Caring for file formats

Ange Albertini
Troopers 2016

TL;DR

- Attack surface with file formats is too big.
- Specs are useless (just a nice ‘guide’), not representing reality.
- We can’t deprecate formats because we can’t preserve and we can’t define how they ***really*** work

- We need open good libraries to simplify landscape, and create a corpus to express the reality of file format, which gives us real “documentation”.
- Then we can preserve and deprecate older format, which reduces attack surface.
- From then on, we can focus on making the present more secure.

- We don’t need “new” formats: we need ‘alive’ specs and files corpus. Otherwise specs will always diverge from reality.

ANGE ALBERTINI

reverse engineering

VISUAL DOCUMENTATION

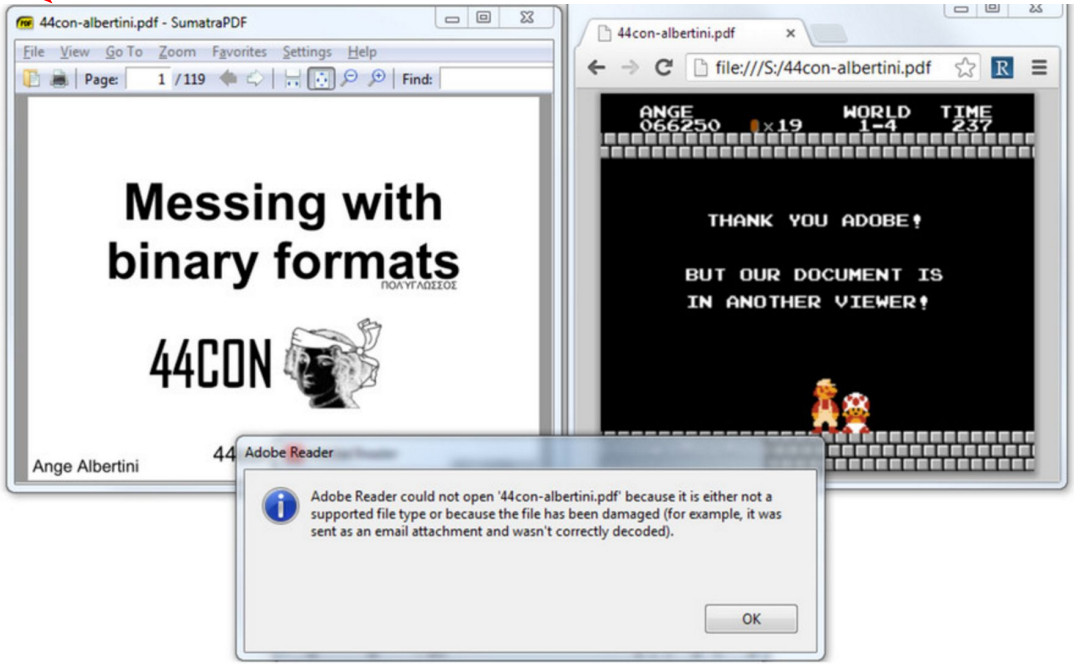
@angealbertini

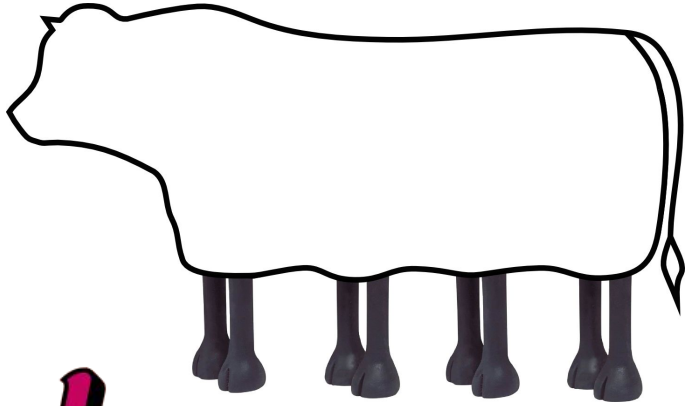
ange@corkami.com

http://www.corkami.com

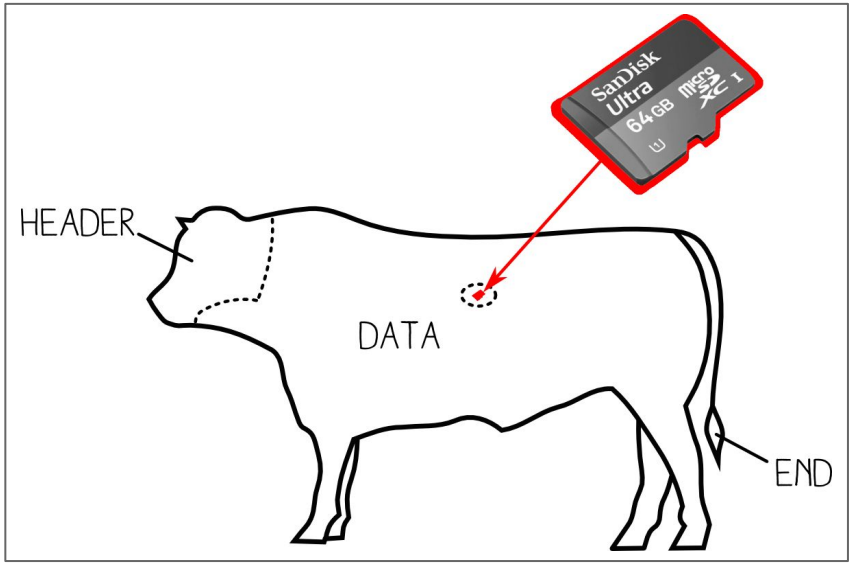
Welcome to my talk!

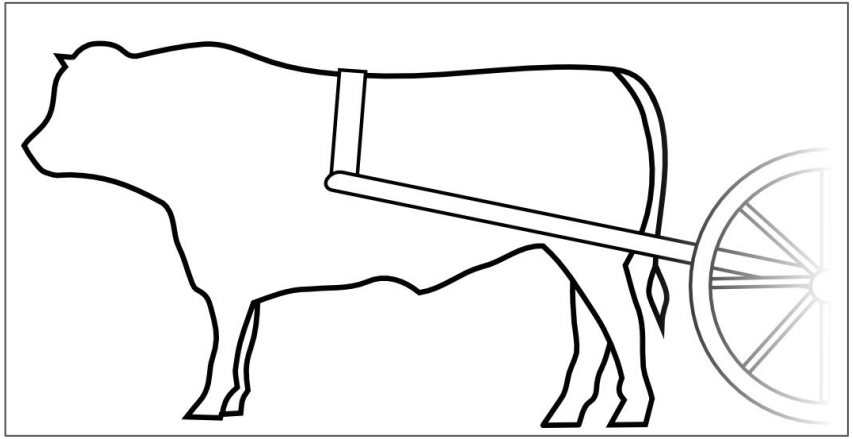
I make polyglots (multi-type files), schizophrenics (multi-behavior)...





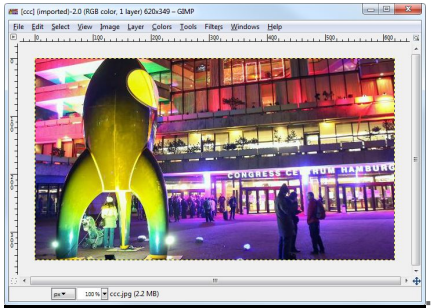
Funky File Formats





I tried to explain file formats with cows…
But that didn't really tell why people should care.

JPG

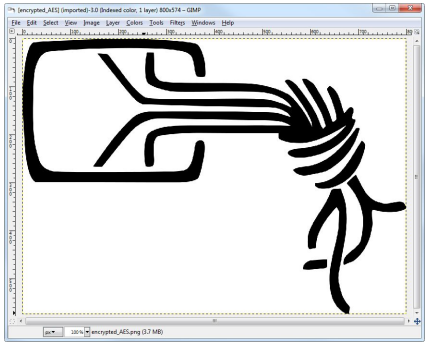


JAR
(ZIP + CLASS)

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

>
```

AES_{K_1}



PNG

3DES

PDF

AES_{K_2}

FLV

I really like to play with file formats...

PoC||GTFO

Proof of Concept or Get The Fuck Out

Preacherman
Editor of Last Resort
TEXnician
Editorial Whipping Boy
Funky File Supervisor
Assistant Scenic Designer

Manul Laphroaig
Melilot
Evan Sultanik
Jacob Torrey
Ange Albertini
Philippe Teuwen

and sundry others

I'm a part of PoC||GTFO, for which I'm a file format user and abuser.

PoC||GTFO: many file formats

- Articles
  PDFLaTeX PDFBook Inkscape GhostScript Scribus Blender Gimp Fontforge PDFFont Mutool
- Proof of Concept
  Qpdf Xpdf Ruby Python Bash Truecrypt Wavpack Audacity Baudline Sox Tar Zip MkIsoFS LSnes PngOpt JpegSnoop AdvPNG Nasm Qemu BPGEnc

And *many* custom scripts handling file formats in unconventional ways…

I'm interested about hardware preservation

and digital preservation.

My interests

- Using file formats
  - graphics, 3d, music…
- Abusing file formats
  - polyglot, schizophrenia, hash collisions…
- Preserving file formats
  - Retro-gaming, digital archeology...

What is a file format?

~~A miserable little pile of secrets~~

Not just a sequence of binary

If you [/your program] generate
a picture of any kind,
you might want to export
the result to something
that you can re-use later.
(same for any form of information)

What is a file format?

A computer dialect to communicate between communities.

File formats are community connectors.

Don’t think so?
Try exporting everything as XML ;)

Most people don’t care about <actor>

They only care about <roles>
We mostly care about the input/output.

Example:

We don’t care about GIF

We mostly care about its characteristics
and how easy it is to use.

No need to be emotional,
and stay in our comfort zone.

We don’t really care about file formats.

We care about their caracteristics.

Not groundbreaking,
but supported “everywhere”.

Why should infosec care?

Fuzz formats. Blame “bad” devs.
Collect CVEs. Boast your ego.

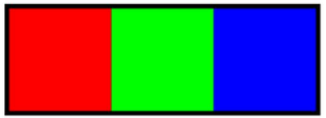
```
10 PRINT “SOLVED ANYTHING YET?”
20 GOTO 10
```

Attack surface

- 1 OS = N supported formats
- For each format:
  - How many parsers?
  - For each parser:
    - Which version, compiler...

The PGM or PPM formats are the easiest way to convert any data in valid grayscale or RGB pictures.

```
<signature> <whitespace>
P5
<width> <whitespace> <height> <whitespace>
3 1
<max. value> <whitespace>
255
 .ÿ
<raw RGB values>
00 80 FF
```



```
<signature> <whitespace>
P6
<width> <whitespace> <height> <whitespace>
3 1
<max. value> <whitespace>
255
ÿ  ÿ  ÿ
<raw RGB values>
FF 00 00 00 FF 00 00 00 FF
```

But most people don’t know it’s supported out of the box by many softwares.

We should reduce the attack surface.

How many unsuspected supported [sub-]formats and parsers?

https://lcamtuf.blogspot.com/2014/10/psa-dont-run-strings-on-untrusted-files.html

How many file formats supported by your browser ?

By your OS?

How many do you really need ?

Think “embedded”.

Capacity is still too cheap:
we keep stacking formats/features,
which doesn’t solve anything.

It’s a problem everywhere.
We keep losing ground.

<!--

PoC||GTFO 10

pocorgtfo10.lsmv - Adobe Reader

File Edit View Window Help

Open 8 / 88 93.4% Tools Fill & Sign Comment

ter looking at other options, we chose to use the Pokémon Red version, which uses a more reliable SGB detection routine that gets us access to the full SGB palette, a custom border, and consistent timing benefits we'll discuss later.[7] Using Pokémon Red also has another added benefit in that the entire game has been skillfully disassembled.[8]

3.3 The Emulator

When we started this project in August 2014, the only emulator capable of emulating an SGB inside of an SNES at a low enough level for our needs was the BSNES emulator. Unfortunately, although BSNES is very accurate at emulating an SNES, it doesn't do a very good job of emulating an SGB. The Gambatte Dot-Matrix Game Boy (DMG) emulator is likewise very accurate, but is unable to emulate an SGB on its own. Ilari was able to create a hybrid emulation core using BSNES to emulate the SNES↔DMG interface chip while using Gambatte for DMG emulation. This was a considerable undertaking, but in the end the emulator was very usable, albeit somewhat slow, as properly emulating the synchronization between the SNES CPU and the DMG CPU is a challenge. Ilari continued to provide emulator development and scripting support throughout the project.

3.4 The Hardware

We didn't just want to exploit a game in the sandbox of a console emulator and call it a Proof of Concept. We [illegible] properly and create an actual [illegible] on real hardware. Only [illegible] of our team (dwangoAC) had all of the [illegible] console, a SGB cartridge, a copy of Pokémon Red [illegible] known as a "bot."[9] Because we wanted to stream [illegible] an older, serial-over-USB connected device, namely

8.50 x 11.00 in

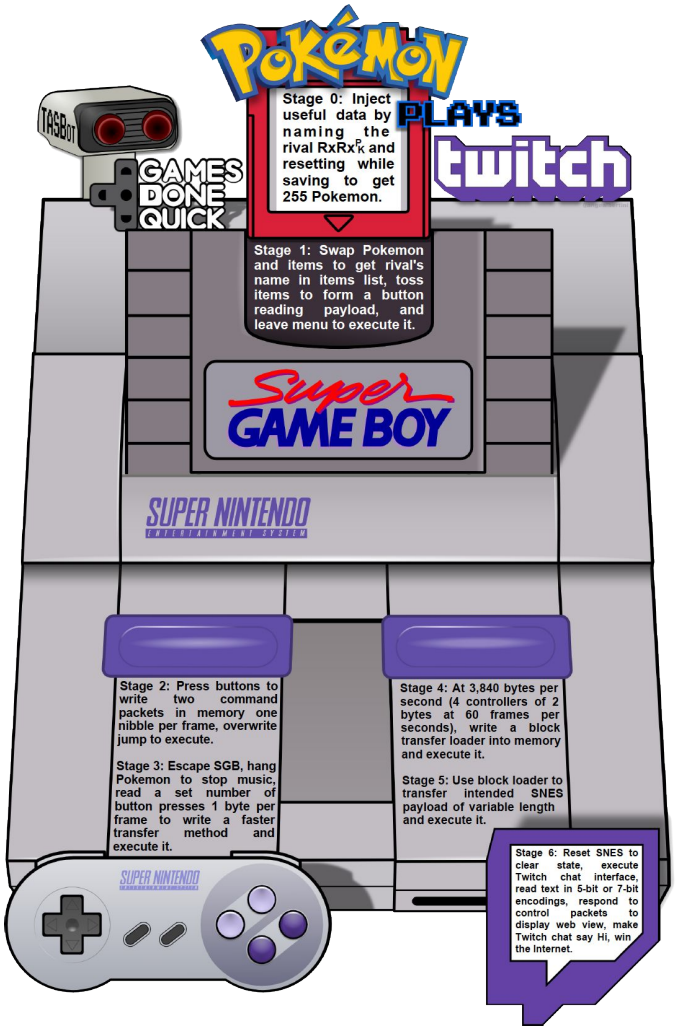
Figure 1 – The legendary TASBot

3.5 The Plan

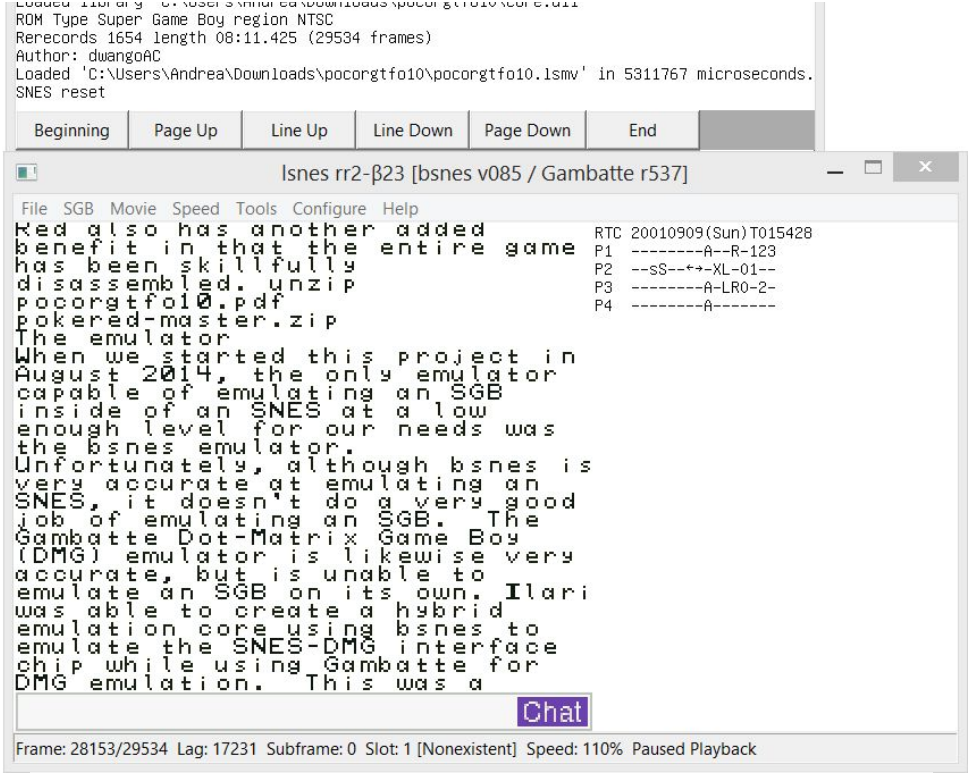
We were initially unsure what kind of payload to create once we had gained the ability to execute arbitrary code on the SNES. Initially we investigated methods of showing crude video, but abandoned it after spending far too much time failing to increase the datarate and running into limits with the processing speed of the SNES's 65C816 CPU. An IRC discussion about Twitch Plays Pokémon[10] led dwangoAC and p4plus2 to brainstorm what it would take to incorporate similar elements into our [illegible] the concept of *Pokémon Plays Twitch* was hatched – where a Pokémon character enters a Twitch chat room and "plays" Twitch. In the end, we took it to the next level by giving Red a voice in a chat interface on the SNES and giving TASBot, the robot holding the replay board, the ability to

"Pokemon plays Twitch"

1. Exploit a GameBoy game via input
2. Take over the Super GameBoy
3. Take over the Super Nintendo



The file itself can perform the exploit (on the hardware or an emulator).

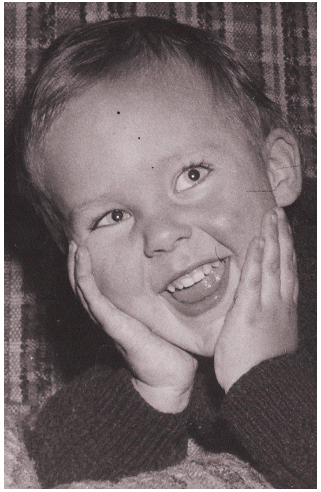


The payload displays the article.

-->

PoC||GTFO 10 is a PoC-ception:

- a PDF article describing the exploit
- a file performing the exploit
  (to display the article)



“young celebs”

What they were supposed to be doesn’t really matter.

What file formats were supposed to be
doesn't matter anymore,
what they are now is all we care.

Security cares about current reality,
not obsolete theory.

We can blame bad parsers. What about the file formats?

If the map is unclear enough, you'll get lost anyway.

A blurry file format will never lead to a clean parser.

2 ways to communicate

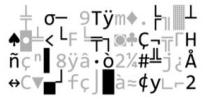
use a ready-made translator:
an import/export library

Write your own:
read the specs.

Landscapes

```
>crypto_hash *
test0.jpg 13990732b0d16c3e112f2356bd3d0dad1....
test1.jpg 13990732b0d16c3e112f2356bd3d0dad1....
```

To exploit hash collisions, I abused JPEG.
To abuse JPEG “everywhere”, just abuse LibJPEG.



JPEG signature — Chunk marker - ff e5 in block 1 - ff e6 in block 2 — Chunk length - c4 00 in block 1 - e4 00 in block 2

```
00000: ff d8 ff e? ?4 00 39 54 ?? 6d 04 2e ?? b7 b2 ??
       ?? 08 cf ?? ?? 46 d4 ?? ?? 0a 05 ?? ?? cb e2 ??  (contains no 0xff)
       ?? 87 fc ?? 38 98 83 ?? ?? 32 ac ?? ?? 6a a8 ??
       ?? 43 1f ?? ?? 66 87 f5 ?? 85 f7 ?? ?? 1c a9 ??

0c404: ff fe b5 e9      <COMment chunk covering Image 1>
0e404: ff e0            <start of Image 1>
       ...
       ff d9            <end of Image 1> <end of comment>
179ed: ff e0            <start of Image 2>
1b0d7: ff d9            <end of Image 2>
```

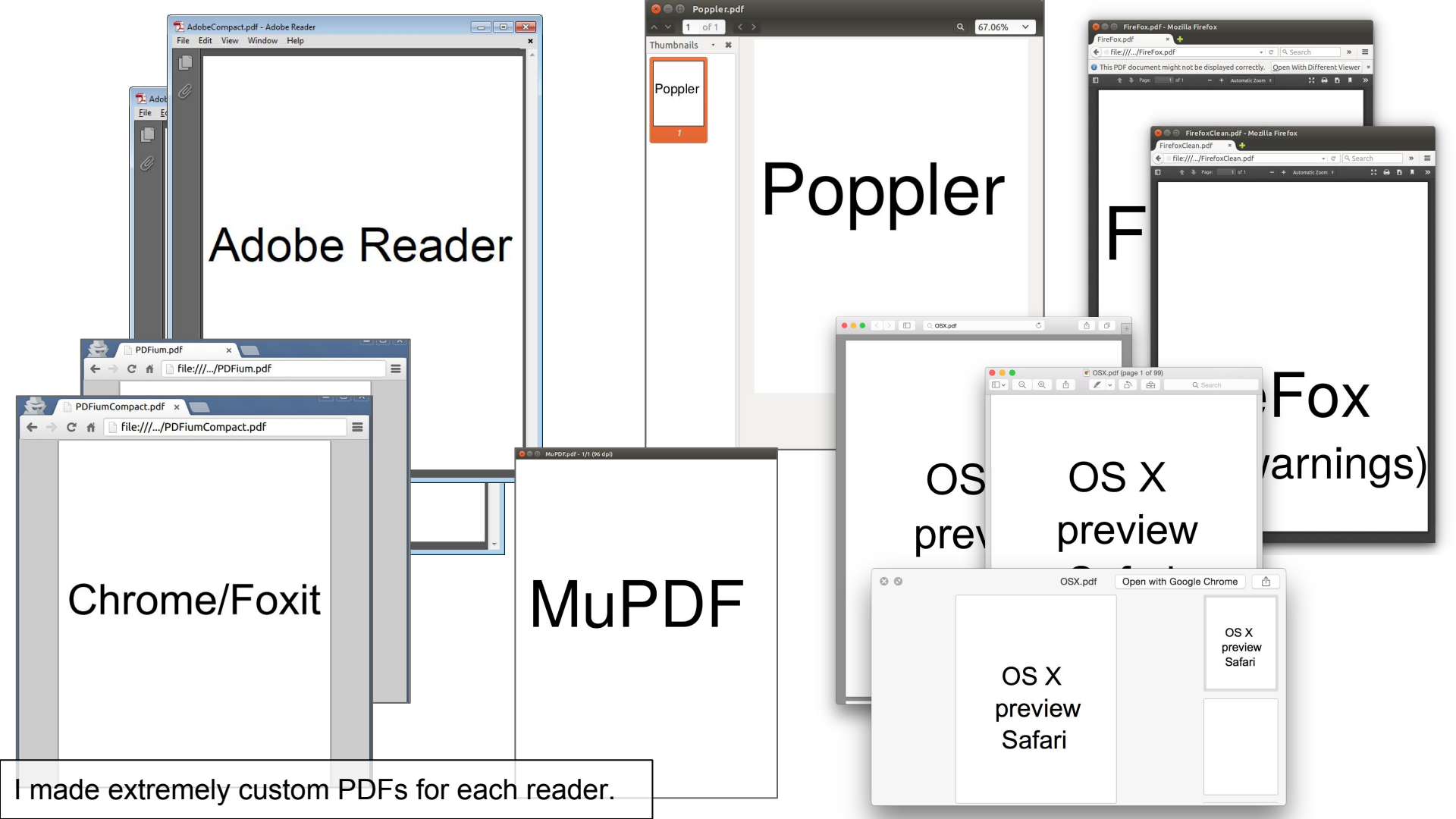
JPEG format’s landscape

in practice, JPEG is LibJPEG turbo v6

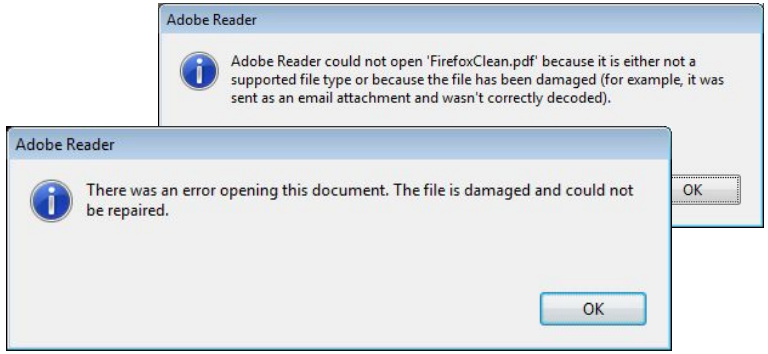
- de facto standard
  - later versions not used (different API)

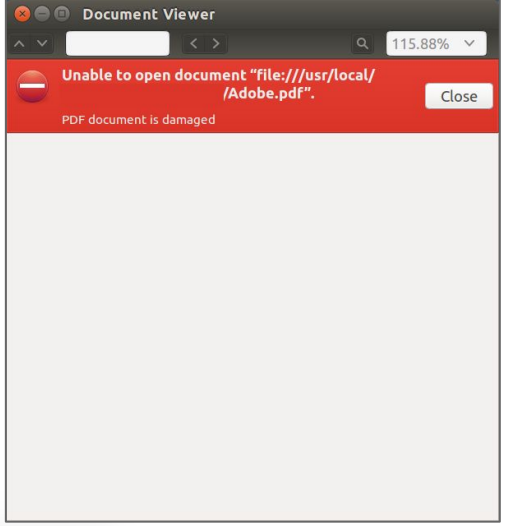
Even if you create your own JPEG library, you want to have full LibJPEG compatibility.

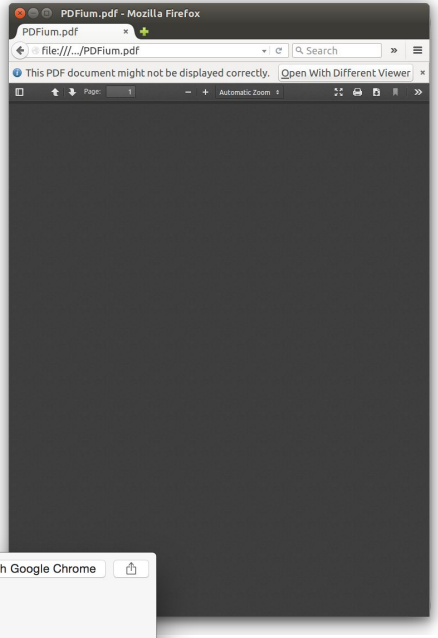
JPEG format is defined by LibJPEG.

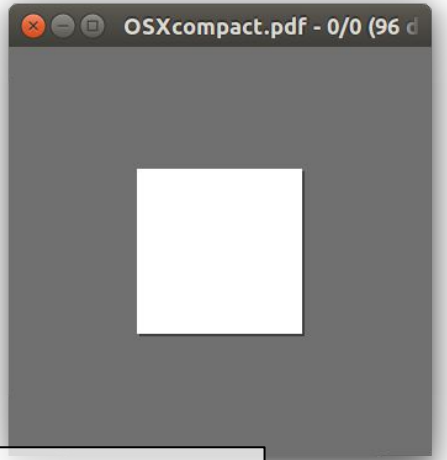


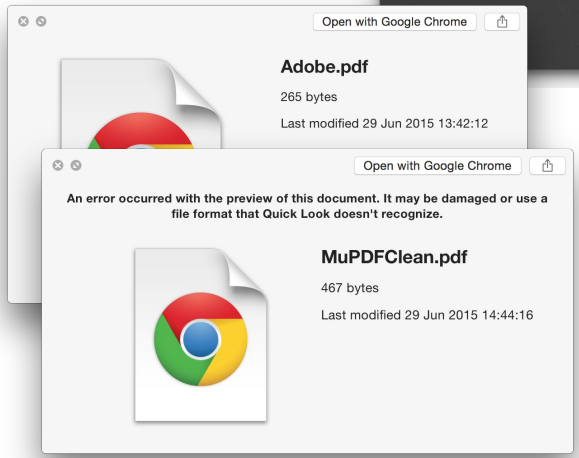
I made extremely custom PDFs for each reader.











These "extreme" PDFs fail on any other reader.

PDF’s current landscape

PDF: 6 interpretations of the specs

- specs are even more useless

One good open library: a unified attack surface

Fuzz it, pwn everyone ?
True, but also fixed for everyone!

Is diversity really good?
We’re all supposed to use the same file format.

Diversity is good?

Attack surface is worse.
Unofficial substandards.

In any cases...

Specs are merely an introduction guide.
A free set of examples w/ corner cases.
A grammar ?

PDF’s future

PDF/E (engineer): 3d crap
PDF/A (archiving): already 8 flavours

Specs:

- specs are now commercial
- the main implementation is not open
- no set of free files.

And all countries preserve their culture with that format?!?!

We’re waiting for a new disaster...

many file formats are abandoned

One specs. then nothing.

It’s like knowing about someone only from a baby’s picture.

<!--

PoC||GTFO 11

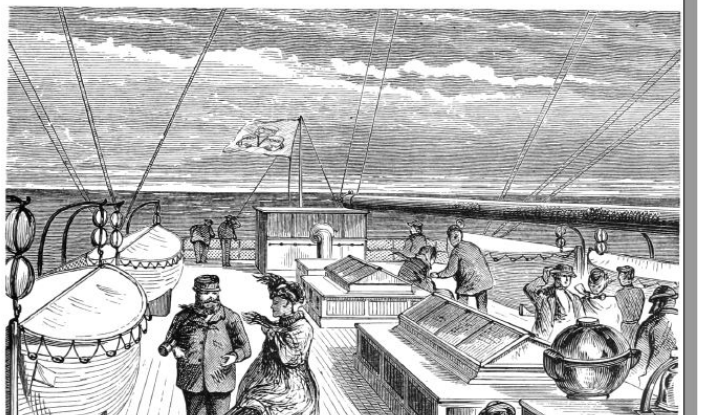
```
$ruby pocorgtfo11.pdf
Listening for connections on port 8080.
To listen on a different port,
re-run with the desired port as a command-li

A neighbor at 127.0.0.1 is requesting /
A neighbor at 127.0.0.1 is requesting /ajax/
A neighbor at 127.0.0.1 is requesting /favic
```

pocorgtfo11.pdf

Open | 1 / 37 | 66% | Tools

PoC||GTFO



PoC||GTFO Issue 0x11

127.0.0.1:8080

International Journal of PoC||GTFO Issue 0x11

IN A FIT OF STUBBORN OPTIMISM,
PASTOR MANUL LAPHROAIG
AND HIS CLEVER CREW
SET SAIL TOWARD
WELCOMING SHORES OF
THE GREAT UNKNOWN!

Click here to download the PDF!

11.1 Please Stand and Be Seated
11.2 In Praise of Junk Hacking
11.3 Emulating Star Wars on a Vector Display
11.4 Tron in 512 Bytes
11.5 Defeating the E7 Protection
11.6 Phrasebook for ARM Cortex M
11.7 Ghetto CFI for x86
11.8 Tourist's Guide to the MSP430
11.9 This PDF is a Webserver
11.10 In Memoriam: Ben "bushing" Byer

This is an HTML/Ruby/PDF/ZIP polyglot. When interpreted by Ruby, it acts as a web server that serves this page. If the URL requested of the webserver matches a path inside the ZIP, that file is served. If loaded directly in a web browser, the file will render as this webpage, too, however, the link to the PDF download is hidden.

Feelies

- index.txt — a text version of this feelies index
- issues.txt — about all issues of PoC||GTFO
- issues.bib — a BibTeX file containing references for all issues of PoC||GTFO
- batteryfirmware.pdf — Battery Firmware Hacking - Charlie Miller
- vst.tar.bz2 — v.st vector board sources
- vectormame.diff — diff for Mame (see Star Wars article)

-->

PoC||GTFO 11 is self-aware:
a PDF that serves itself (HTTP quine),
parses its own ZIP to serve its archived feelies.

Important question

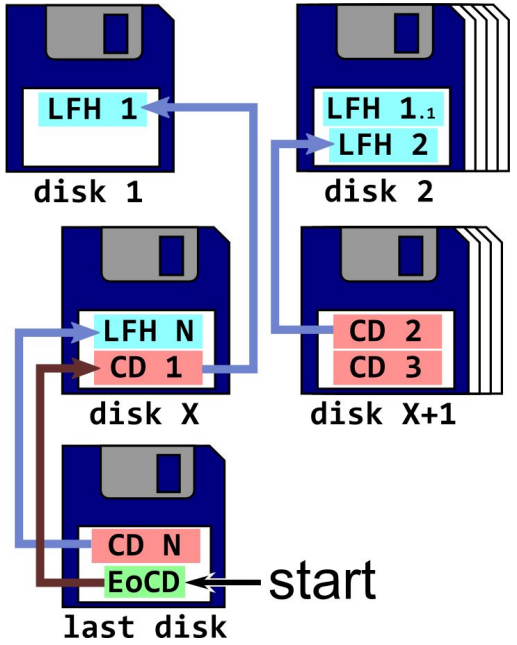
Do you still sleep with a teddy bear?

Kids really deprecate stuff

Our computers still handle always more
and more file formats.
⇒ The attack surface just keeps growing.

Obsolete formats are still omnipresent

Formats, sub-formats, features...

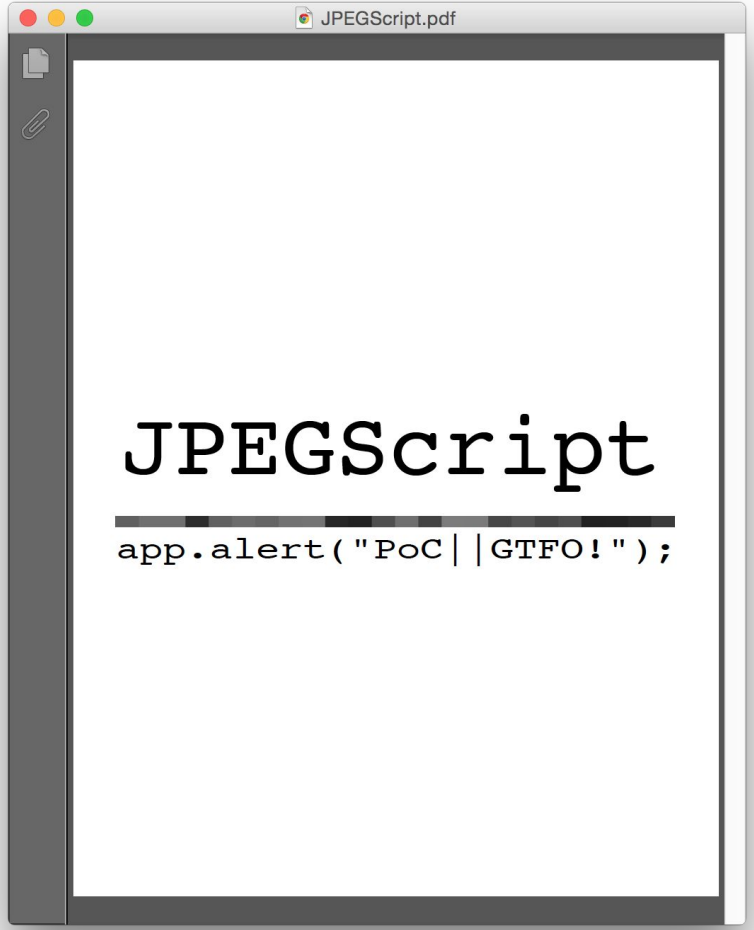


Because it’s unclear if we can go back.

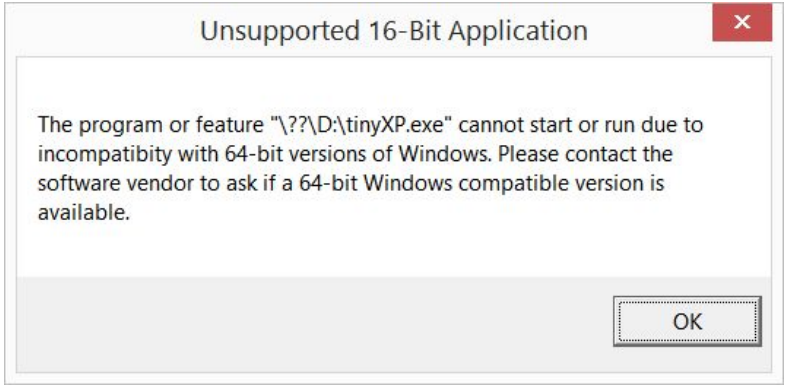
We’d be too afraid to deprecate them.

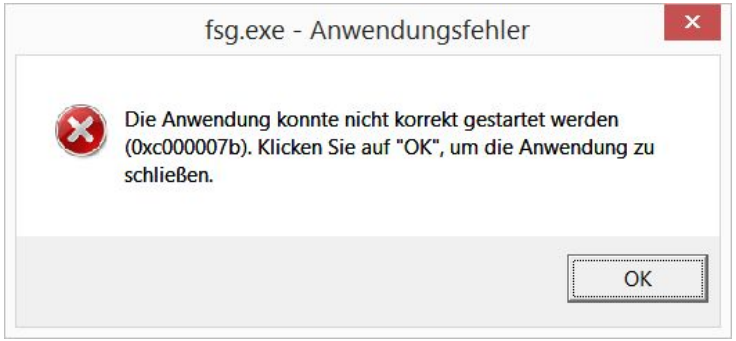
Yet we deprecate for security.

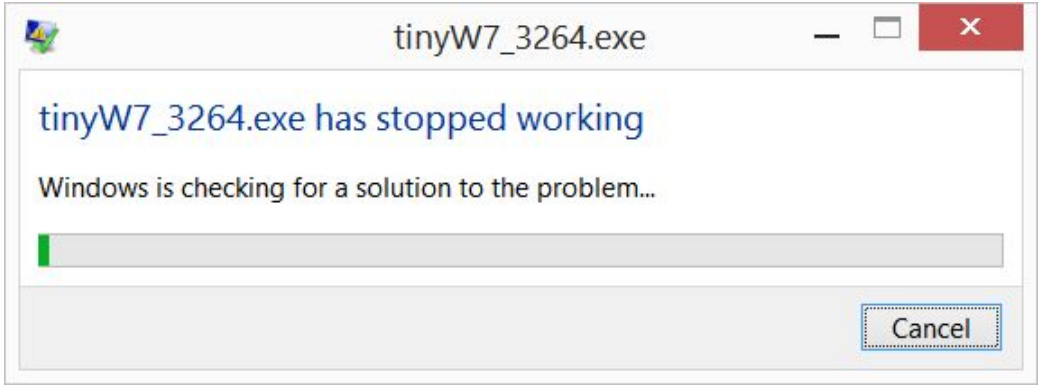
Example for PDF:
JPEG-compressed text
is not supported anymore
(it could bypass security).



Windows PE format becomes stricter (deprecates packers)







Sometimes,
it's not even for
security reasons

For example,
EPUB 3.1 suddenly killed
backward compatibility.

EPUB 3.1 or *EPUB* 4.0?

By now it should be clear that the aggressive removal of features in *EPUB* 3.1 would have some far-reaching consequences. This is particularly true for the removal of the *NCX*, which would make *EPUB* 3.1 files incompatible with many existing E-ink readers. It would do this by ruling out the option to make backward-compatible "hybrid" files.

As Gary McGath pointed out earlier, introducing "radical changes" in what is essentially a minor version is pretty unusual practice for any standard. Nowadays, most software and file formats use some variation of semantic versioning, with version numbers that follow the general form *MAJOR.MINOR.PATCH*. Here, each component of the version number has a well-defined meaning:

1. *MAJOR* version is increased in case of incompatible API changes,
2. *MINOR* version is increased when functionality is added in a backwards-compatible manner, and
3. *PATCH* version is increased in case of backwards-compatible bug fixes.

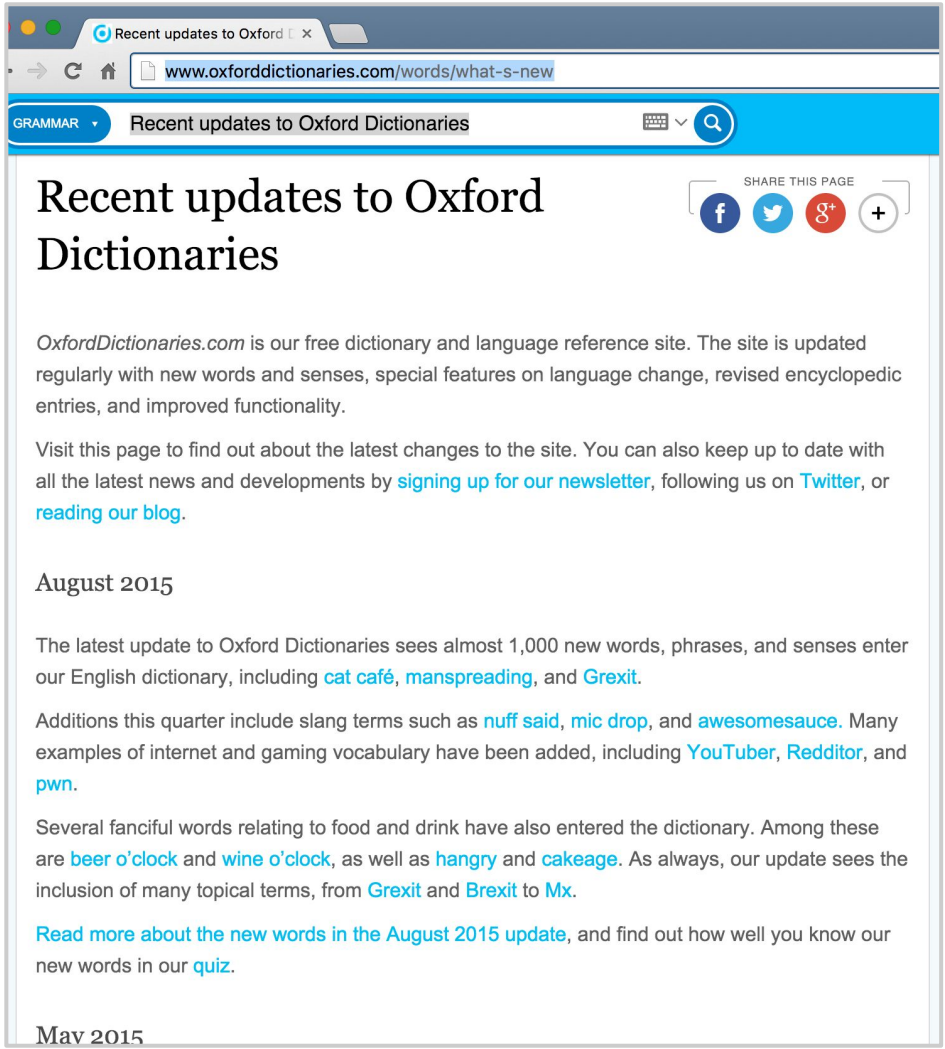
Since the current draft includes multiple backward-incompatible changes, this makes

http://blog.kbresearch.nl/2016/03/10/the-future-of-epub-a-first-look-at-the-epub-3-1-editors-draft/

We don’t need new file formats.

It’s the same problem again if eventually their specs stop reflecting reality.

Even dictionaries have regular updates, to reflect reality.



Story time

Digipres = PDF worshippers. 150 years of availability?

- Non free specs + closed source software?

Here comes the grim reaper:

- Fix your stuff or it will be killed (like Flash)

We store our knowledge. What about files born digital?

Not infosec, but worrying.

veraPDF Corpus

The repository contains the veraPDF test corpus for PDF/A specifications (Versions 1B, 1A, 2B, 2U, 2A, 3B, 3U, 3A) as well as a number of additional tests files for ISO 32000-1.

This test suite complements Isartor and Bavaria test suites and follows their test file pattern:

- all test files are atomic;
- they are self-documented via the document outlines; and
- the naming pattern and the directory structure indicate relevant parts of ISO 19005-1 specification

veraPDF

© 2015 veraPDF Consortium

This work is licensed under a Creative Commons Attribution 4.0 International (CC BY 4.0)

veraPDF-corpus / PDF_A-1b / 6.

DmitryRemezow Rename 6-1-12-t0

..

veraPDF test suite 6-1-12-t01-fail-a.pd

veraPDF test suite 6-1-12-t02-fail-a.pd

veraPDF test suite 6-1-12-t02-fail-b.pd

veraPDF test suite 6-1-12-t02-fail-c.pd

veraPDF test suite 6-1-12-t02-fail-d.pd

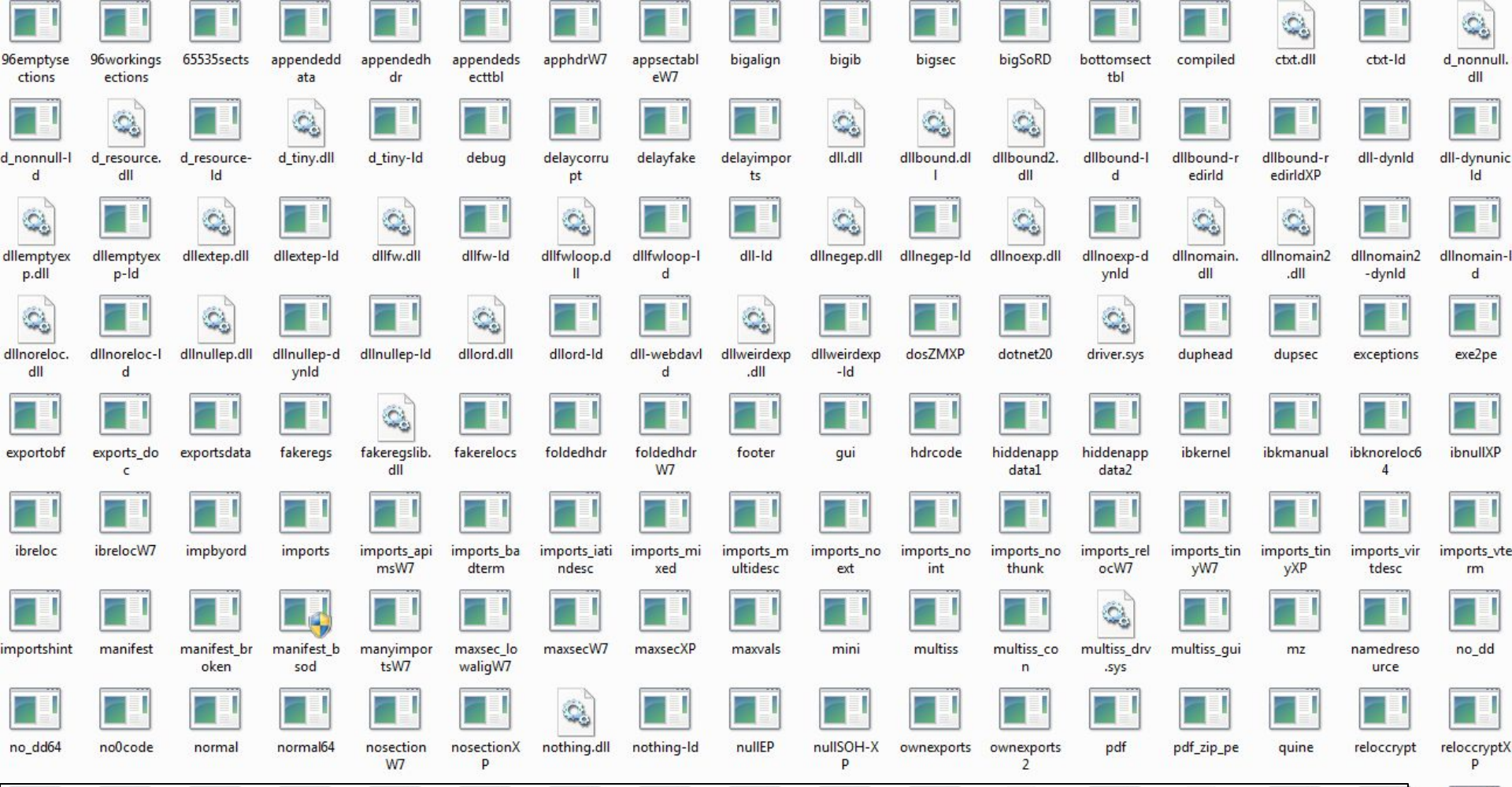
veraPDF test suite 6-1-12-t02-fail-e.pd

veraPDF test suite 6-1-12-t02-fail-f.pd

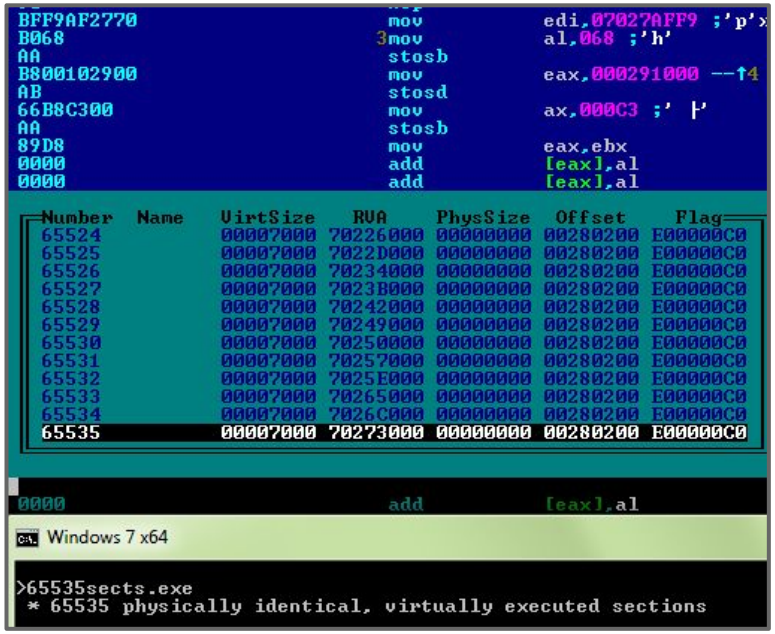
veraPDF test suite 6-1-12-t02-fail-l.pd

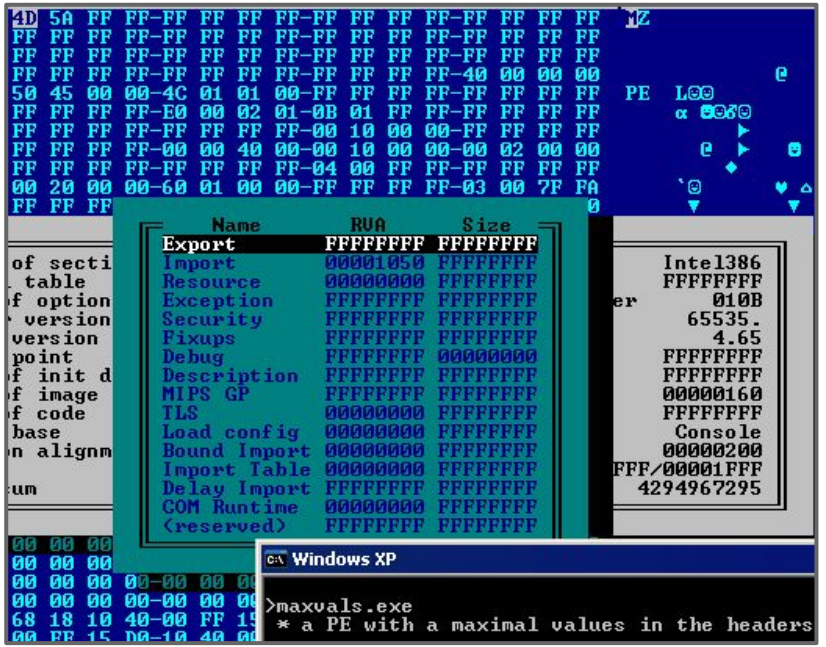
veraPDF test suite 6-1-12-t02-pass-g

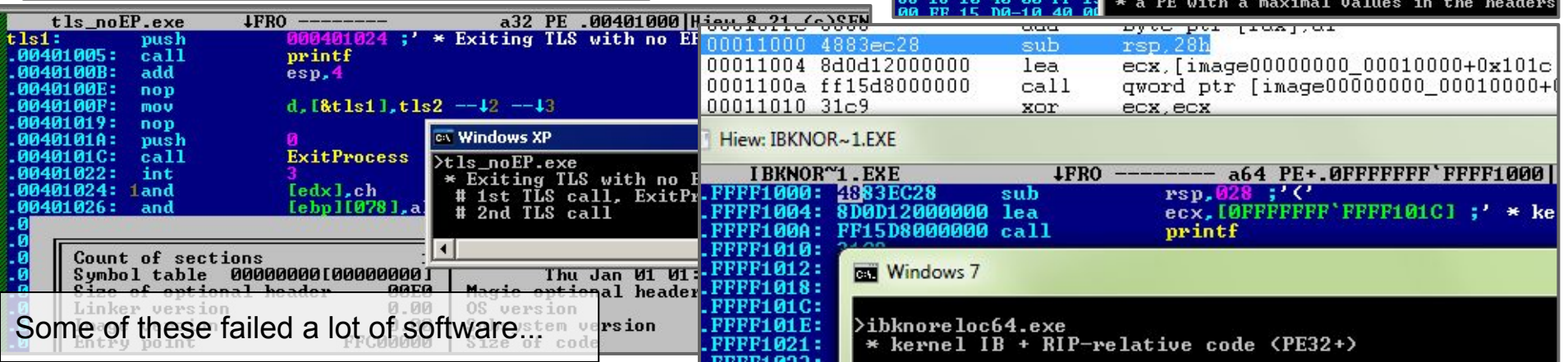
veraPDF and its test files:
a great initiative.



PE.corkami.com: my own collection of hand-made executables and "documentation" (completely free).







Some of these failed a lot of software...

Consequence of my PE page+corpus

- 'corkami-proof' software
- raises the bar for everyone
- become a hub of knowledge
  - "I can't share the sample", but from the knowledge, my own file will be shared
    ⇒ even useful for the original contact

Conclusion

Attack surface

Too many (sub)formats

Too many parsers (= no good open lib)

Specs

Specs shouldn’t be a religious text

- Worshipped, but outdated and worthless

Specs should reflect reality (a law)

- updated, enforced, realistic, freely available

A good open lib

Deprecation

Deprecation is a natural cycle, and yet...

We are afraid to deprecate because
no file format is fully preserved:

- open, up to date specs
- free test coverage

But it won’t happen...

...until a great disaster ?

It ends up on CNN, with a logo & a website :)

Ack

Phil Fabrice Travis Sergey Micah Kurt QKumba Hanno...

Thank you!

@angealbertini

corkami.com

CC BY