

# Encryption with PDF 2.0

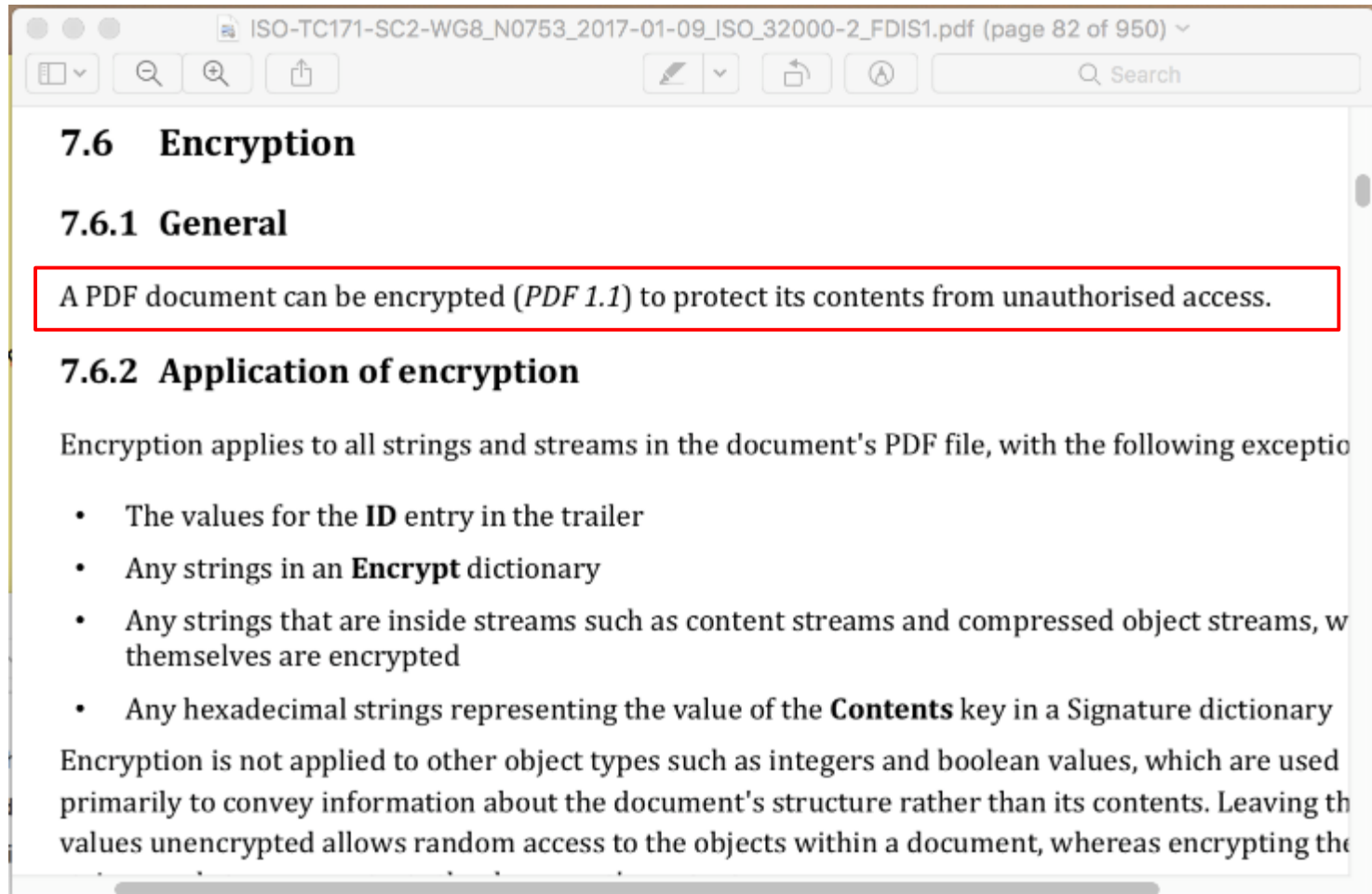
Roman Toda, Normex

**NORMEX**

Roman Toda  
CTO, Normex

[www.digitaldocuments.org](http://www.digitaldocuments.org)





ISO-TC171-SC2-WG8\_N0753\_2017-01-09\_ISO\_32000-2\_FDIS1.pdf (page 82 of 950) ✓

7.6 Encryption

7.6.1 General

A PDF document can be encrypted (*PDF 1.1*) to protect its contents from unauthorised access.

7.6.2 Application of encryption

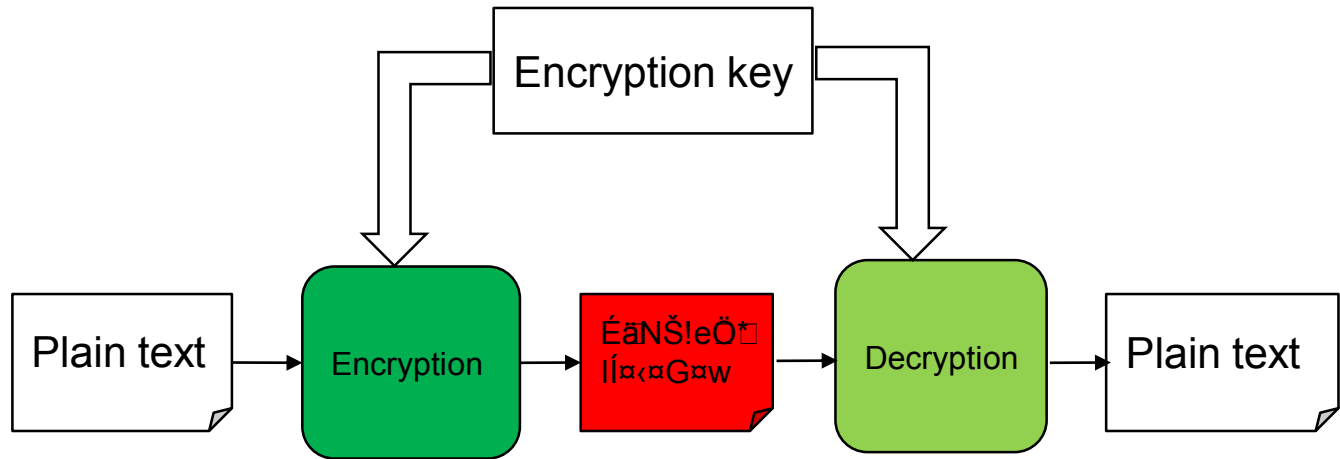
Encryption applies to all strings and streams in the document's PDF file, with the following exceptions:

- The values for the **ID** entry in the trailer
- Any strings in an **Encrypt** dictionary
- Any strings that are inside streams such as content streams and compressed object streams, which themselves are encrypted
- Any hexadecimal strings representing the value of the **Contents** key in a Signature dictionary

Encryption is not applied 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 contents. Leaving these values unencrypted allows random access to the objects within a document, whereas encrypting the



Uses symmetric-key algorithm to encrypt strings and streams in PDF file



## Encryption – general

---

- What it does: Encrypting all Strings and Streams
  - Few exceptions: Encryption dictionary itself, file IDs
  - Special cases: metadata, external files, Crypt filters (1.5), Unencrypted wrapper (2.0)
- What it doesn't: Encrypting parts of the content, Reliable access restrictions
- How: Encrypt dictionary in trailer
  - How to compute the Encryption key
  - Which symmetric-key algorithm to use (RC, AES)
  - /Filter = “Security handler” – additional behavior of authorization

**NORMEX**

Roman Toda  
CTO, Normex

www.digitaldocuments.org



- *V* entry in Encryption dictionary

<i>V</i>	number	(Required) A code specifying the algorithm to be used in encrypting and decrypting the document.
	0	An algorithm that is undocumented. This value shall not be used.
	1	<i>(Deprecated)</i> Indicates the use of 7.6.3.1, "Algorithm 1: Encryption of data using the RC4 or AES algorithms" (deprecated) with a file encryption key length of 40 bits; see below.
	2	<i>(PDF 1.4; deprecated)</i> Indicates the use of 7.6.3.1, "Algorithm 1: Encryption of data using the RC4 or AES algorithms" (deprecated) but permitting file encryption key lengths greater than 40 bits.
	3	<i>(PDF 1.4; deprecated)</i> An unpublished algorithm that permits file encryption key lengths ranging from 40 to 128 bits. This value shall not appear in a conforming PDF file.
	4	<i>(PDF 1.5; deprecated)</i> The security handler defines the use of encryption and decryption in the document, using the rules specified by the <b>CF</b> , <b>StmF</b> , and <b>StrF</b> entries using 7.6.3.1, "Algorithm 1: Encryption of data using the RC4 or AES algorithms" (deprecated) with a file encryption key length of 128 bits.
	5	<i>(PDF 2.0)</i> The security handler defines the use of encryption and decryption in the document, using the rules specified by the <b>CF</b> , <b>StmF</b> , <b>StrF</b> and <b>EFF</b> entries using 7.6.3.2, "Algorithm 1.A: Encryption of data using the AES algorithms" with a file encryption key length of 256 bits.

1.7



- *V* entry in Encryption dictionary

<i>V</i>	number	(Required) A code specifying the algorithm to be used in encrypting and decrypting the document.
	0	An algorithm that is undocumented. This value shall not be used.
	1	<i>(Deprecated)</i> Indicates the use of 7.6.3.1, "Algorithm 1: Encryption of data using the RC4 or AES algorithms" (deprecated) with a file encryption key length of 40 bits; see below.
	2	<i>(PDF 1.4; deprecated)</i> Indicates the use of 7.6.3.1, "Algorithm 1: Encryption of data using the RC4 or AES algorithms" (deprecated) but permitting file encryption key lengths greater than 40 bits.
	3	<i>(PDF 1.4; deprecated)</i> An unpublished algorithm that permits file encryption key lengths ranging from 40 to 128 bits. This value shall not appear in a conforming PDF file.
	4	<i>(PDF 1.5; deprecated)</i> The security handler defines the use of encryption and decryption in the document, using the rules specified by the <b>CF</b> , <b>StmF</b> , and <b>StrF</b> entries using 7.6.3.1, "Algorithm 1: Encryption of data using the RC4 or AES algorithms" (deprecated) with a file encryption key length of 128 bits.
	5	<i>(PDF 2.0)</i> The security handler defines the use of encryption and decryption in the document, using the rules specified by the <b>CF</b> , <b>StmF</b> , <b>StrF</b> and <b>EFF</b> entries using 7.6.3.2, "Algorithm 1.A: Encryption of data using the AES algorithms" with a file encryption key length of 256 bits.

2.0

- Deprecations. Because it is unsecure !!



- Can I use old (V<5) in 2.0 files?
  - **Don't even think about it !**
  - Not because it's deprecated but because it is not secure
- When developing 2.0 consumer/viewer. Should I care about V<5 files?
  - **Yes !**
  - PDF has been developed with backward compatibility in mind
  - There is a lot of old pdf files you still want to open
- Can I use 256 AES Encryption (V=5) in 1.7 files?
  - **Maybe !**
  - Everybody does that
  - Nobody really cares about file header (%PDF-)
  - But there is a perfectly legal way



- In 06/2008 Adobe published **Adobe® Supplement to the ISO 32000 BaseVersion: 1.7 ExtensionLevel: 3** with the AES256 algorithm and companies adopted that quickly
- Fun fact: in 12/2008 hackers found weaknesses in the algorithm
- Adobe published revision (**ExtensionLevel 8**) and that is in current ISO 32000-2
- ISO 32000-1:2008 (1.7) chapter **7.12 Extensions Dictionary**
- In root (catalog)  
  /Extensions <<  
    /ADBE<<  
      /BaseVersion /1.7  
      /ExtensionLevel 8  
    >>  
  >>





- Powerful language
  - Not all strings and stream could be encrypted with the same key and the same algorithm
  - /StmF – how do we encrypt/decrypt streams
  - /StrF – how do we encrypt/decrypt strings
- Each stream could have a Crypt Filter (like FlateDecode) that says how this specific stream is encrypted (or even left untouched = Identity)
- Consequence: We don't have just one Encryption key for whole document, but an array
- Reality in 2.0:
  - only Identity and StdCF could be used in password security
  - DefaultCryptFilter or DefEmbeddedFile with public-key encryption



- /Filter /Standard % aka password protection
  - User password (open password)
    - Can't open document without entering it
  - Owner password (master password, change security password)
    - Only used to change security
  - Permissions
    - Print, Modify content, Copy/Extract, Annotations, Forms, Assemble
    - **2.0** deprecated bit 10 that controlled the extraction for accessibility purposes
  - EncryptMetadata
    - Even encrypted files can be parsed for metadata, stored in archives, searched
- Consequences:
  - Doesn't make sense to set permission if there is no Owner password set
  - If User password is not set, document could be decrypted without any restrictions.
  - It is up to PDF Viewer to respect permissions once the file is decrypted
  - Once password is sent, you can't take it back



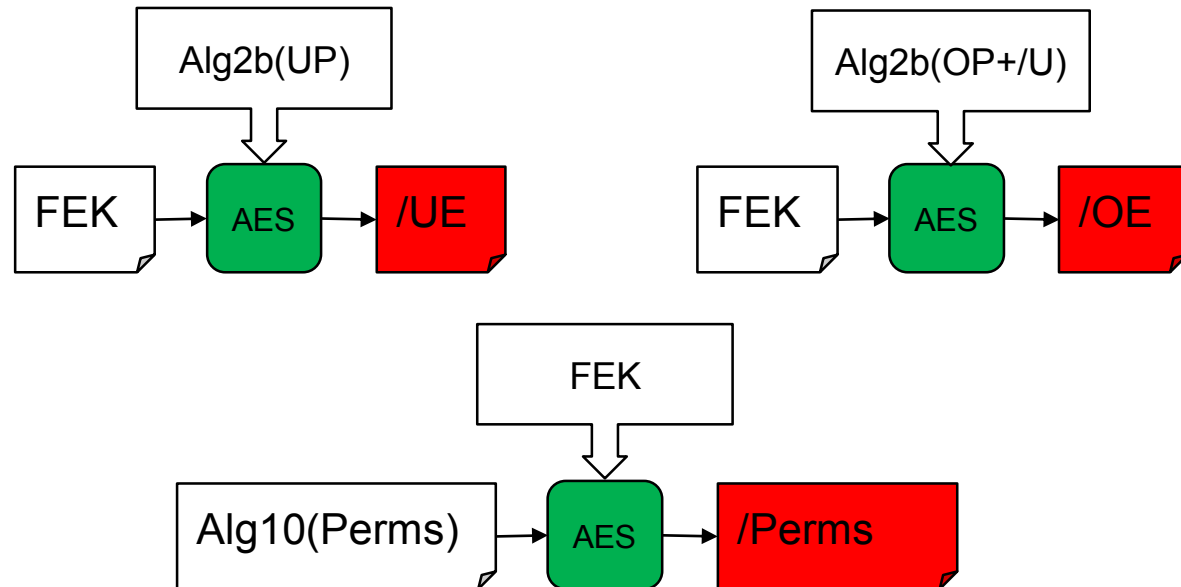
## Standard Security handler – technical description

File Encryption Key (FEK) = random value,

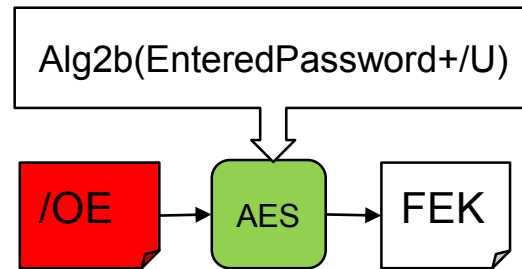
UP = user password, OP = owner password

Alg8: /U = Alg2b(UP)

Alg9: /O = Alg2b(OP+/U)

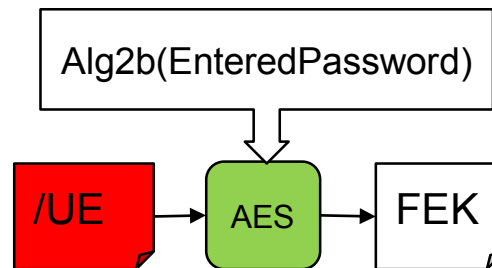


if ( Alg2b(EnteredPassword+/U) == /O)



else

if (Alg2b(EnteredPassword) == /U)

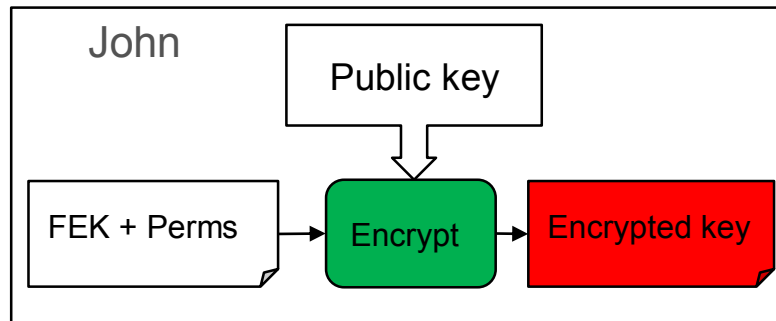


else alert("Invalid password")



- /Filter /Adobe.PubSec /SubFilter /adbe.pkcs7.s5
  - Basic idea is to encrypt using recipient's public key so only the owner of Private key can access the document
  - We may have more recipients each group with different permissions
  - As with passwords crypt filters may be used

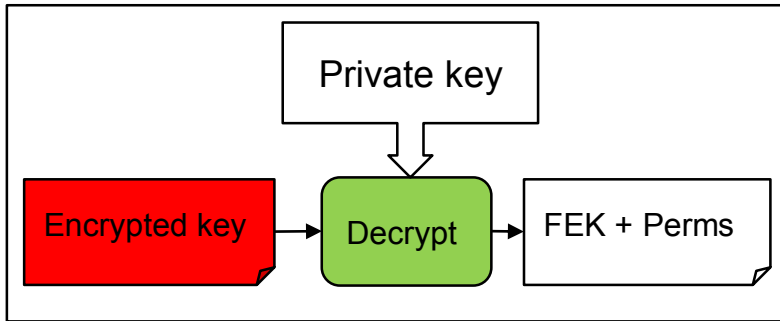
File Encryption Key (FEK) = random value



- /Recipients [ EncryptedKey1 EncryptedKey2 ]
  - one Encrypted key for each recipient
  - CMS form



- Use private key
  - go through /Recipients
  - If possible to decrypt the EncryptedKey



- Status in ISO 32000-2
  - No update, no deprecations. FEK is generated from 24bytes (seed+perms) vs. 32bytes in AES
  - Allowing all pkcs7 (CMS) certificates without restrictions
  - No new methods like: EC (Elliptic curves) set of algorithms although they are endorsed since 2009



- /Filter /YourOwnFilter
  - You'd need to specify how encryption key is calculated
  - Code has to be added to any PDF viewer (Acrobat allows plug ins)
  - Allows control encryption and use, overwriting of standard behavior

```
trailer
```

```
<</Encrypt 37 0 R
```

```
/ID [<7D05F66A42D2F9428C28D2D87853FF06> <3F57033B47A1D2791CDD042DBDC881C4>]
```

```
/Root 11 0 R
```

```
/Info 9 0 R
```

```
/Size 38
```

```
>>
```



- /Filter /YourOwnFilter
  - You'd need to specify how encryption key is calculated
  - Code has to be added to any PDF viewer (Acrobat allows plug ins)
  - Allows control encryption and use, overwriting of standard behavior

```
trailer
<</Encrypt 37 0 R
/ID [<7D05F66A42D2F9428C28D2D87853FF06> <3F57033B47A1D2791CDD042DBDC881C4>]
/Root 11 0 R
/Info 37 0 obj
/Size <</Filter /MicrosoftIRMServices
>>
/MicrosoftIRMVersion 1
/PublishingLicense (eJztvdeW5FaWJTifEiv6EcsTWuVishtaGcyg1UsvaIMWBsAA9Oov
>>
```





- /Filter /YourOwnFilter
  - You'd need to specify how encryption key is calculated
  - Code has to be added to any PDF viewer (Acrobat allows plug ins)
  - Allows control encryption and use, overwriting of standard behavior

```
trailer
<</Encrypt 37 0 R
/ID [<7D05F66A42D2F9428C28D2D87853FF06> <3F57033B47A1D2791CDD042DBDC881C4>]
/Root 11 0 obj
/Info 37 0 obj
/Size <</Filter /MicrosoftIRMServices
/MicrosoftIRMServices
>>
>>
/PublicKey <</Filter/FOPN_foweb
>>
/V 2
/Length 128
/VEID(9.0)
/BUILD(907)
/SVID(fKIvIE9yJBKlvOoZIF8HOTgOhf_dwA01-UCCD5tivCJJFENDhmMIPPnrM3mMjnz)
/DUID(DzhGAIIdoDen09NJ6gBE1Uk-yUjwF3B1IAROVH1.dOaMFGeBu7-6pBMhHF6U0Ku)
/INFO(Hgr50GSLkqXShHKestPel1/ocyos1BDzOQxbboligGDzJg3a0iTM9nsUYTCH8yDE+S
>>
```



- /Filter /YourOwnFilter
  - You'd need to specify how encryption key is calculated
  - Code has to be added to any PDF viewer (Acrobat allows plug ins)
  - Allows control encryption and use, overwriting of standard behavior

```
trailer
<</Encrypt 37 0 R
/ID [<7D05F66A42D2F9428C28D2D87853FF06> <3F57033B47A1D2791CDD042DBDC881C4>]
/Root 11 0 obj
/Info 37 0 obj
/Size <</Filter /MicrosoftIRMServices
/Size
>>
/Info <</Filter /MicrosoftIRMServices
/Info 37 0 obj
/Info <</Filter /FOPN_foweb
>>
>>
37 0 obj
<</Filter/FoxitConnectedPDFDRM
/ConnectedPDF <<
  /DocumentID <<
    /Type /DocumentID
    /URI (https://cws.connectedpdf.com/connected-pdf?foxit/windows/cAppID/cDocID/7F38E61)
  >>
  >>
>>
endobj
```



- /Filter /YourOwnFilter
  - You'd need to specify how encryption key is calculated
  - Code has to be added to any PDF viewer (Acrobat allows plug ins)
  - Allows control encryption and use, overwriting of standard behavior

```
trailer
<</Encrypt 37 0 R
/ID [<7D05F66A42D2F9428C28D2D87853FF06> <3F57033B47A1D2791CDD042DBDC881C4>]
/Root 11 0 obj
/Info 37 0 obj
/Size <</Filter /MicrosoftIRMServices
/Info <<-----
/Size <<-----
>> /Micro 37 0 obj
/Publi 37 0 obj
37 0 obj
37 << /Filter /EBX_HANDLER
<</
/EBX_AUTHOR (Roman Toda\\ (roman.toda@gmail.com\\))
/Co /Length 128
/ADEPT_ID (urn:uuid:7fbdd8ef-4a7d-4162-a89f-9c2732cd141f)
/V 4
/EBX_TITLE (My book_Adobe)
/EBX_BOOKID (urn:uuid:7fbdd8ef-4a7d-4162-a89f-9c2732cd141f)
/ADEPT_LICENSE (rVdbz6LKE n0/v2Iyr8bhIih8cdxp7iiooKj4BjRXEZE7/vrT6PfNuGcnJycnh6QTrK5
>> >>
end endobj
```



## Unencrypted wrapper

---

- What happens if reader doesn't have specific code?
- Author doesn't control what message end user gets when encryption is unsupported
- In 2012 Microsoft Supplement when introducing IRM protection introduced /Wrapper - similar to incremental update
- Basically we need 2 files
  - Message for end user in unencrypted form
  - Encrypted file
  - Reader app has to decide which one to show

**NORMEX**

Roman Toda  
CTO, Normex

www.digitaldocuments.org



- How: using Collection (portfolios) with few additions
  1. Define Associate file with File Specification dictionary
    - /EF = the encrypted file and in 17 0 obj %File Specification Dictionary
    - /AFRelationship /EncryptedPayload
    - /EP << /Type /EncryptedPayload  
/Subtype /NameOfYourCryptHandler  
/Version /1.0 >>
  2. Define /Collection with
    - /D (encrypted\_payload.pdf)
    - /View /H
- Consequences:
  - App that doesn't recognize Collections (AA7) shows default file
  - App that knows Collections and doesn't recognize Unencrypted wrapper (AA9) will try to show the Encrypted payload



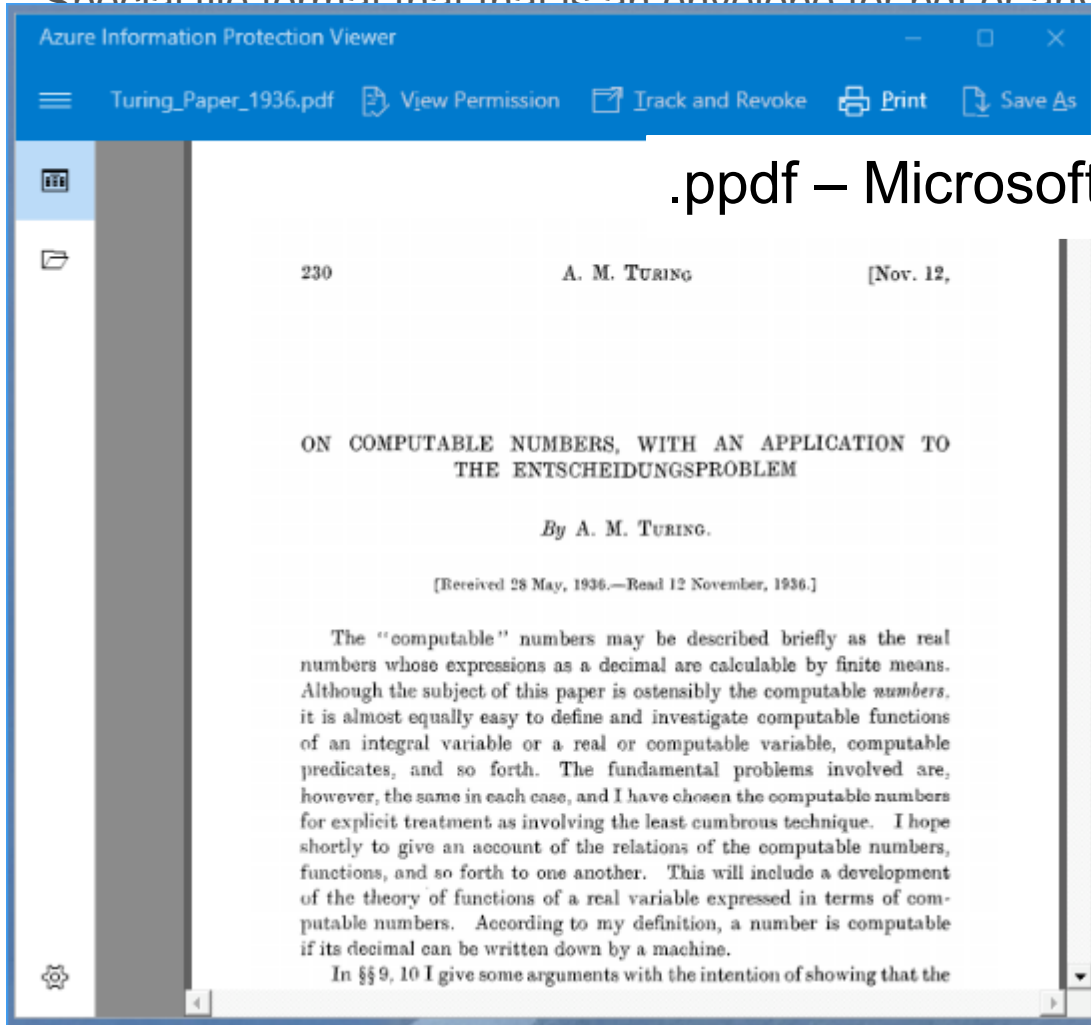
- Special file format that that is an envelope for pdf or anything else
  - Requires special viewer
  - .pdc .drmz .ppdf .drmX ...
  - allows complete control over content, permissions, tracking, revocation etc.



- Special file format that that is an envelope for pdf or anything else

ing, revocation etc.

## .ppdf – Microsoft Azure protection

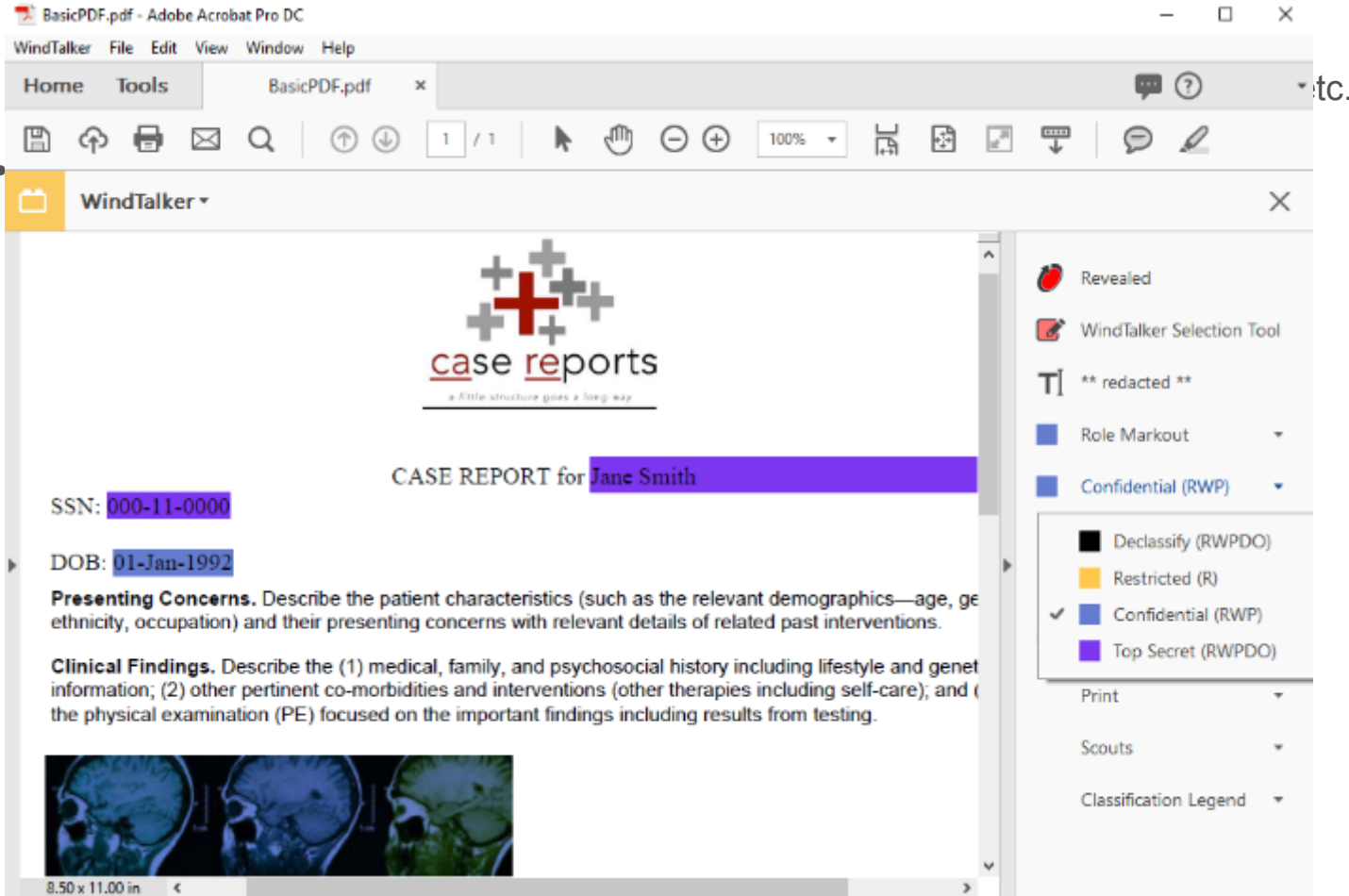


- Special file format that that is an envelope for pdf or anything else
  - Requires special viewer
  - .pdc .drmz .ppdf .drmX ...
  - allows complete control over content, permissions, tracking, revocation etc.
  
- Show/decrypt only content user is allowed to see



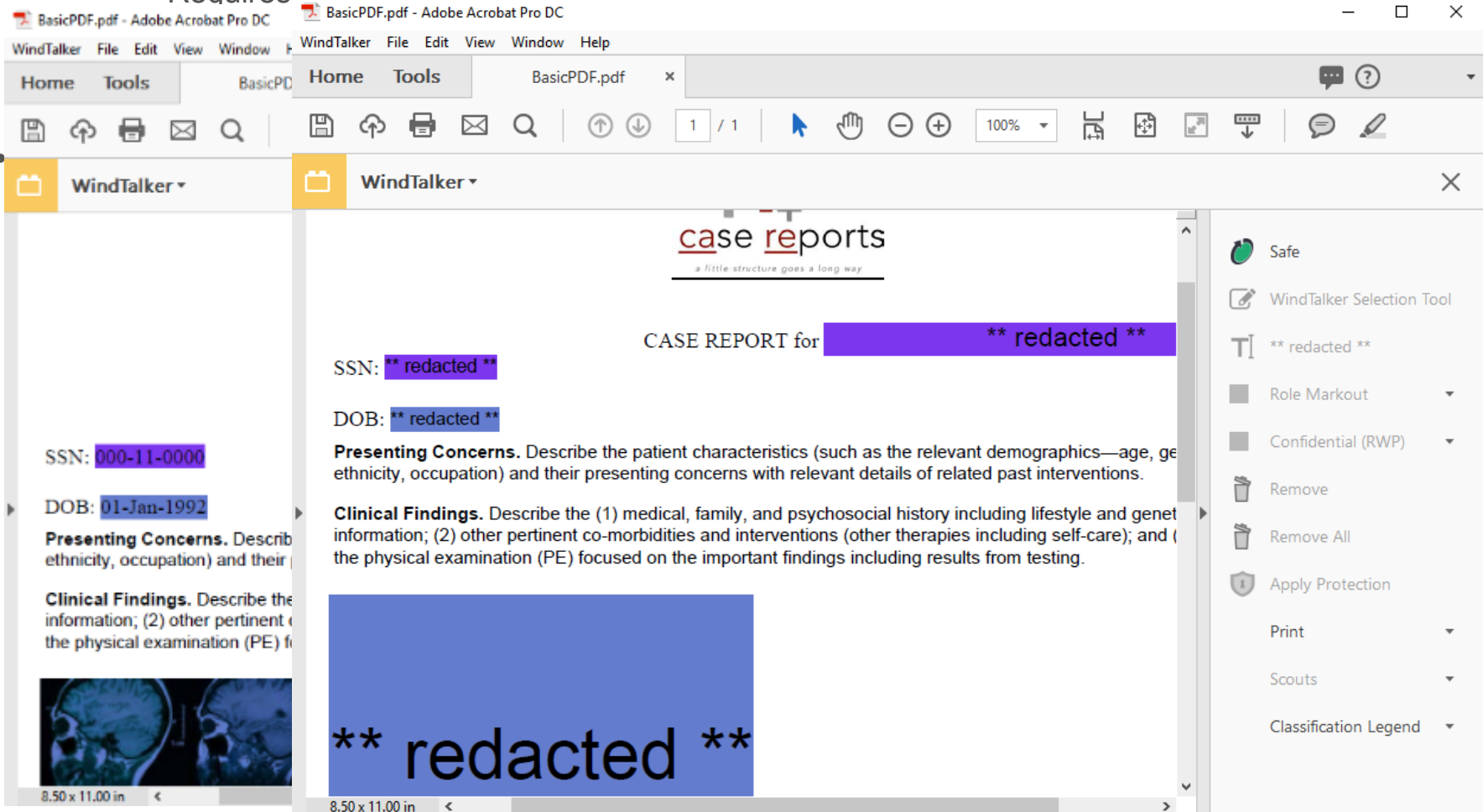


- Special file format that that is an envelope for pdf or anything else
  - Requires special viewer



- Special file format that that is an envelope for pdf or anything else

- Requires special viewer



# Thank you

# Questions ?

**NORMEX**

Roman Toda  
CTO, Normex  
[www.digitaldocuments.org](http://www.digitaldocuments.org)

