyber Secure CloudDisk

ZIP encryption and CRC32

CRC-32 is a fun one!

Some facts:

- ZIP uses the 0xEDB88320 polynomial
- CRC-32 is not a cryptographic hash
- Reversible to some extent
  - Actually there is quite a lot of fun stuff you can do with CRC

Example reading: "Reversing CRC - Theory and Practice" by M. Stigge, H. Plotz, W. Muller, J-P. Redlich

CRC-32 is a fun one!

Definitely MUST NOT be in the clear!

In some versions, metadata is in the clear

LFH

```
local file header signature     4 bytes  (0x04034b50)
version needed to extract       2 bytes
general purpose bit flag        2 bytes
compression method              2 bytes
last mod file time              2 bytes
last mod file date              2 bytes
crc-32                          4 bytes
compressed size                 4 bytes
uncompressed size               4 bytes
file name length                2 bytes
extra field length              2 bytes

file name (variable size)
extra field (variable size)
file data (variable size)
```

CRC-32 is a fun one!

Definitely MUST NOT be in the clear!

... but it sometimes is.

See also:

- "ZIP file encryption weakness"
  by K. Matusiewicz, N. Wochtman

- Also on CTFs!
  Task: "A hopeless case" (CONFidence CTF 2015)

Cyber Secure CloudDisk

https://github.com/theonlypwner/crc32

Miscellaneous, i.e. all the things not mentioned so far.

ZIP vs low-level

Just enumerating ideas:

- Known and well loved Buffer Overflow
  - compressed size < after-unpack(data)
  - long file names
- Memory disclosure
  - uncompressed size > after-unpack(data)
  - uncompressed size > compressed size for 'STORED'
- Signed/Unsigned issues in various fields
  - size, offsets

GIFAR / Ange CorkaMIX / etc (binary polyglots)

http://en.wikipedia.org/wiki/Gifar

https://code.google.com/p/corkami/wiki/mix

CorkaMIX is a Windows executable, and also a working PDF document, Jar (Zip + Class + manifest), and HTML + JavaScript files.

PHP LFI, ZIP polyglot upload, zip:// or phar://

More ZIP steganography

Mostly useful on CTFs / in forensics.

- Office XML Steganography Tool (extra field)
- "Empty" space between files
- More data than "uncompressed size" field claims there is. Or data after the DEFLATE "end tag".
- Extra fields, comments.
- Files of the same name, or with \0 in the name
- Well, abstract.zip ;)
- Stegano using the compression protocol/layer

Bonus - ZIP download!

It's a "list + offsets" type format, so...

HTTP Range: parameter can be used to download individual files from a ZIP archive hosted online.

```
> python zipdl.py http://example.com/example.zip
File Name                    ...       Size
readme_EndFirst.txt          ...        231
> python zipdl.py http://example.com/example.zip readme_EndFirst.txt
> ls -la readme_EndFirst.txt
-rw-r----- 1 gynvael gynvael 231 May 13 14:45 readme_EndFirst.txt
>
```

http://gynvael.coldwind.pl/n/python_zipdl

Bonus - ZIP download! Pretty easy in Python...

```
class MyFileWrapper:
  def __init__(self, url):
    --> HEAD ...

  def seek(self, offset, whence):

  def tell(self):

  def read(self, amount=-1):
    --> GET ...
        Range: bytes=%u,%u

z = zipfile.ZipFile( MyFileWrapper_object)
```

Bonus - ZIP DoS aka pack bombs

Three types:

1. **small ZIP --> super large file**
   (unreal cmd uncomp size)

2. **small ZIP --> multiple ZIPs --> multiple ZIPs from each --> ...**
   --> very large files

3. **Infinite recursion (ZIP quine)**
   http://research.swtch.com/zip ← EPIC!
   (by Russ Cox)

Bonus - ZIP compression

Usually DEFLATE (zlib), but also:

- Uncompressed (STORED)
- BZ2
- XZ
- WavPack
- (several others)

THE END

Are there any easy questions?

(If there are only hard ones then I'm sorry, but we run out of time ;>)

P.S. We're hiring at Google - reach out to me if you're interested!

E-mail: gynvael@coldwind.pl Twitter: @gynvael YT: GynvaelEN

Blog: http://gynvael.coldwind.pl/ (Soon also: http://gynvael.live)