SECCON

Backdooring MS Office documents with secret master keys

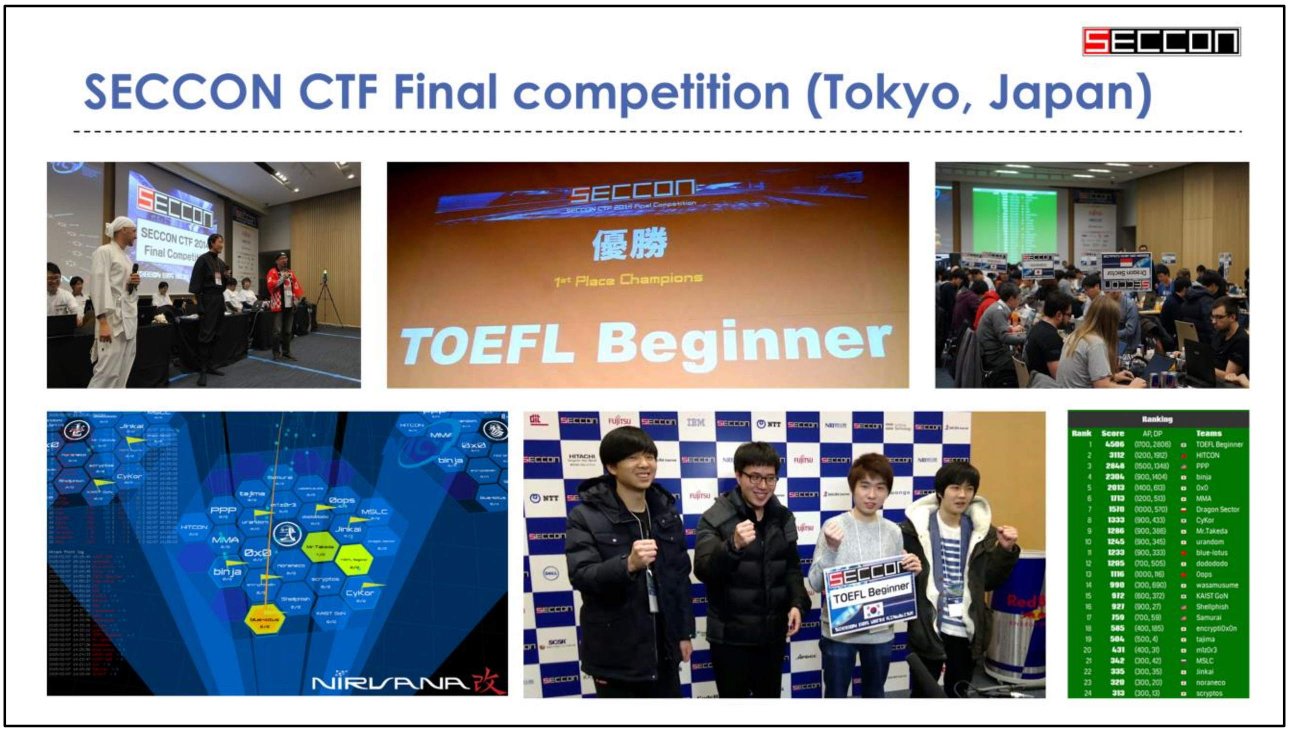
Yoshinori Takesako (SECCON),
Shigeo Mitsunari (Cybozu Labs)

My name is Yoshinori Takesako. It's very long name. so call me "Yoshi". I came from Tokyo, Japan.

Yoshinori Takesako (SECCON)

- Twitter: @takesako
- chair of the SECCON (largest CTF in Japan)
- advisory board of the OWASP Japan
- leader of the Shibuya Perl Mongers group
- Microsoft MVP award of Developer Security in 2008
- review board for the CODE BLUE security conference

I am a chairman of the SECCON. SECCON is a largest security CTF contest in Japan. I am an CTF organizer and a challenge creator. I am also on the Open Web Application Security Project - OWASP Japan advisory board.
And I am the review board for the CODE BLUE which is a biggest international security conference in Japan.



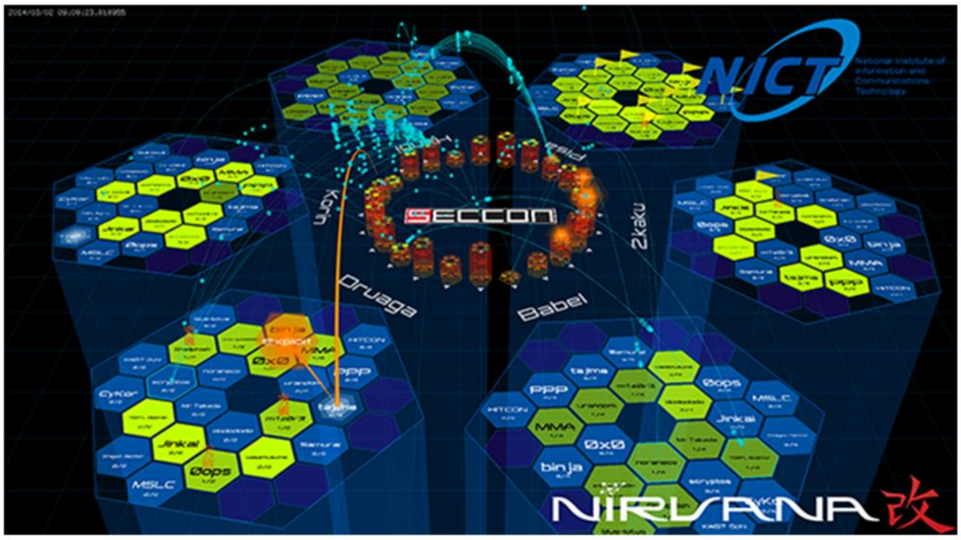
SECCON CTF Final competition (Tokyo, Japan)

SECCON
優勝
1st Place Champions
TOEFL Beginner
NIRVANA改

Rank	Score	AP, DP		Teams
1	4506	(1700, 2806)		TOEFL Beginner
2	3112	(1200, 1912)		HITCON
3	2848	(1500, 1348)		PPP
4	2304	(900, 1404)		binja
5	2013	(1400, 613)		0x0
6	1713	(1200, 513)		MMA
7	1570	(1000, 570)		Dragon Sector
8	1333	(900, 433)		CyKor
9	1286	(900, 386)		Mr.Takeda
10	1245	(900, 345)		urandom
11	1233	(900, 333)		blue-lotus
12	1205	(700, 505)		dododododo
13	1116	(1000, 116)		0ops
14	990	(300, 690)		wasamusume
15	972	(600, 372)		KAIST GoN
16	927	(900, 27)		Shellphish
17	759	(700, 59)		Samurai
18	585	(400, 185)		encrypt0x0n
19	504	(500, 4)		tajima
20	431	(400, 31)		m1z0r3
21	342	(300, 42)		MSLC
22	335	(300, 35)		Jinkai
23	320	(300, 20)		noraneco
24	313	(300, 13)		scryptos

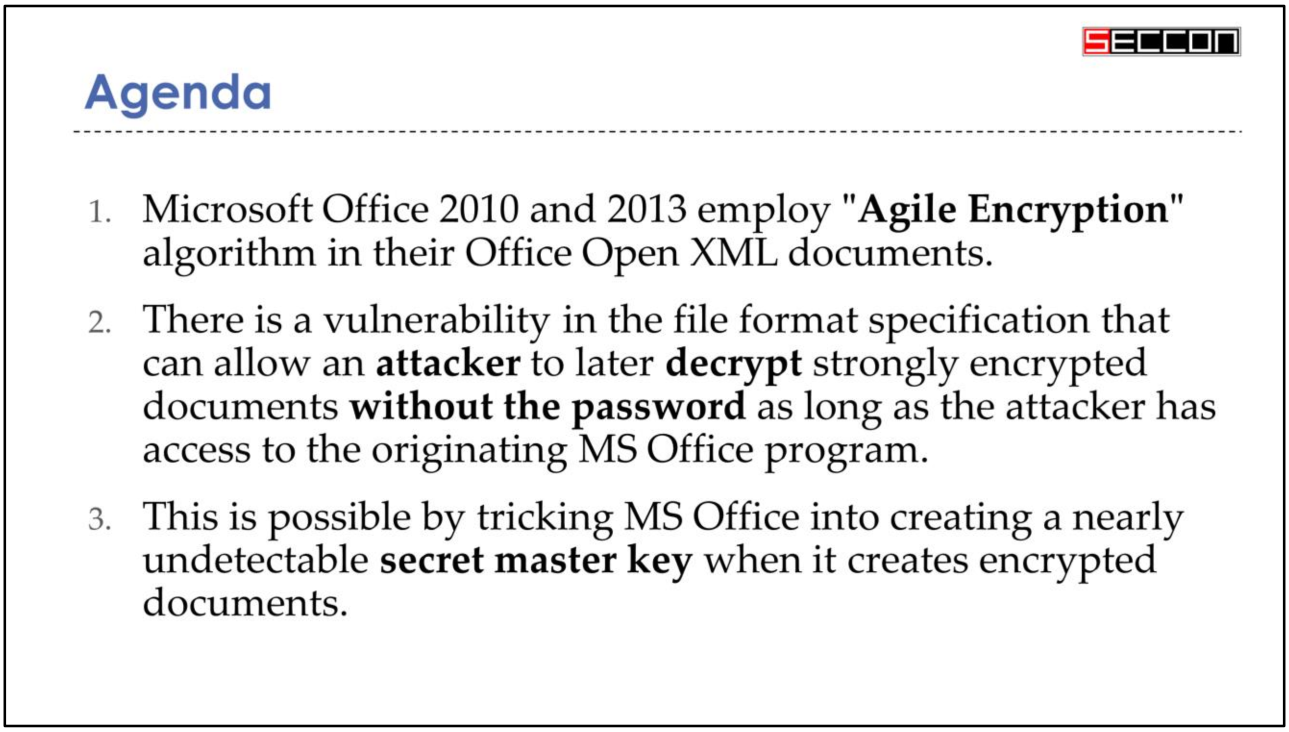
In this year, About 2,500 people took part in the SECCON CTF qualifier from 58 countries around the world. We held the international SECCON CTF Final competition in this year at Tokyo, Japan. Finally, Korean hacker team had won. That was great.

NIRVANA-KAI SECCON Customized Mk-II


SECCON
NICT
National Institute of Information and Communications Technology
Karin
Raid
2kaku
Oruaga
Babel
NIRVANA改

[1] http://www.nict.go.jp/info/topics/2015/02/150203-1.html

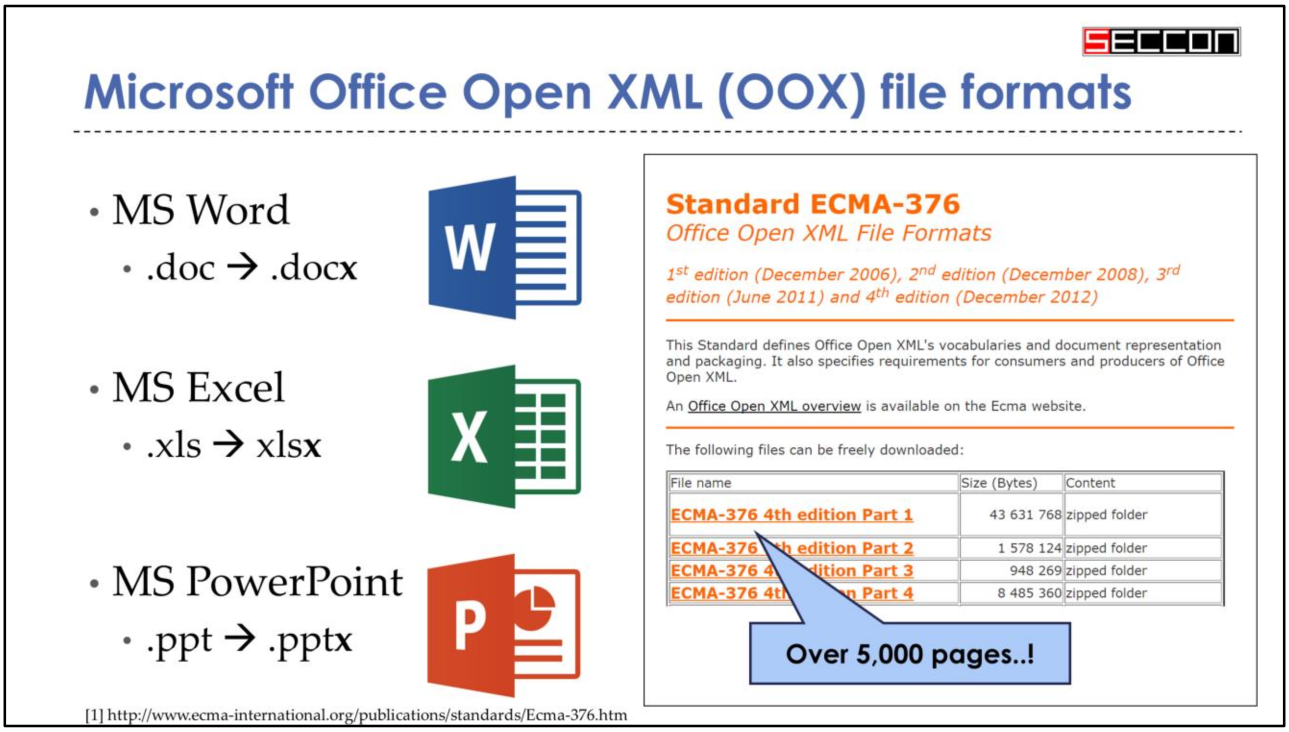
I want to show the NIRVANA-KAI SECCON Customized Mk-II. This is a real time visualization system for attack and defense battle of CTF. This real time visualization system was developed by National Institute of Information and Communications Technology - NICT in Japan.


Agenda

1. Microsoft Office 2010 and 2013 employ "**Agile Encryption**" algorithm in their Office Open XML documents.
2. There is a vulnerability in the file format specification that can allow an **attacker** to later **decrypt** strongly encrypted documents **without the password** as long as the attacker has access to the originating MS Office program.
3. This is possible by tricking MS Office into creating a nearly undetectable **secret master key** when it creates encrypted documents.

Okay, I talk about "Backdooring MS Office documents with secret master keys". We made a lot of CTF challenges such as a XSS, reversing, pwn, cryptography at SECCON CTF project. when I created some cryptography challenges, I found this backdoor problem. Microsoft Office twenty ten or later version employ "Agile Encryption" algorithm in their OOX documents. We found a vulnerability in the file format specification that can allow an attacker to later decrypt strongly encrypted documents without the password. This is possible by tricking MS Office into creating an undetectable secret master key when it creates encrypted

documents.



Microsoft Office Open XML (OOX) file formats

- MS Word
  - .doc → .docx
- MS Excel
  - .xls → xlsx
- MS PowerPoint
  - .ppt → .pptx

Standard ECMA-376

Office Open XML File Formats

1st edition (December 2006), 2nd edition (December 2008), 3rd edition (June 2011) and 4th edition (December 2012)

This Standard defines Office Open XML's vocabularies and document representation and packaging. It also specifies requirements for consumers and producers of Office Open XML.

An Office Open XML overview is available on the Ecma website.

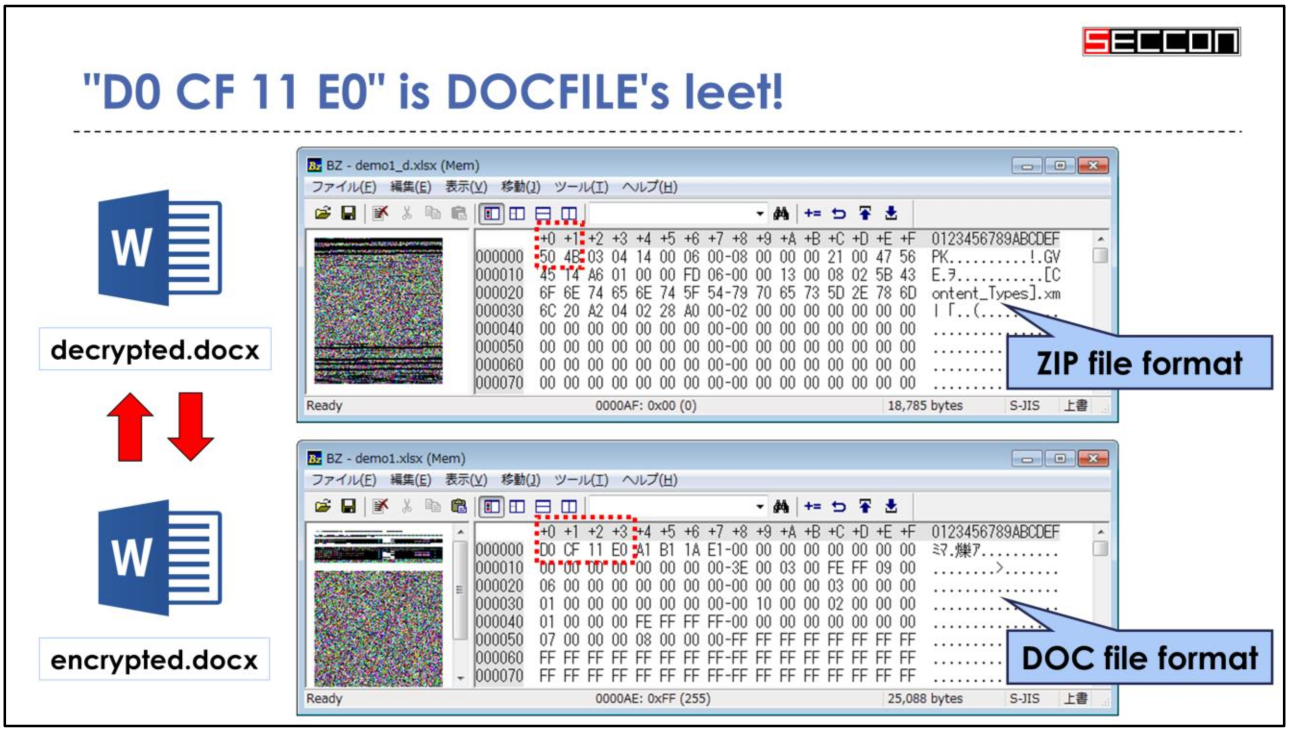
The following files can be freely downloaded:

File name	Size (Bytes)	Content
ECMA-376 4th edition Part 1	43 631 768	zipped folder
ECMA-376 th edition Part 2	1 578 124	zipped folder
ECMA-376 4 ition Part 3	948 269	zipped folder
ECMA-376 4th n Part 4	8 485 360	zipped folder

Over 5,000 pages..!

[1] http://www.ecma-international.org/publications/standards/Ecma-376.htm

Microsoft has standardized the OOX file format by ECMA international. It is not "Open Office" XML format, Open Office is a rival application for Microsoft. So "Office Open" XML format is correct.
You can see at the DOCX suffix from filename. And you know it is just a zip archive file.


"D0 CF 11 E0" is DOCFILE's leet!

decrypted.docx

```
BZ - demo1_d.xlsx (Mem)
         +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +A +B +C +D +E +F  0123456789ABCDEF
000000   50 4B 03 04 14 00 06 00-08 00 00 00 21 00 47 56  PK..........!.GV
000010   45 14 A6 01 00 00 FD 06-00 00 13 00 08 02 5B 43  E.ｦ...........[C
000020   6F 6E 74 65 6E 74 5F 54-79 70 65 73 5D 2E 78 6D  ontent_Types].xm
000030   6C 20 A2 04 02 28 A0 00-02 00 00 00 00 00 00 00  l ｢..(.........
000040   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .........
000050   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .........
000060   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .........
000070   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  .........
Ready                0000AF: 0x00 (0)          18,785 bytes   S-JIS   上書
```

ZIP file format

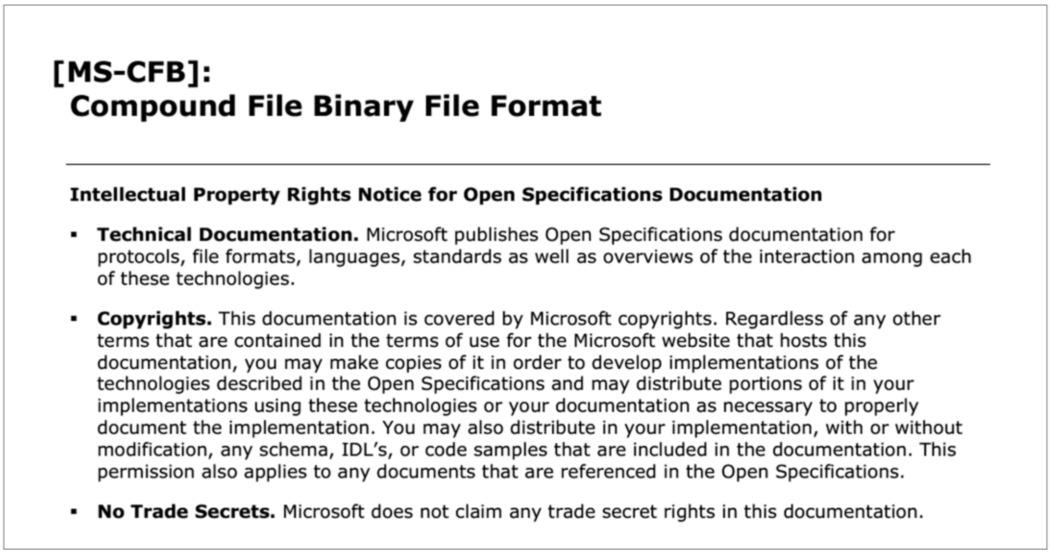
encrypted.docx

```
BZ - demo1.xlsx (Mem)
         +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +A +B +C +D +E +F  0123456789ABCDEF
000000   D0 CF 11 E0 A1 B1 1A E1-00 00 00 00 00 00 00 00  ﾐﾏ.爍ｱ..........
000010   00 00 00 00 00 00 00 00-3E 00 03 00 FE FF 09 00  ........>.......
000020   06 00 00 00 00 00 00 00-00 00 00 00 03 00 00 00  ................
000030   01 00 00 00 00 00 00 00-00 10 00 00 02 00 00 00  ................
000040   01 00 00 00 FE FF FF FF-00 00 00 00 00 00 00 00  .........
000050   07 00 00 00 08 00 00 00-FF FF FF FF FF FF FF FF  .........
000060   FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF  .........
000070   FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF  .........
Ready                0000AE: 0xFF (255)        25,088 bytes   S-JIS   上書
```

DOC file format

However, when you encrypt a DOCX file, it will become an old classical DOC FILE format. This encrypted.DOCX file has a DOCX suffix. But, it is not the zip archive file. You can see at file hex dump header, "D0 CF 11 E0“. it is DOCFILE's leet character!

[MS-CFB] Compound File Binary File Format

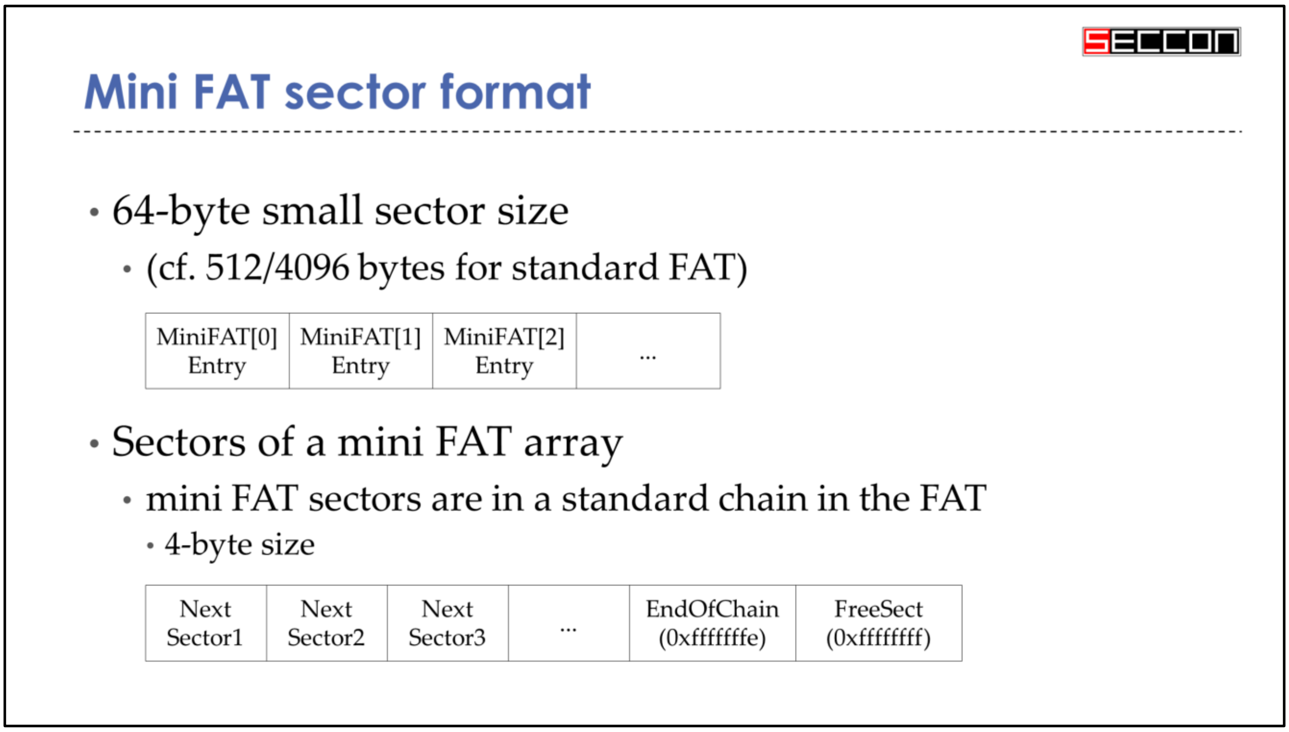


[MS-CFB]:
Compound File Binary File Format

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.

Old classical DOC FILE format has been standardized as the MS-CFB. This specifications documentation was opened by Microsoft. I think it is a great job.



Mini FAT sector format

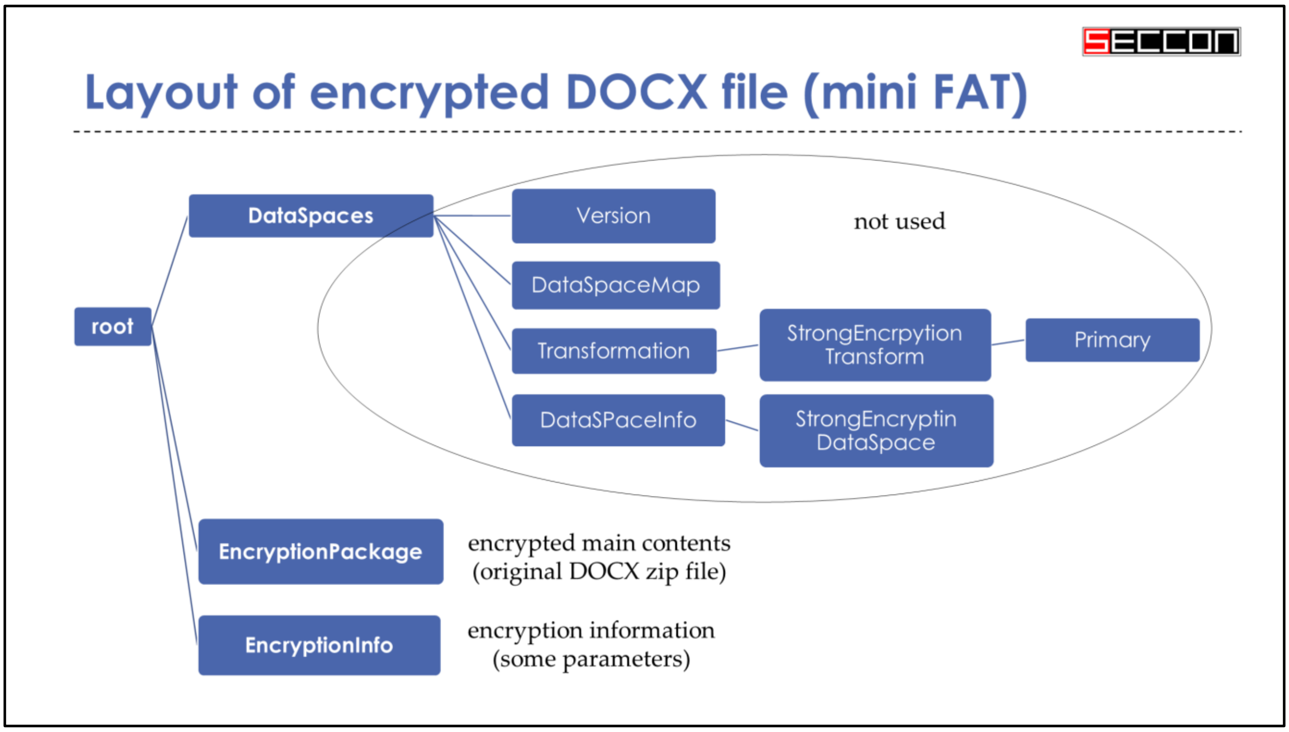
- 64-byte small sector size
  - (cf. 512/4096 bytes for standard FAT)

MiniFAT[0] Entry	MiniFAT[1] Entry	MiniFAT[2] Entry	…

- Sectors of a mini FAT array
  - mini FAT sectors are in a standard chain in the FAT
    - 4-byte size

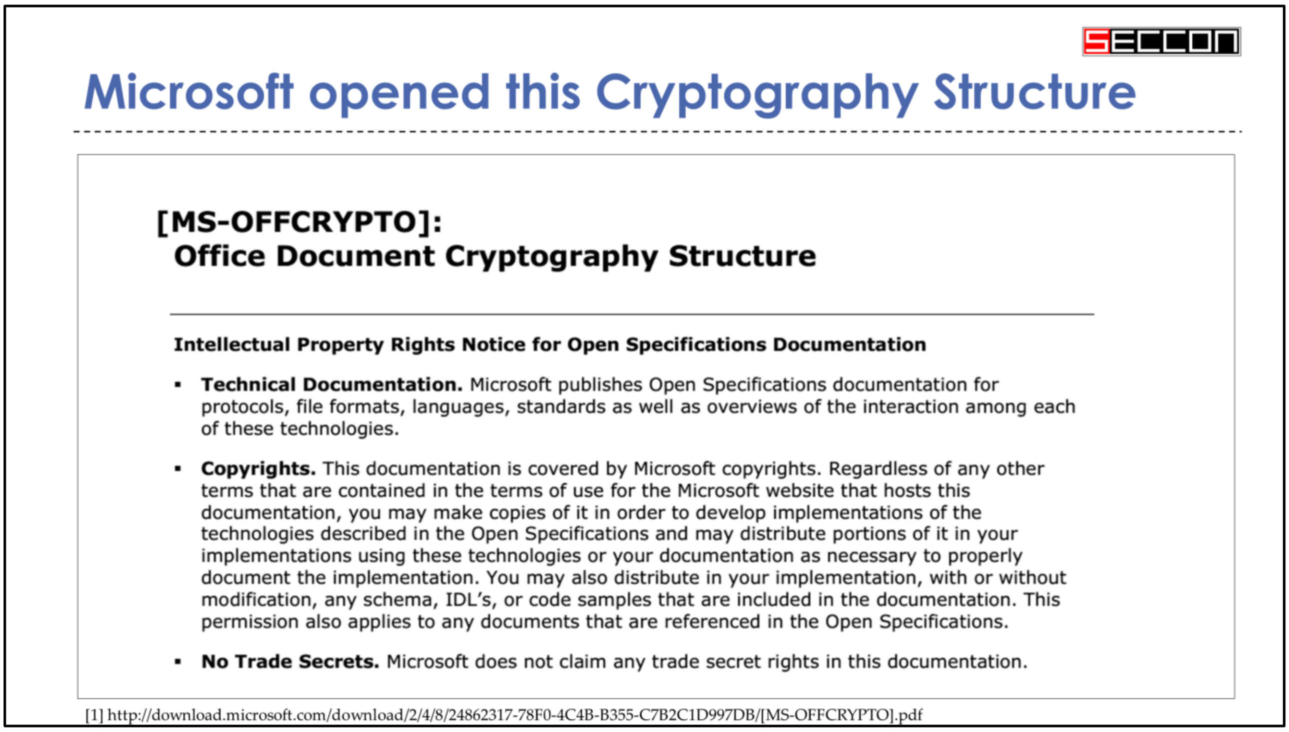
Next Sector1	Next Sector2	Next Sector3	…	EndOfChain (0xfffffffe)	FreeSect (0xffffffff)

MS-CFB file format has some mini FAT sectors. Mini FAT has a 64-byte small sector size. mini FAT sectors are in a standard chain in the FAT.


Layout of encrypted DOCX file (mini FAT)

SECCON
root
DataSpaces
Version
DataSpaceMap
Transformation
DataSPaceInfo
StrongEncrpytion Transform
Primary
StrongEncryptin DataSpace
not used
EncryptionPackage
encrypted main contents (original DOCX zip file)
EncryptionInfo
encryption information (some parameters)

This is a figure of File layout of encrypted DOCX file. Any encrypted DOCX file have a these file in mini FAT. EncryptionPackage is an binary file which was encrypted from original DOCX zip file object. EncryptionInfo is very important information for these encryption parameters.



Microsoft opened this Cryptography Structure

[MS-OFFCRYPTO]:
Office Document Cryptography Structure

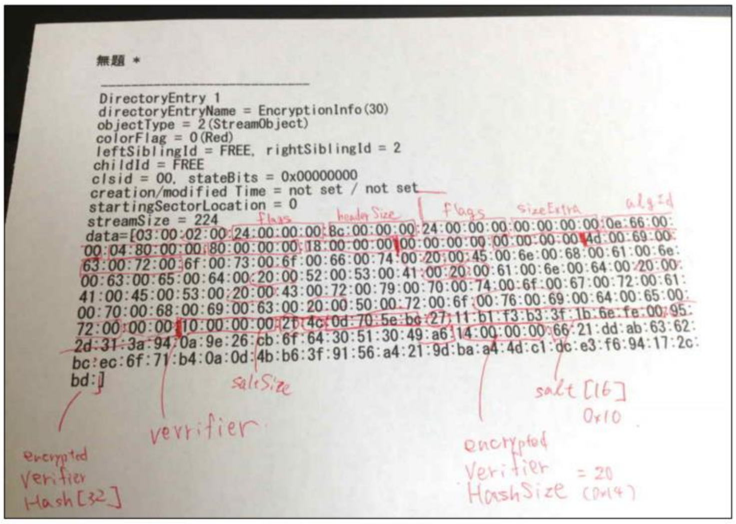
Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.

[1] http://download.microsoft.com/download/2/4/8/24862317-78F0-4C4B-B355-C7B2C1D997DB/[MS-OFFCRYPTO].pdf

Microsoft opened this Office cryptography Cryptography Structure as [MS-OFFCRYPTO]. We tried to read the MS-OFFCRYPTO document carefully.

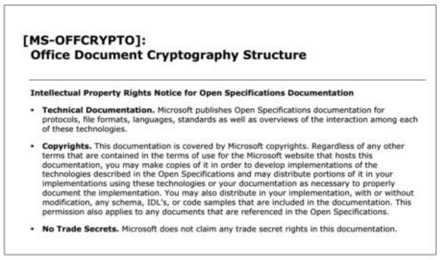
Yes, we can!


```
無題 *

DirectoryEntry 1
directoryEntryName = EncryptionInfo(30)
objectType = 2(StreamObject)
colorFlag = 0(Red)
leftSiblingId = FREE, rightSiblingId = 2
childId = FREE
clsid = 00, stateBits = 0x00000000
creation/modified Time = not set / not set
startingSectorLocation = 0
streamSize = 224
data=[03:00:02:00:24:00:00:00:8c:00:00:00:24:00:00:00:00:00:00:00:0e:66:00:
00:04:80:00:00:80:00:00:00:18:00:00:00:00:00:00:00:00:00:00:00:4d:00:69:00:
63:00:72:00:6f:00:73:00:6f:00:66:00:74:00:20:00:45:00:6e:00:68:00:61:00:6e:
00:63:00:65:00:64:00:20:00:52:00:53:00:41:00:20:00:61:00:6e:00:64:00:20:00:
41:00:45:00:53:00:20:00:43:00:72:00:79:00:70:00:74:00:6f:00:67:00:72:00:61:
00:70:00:68:00:69:00:63:00:20:00:50:00:72:00:6f:00:76:00:69:00:64:00:65:00:
72:00:00:00:10:00:00:00:2f:4c:0d:70:5e:bc:27:11:b1:f3:b3:3f:1b:6e:fe:00:95:
2d:31:3a:94:0a:9e:26:cb:6f:64:30:51:30:49:a6:14:00:00:00:66:21:dd:ab:63:62:
bc:ec:6f:71:b4:0a:0d:4b:b6:3f:91:56:a4:21:9d:ba:a4:4d:c1:dc:e3:f6:94:17:2c:
bd:]
```
flags
header Size
flags
sizeExtra
algId
saltSize
verrifier
salt [16]
0x10
encrypted Verifier Hash [32]
encrypted Verifier HashSize = 20 (0x14)

[MS-CFB]:
Compound File Binary File Format

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.

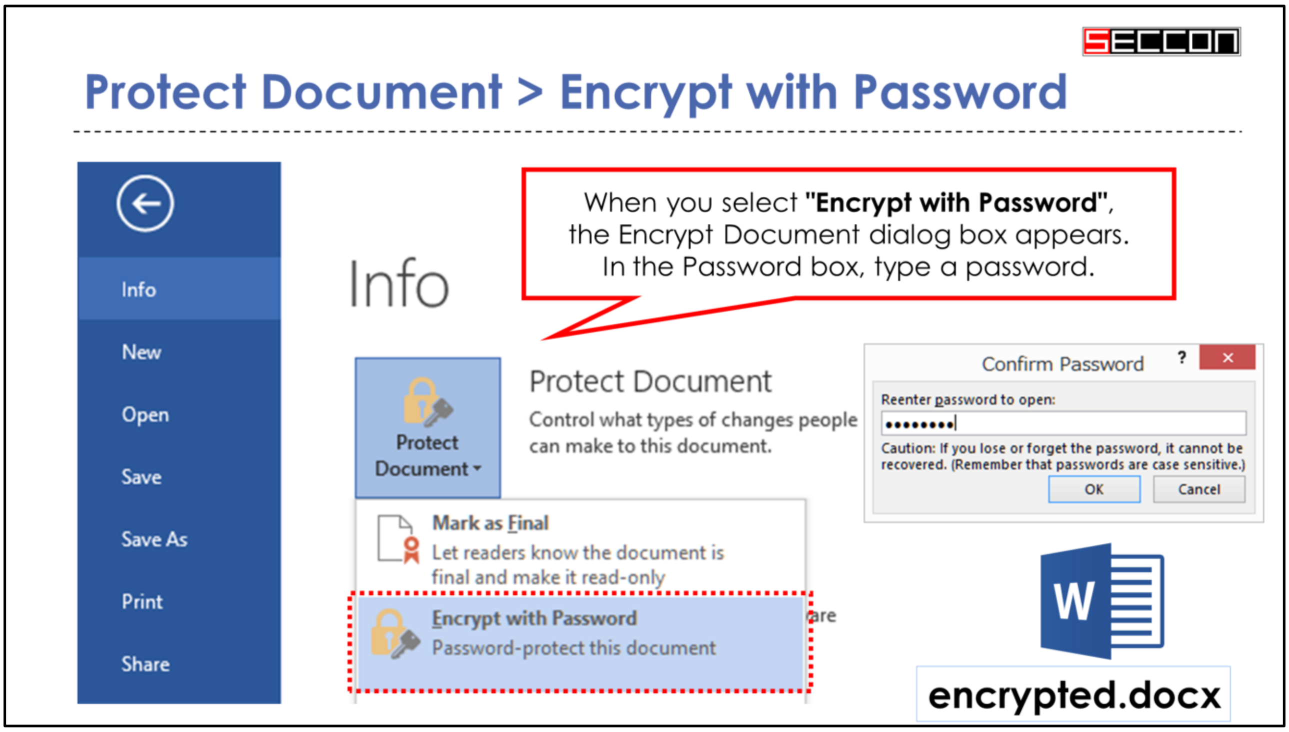

[MS-OFFCRYPTO]:
Office Document Cryptography Structure

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.

Yes, we can read the binary of DOC FILE!

Protect Document > Encrypt with Password



When you select **"Encrypt with Password"**, the Encrypt Document dialog box appears. In the Password box, type a password.

Info

Protect Document
Control what types of changes people can make to this document.

Mark as Final
Let readers know the document is final and make it read-only

Encrypt with Password
Password-protect this document

Confirm Password

Reenter password to open:
••••••••

Caution: If you lose or forget the password, it cannot be recovered. (Remember that passwords are case sensitive.)

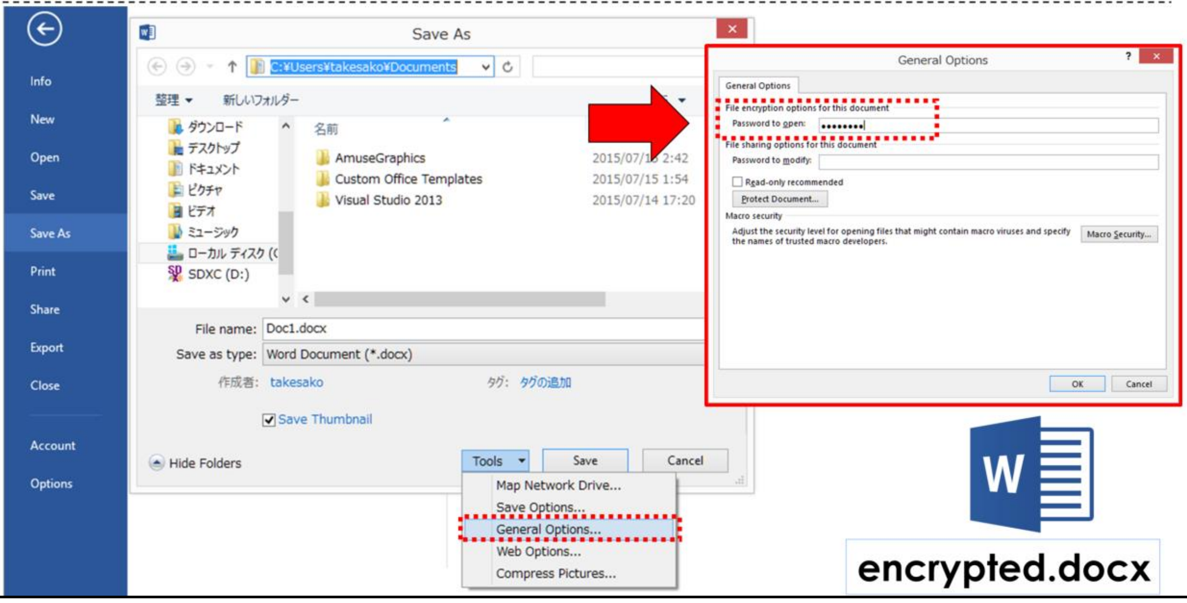
OK Cancel

encrypted.docx

If you want to protect your document with passwords. In Microsoft Office, you can use passwords to help prevent other people from opening or modifying your documents.
Select the Protect Document menu on Info tab.
You can choose some submenu.

When you select "Encrypt with Password", the Encrypt Document dialog box appears.
In the Password box, type a password. And confirm password. Then Encrypted.docx is saved.

SaveAs > Tools > GeneralOptions > Password



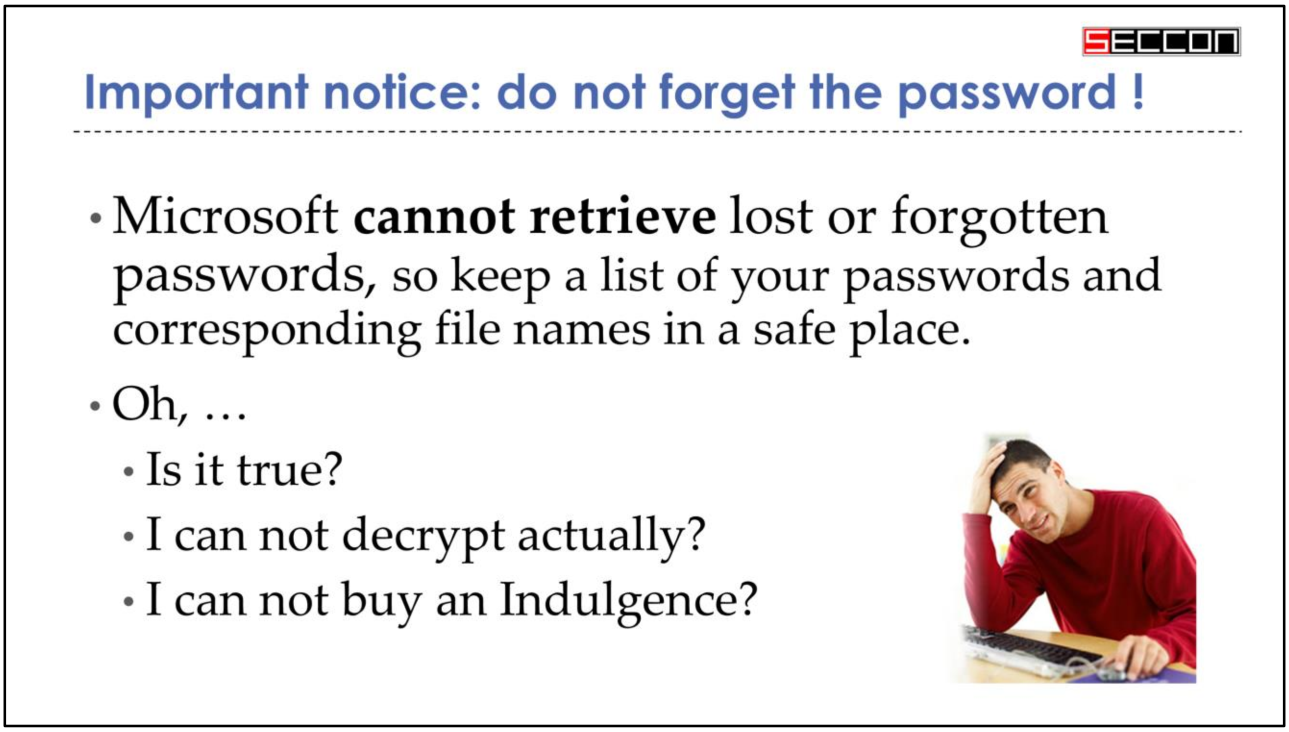
AmuseGraphics 2015/07/15 2:42
Custom Office Templates 2015/07/15 1:54
Visual Studio 2013 2015/07/14 17:20

File name: Doc1.docx
Save as type: Word Document (*.docx)
作成者: takesako タグ: タグの追加
☑ Save Thumbnail

File encryption options for this document
Password to open: ••••••••
File sharing options for this document
Password to modify:
☐ Read-only recommended
Macro security
Adjust the security level for opening files that might contain macro viruses and specify the names of trusted macro developers.

encrypted.docx

**There are another manipulation.
You can select SaveAs menu.
And push the Tool button and select GeneralOption.
Then you can input the password just the same way.**

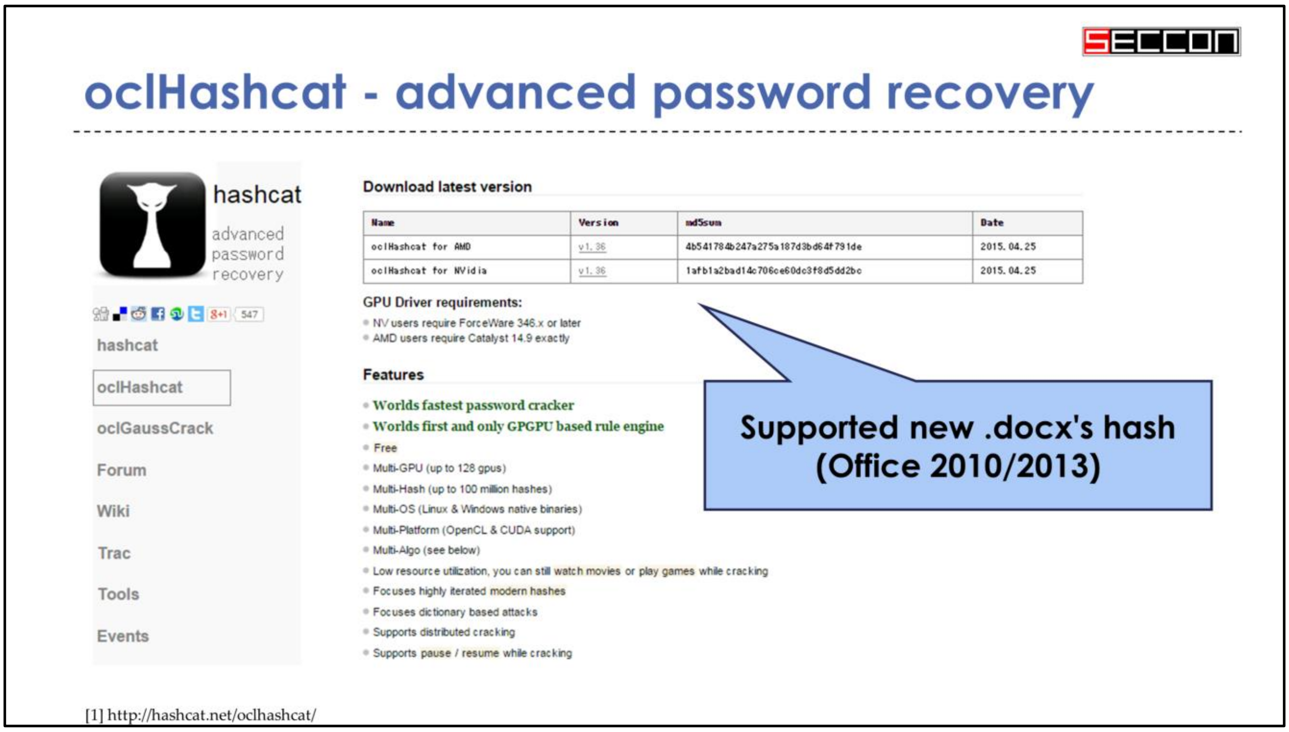


Important notice: do not forget the password !

- Microsoft **cannot retrieve** lost or forgotten passwords, so keep a list of your passwords and corresponding file names in a safe place.
- Oh, …
  - Is it true?
  - I can not decrypt actually?
  - I can not buy an Indulgence?

It's important to know that you don't forget the password. Because Microsoft cannot retrieve your forgotten passwords. If you forget the password, we can not retrieve the original documents.

But, is it true? I cannot decrypt actually?



oclHashcat - advanced password recovery

Download latest version

Name	Version	md5sum	Date
oclHashcat for AMD	v1.36	4b541784b247a275a187d3bd64f791de	2015.04.25
oclHashcat for NVidia	v1.36	1afb1a2bad14c706ce60dc3f8d5dd2bc	2015.04.25

GPU Driver requirements:

- NV users require ForceWare 346.x or later
- AMD users require Catalyst 14.9 exactly

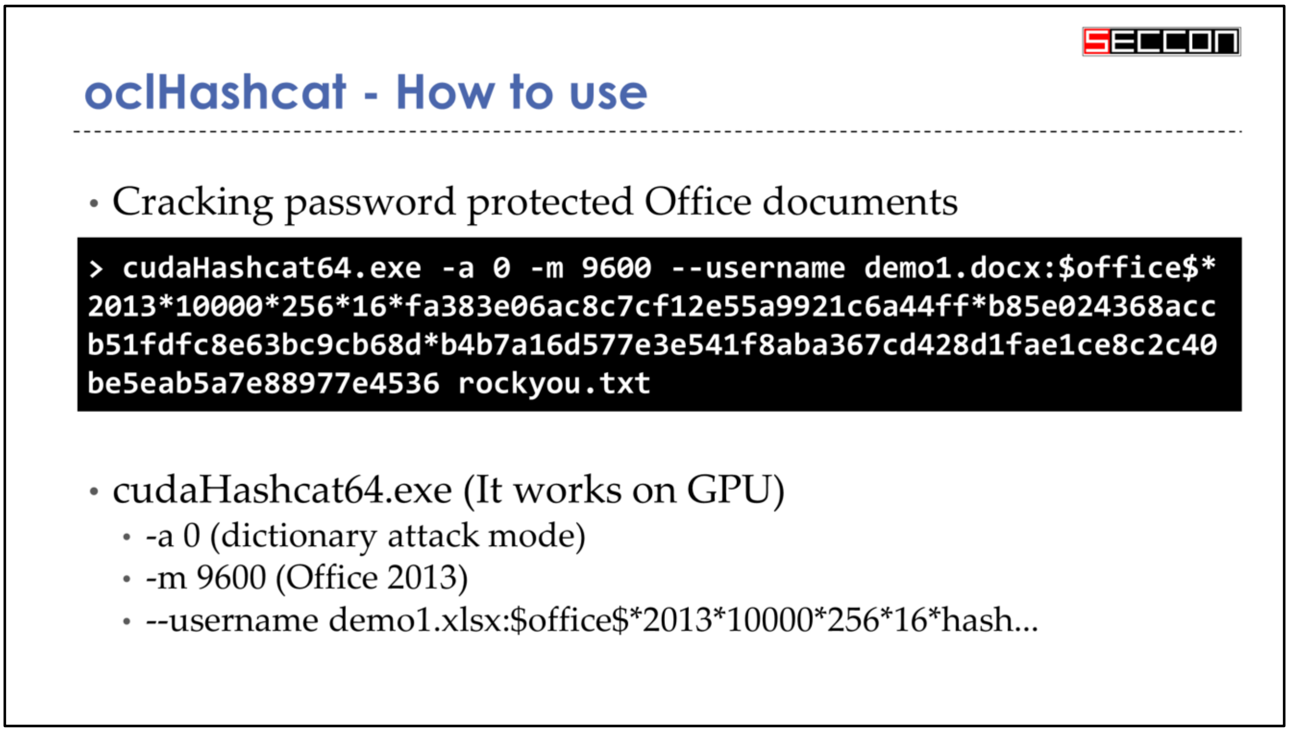
Features

- Worlds fastest password cracker
- Worlds first and only GPGPU based rule engine
- Free
- Multi-GPU (up to 128 gpus)
- Multi-Hash (up to 100 million hashes)
- Multi-OS (Linux & Windows native binaries)
- Multi-Platform (OpenCL & CUDA support)
- Multi-Algo (see below)
- Low resource utilization, you can still watch movies or play games while cracking
- Focuses highly iterated modern hashes
- Focuses dictionary based attacks
- Supports distributed cracking
- Supports pause / resume while cracking

Supported new .docx's hash (Office 2010/2013)

[1] http://hashcat.net/oclhashcat/

For when this occurs , there are password recovery software. oclHashcat is one of the famous password recovery tools by command line.



SECCON

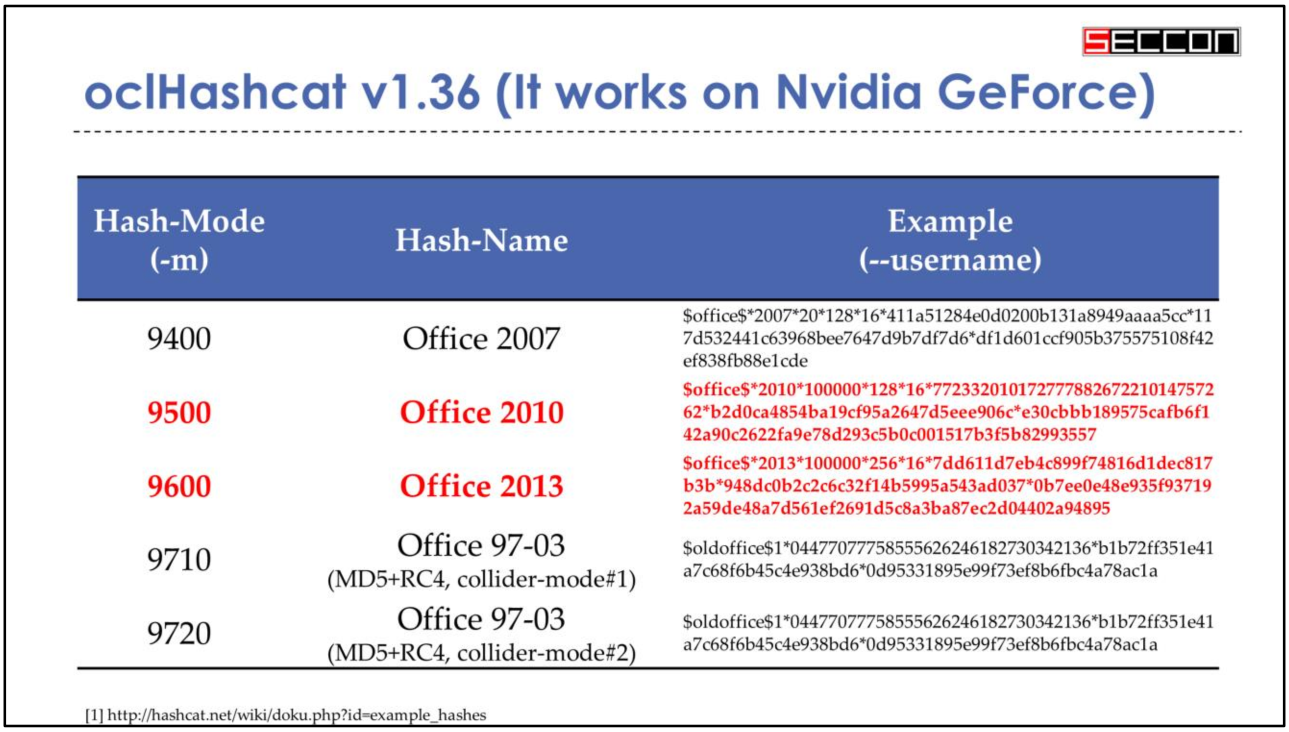
oclHashcat - How to use

- Cracking password protected Office documents

```
> cudaHashcat64.exe -a 0 -m 9600 --username demo1.docx:$office$*
2013*10000*256*16*fa383e06ac8c7cf12e55a9921c6a44ff*b85e024368acc
b51fdfc8e63bc9cb68d*b4b7a16d577e3e541f8aba367cd428d1fae1ce8c2c40
be5eab5a7e88977e4536 rockyou.txt
```

- cudaHashcat64.exe (It works on GPU)
  - -a 0 (dictionary attack mode)
  - -m 9600 (Office 2013)
  - --username demo1.xlsx:$office$*2013*10000*256*16*hash...

If you want to crack password protected MS Office documents, type this command.

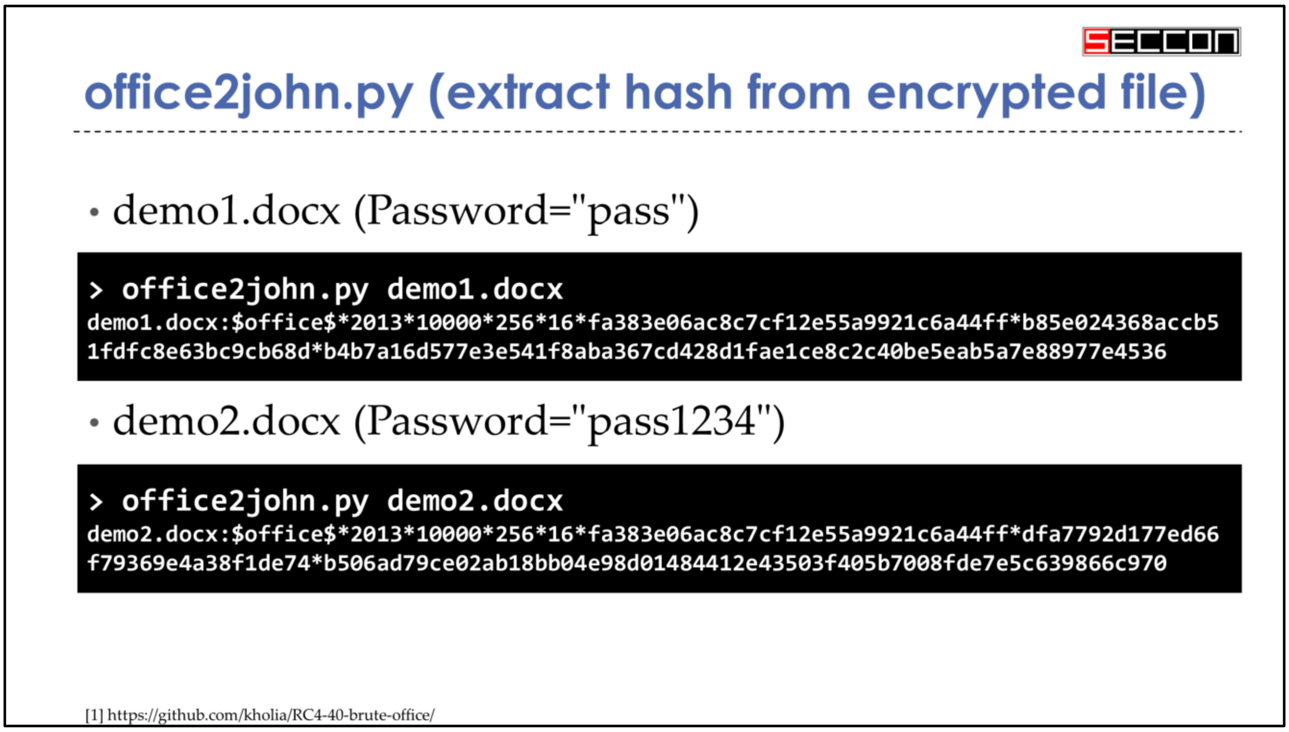


oclHashcat v1.36 (It works on Nvidia GeForce)

Hash-Mode (-m)	Hash-Name	Example (--username)
9400	Office 2007	$office$*2007*20*128*16*411a51284e0d0200b131a8949aaaa5cc*117d532441c63968bee7647d9b7df7d6*df1d601ccf905b375575108f42ef838fb88e1cde
9500	**Office 2010**	**$office$*2010*100000*128*16*77233201017277788267221014757262*b2d0ca4854ba19cf95a2647d5eee906c*e30cbbb189575cafb6f142a90c2622fa9e78d293c5b0c001517b3f5b82993557**
9600	**Office 2013**	**$office$*2013*100000*256*16*7dd611d7eb4c899f74816d1dec817b3b*948dc0b2c2c6c32f14b5995a543ad037*0b7ee0e48e935f937192a59de48a7d561ef2691d5c8a3ba87ec2d04402a94895**
9710	Office 97-03 (MD5+RC4, collider-mode#1)	$oldoffice$1*04477077758555626246182730342136*b1b72ff351e41a7c68f6b45c4e938bd6*0d95331895e99f73ef8b6fbc4a78ac1a
9720	Office 97-03 (MD5+RC4, collider-mode#2)	$oldoffice$1*04477077758555626246182730342136*b1b72ff351e41a7c68f6b45c4e938bd6*0d95331895e99f73ef8b6fbc4a78ac1a

[1] http://hashcat.net/wiki/doku.php?id=example_hashes

Recently oclHashcat supports GP-GPU power, and supported new Office document OOX file format.



office2john.py (extract hash from encrypted file)

- demo1.docx (Password="pass")

```
> office2john.py demo1.docx
demo1.docx:$office$*2013*10000*256*16*fa383e06ac8c7cf12e55a9921c6a44ff*b85e024368accb5
1fdfc8e63bc9cb68d*b4b7a16d577e3e541f8aba367cd428d1fae1ce8c2c40be5eab5a7e88977e4536
```

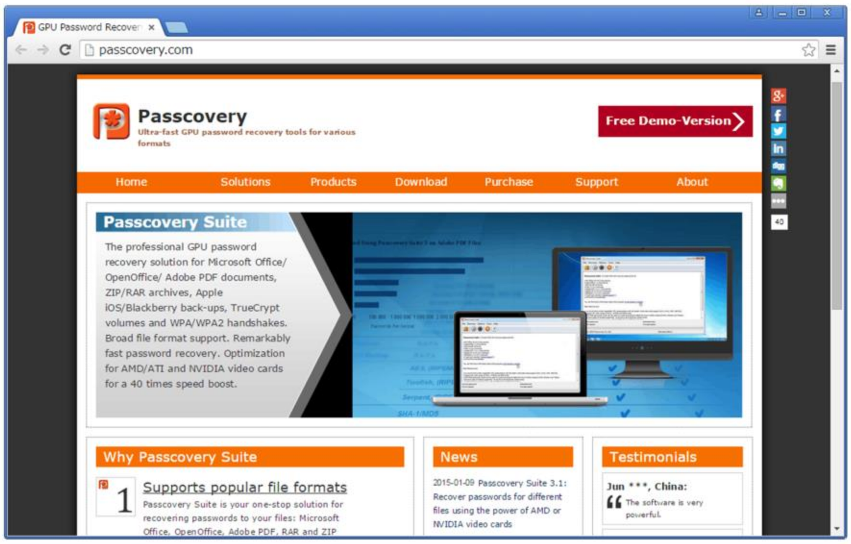
- demo2.docx (Password="pass1234")

```
> office2john.py demo2.docx
demo2.docx:$office$*2013*10000*256*16*fa383e06ac8c7cf12e55a9921c6a44ff*dfa7792d177ed66
f79369e4a38f1de74*b506ad79ce02ab18bb04e98d01484412e43503f405b7008fde7e5c639866c970
```

[1] https://github.com/kholia/RC4-40-brute-office/

Before you have to extract hash from encrypted file by office2john.py.

Passcovery - powerful password recovery tools


GPU Password Recover
passcovery.com

Passcovery
Ultra-fast GPU password recovery tools for various formats

Free Demo-Version

Home Solutions Products Download Purchase Support About

Passcovery Suite
The professional GPU password recovery solution for Microsoft Office/ OpenOffice/ Adobe PDF documents, ZIP/RAR archives, Apple iOS/Blackberry back-ups, TrueCrypt volumes and WPA/WPA2 handshakes. Broad file format support. Remarkably fast password recovery. Optimization for AMD/ATI and NVIDIA video cards for a 40 times speed boost.

Why Passcovery Suite
1 Supports popular file formats
Passcovery Suite is your one-stop solution for recovering passwords to your files: Microsoft Office, OpenOffice, Adobe PDF, RAR and ZIP

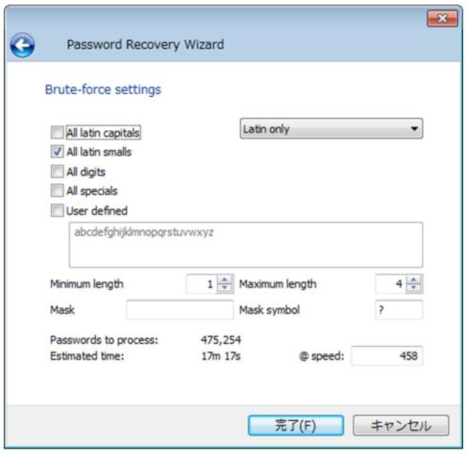
News
2015-01-09 Passcovery Suite 3.1: Recover passwords for different files using the power of AMD or NVIDIA video cards

Testimonials
Jun ***, China:
The software is very powerful.

[1] http://passcovery.com/

There are another password recovery software with simple User interface. I used the Passcovery commercial edition which is very powerful password recovery tools.

Passcovery > Password Recovery Wizard (GUI)



Brute-force settings

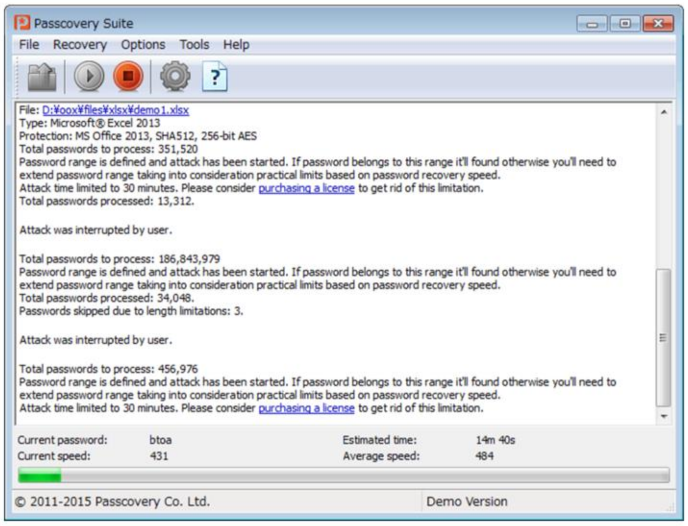
[ ] All latin capitals
[✓] All latin smalls
[ ] All digits
[ ] All specials
[ ] User defined

Latin only

abcdefghijklmnopqrstuvwxyz

Minimum length 1 Maximum length 4
Mask Mask symbol ?

Passwords to process: 475,254
Estimated time: 17m 17s @ speed: 458

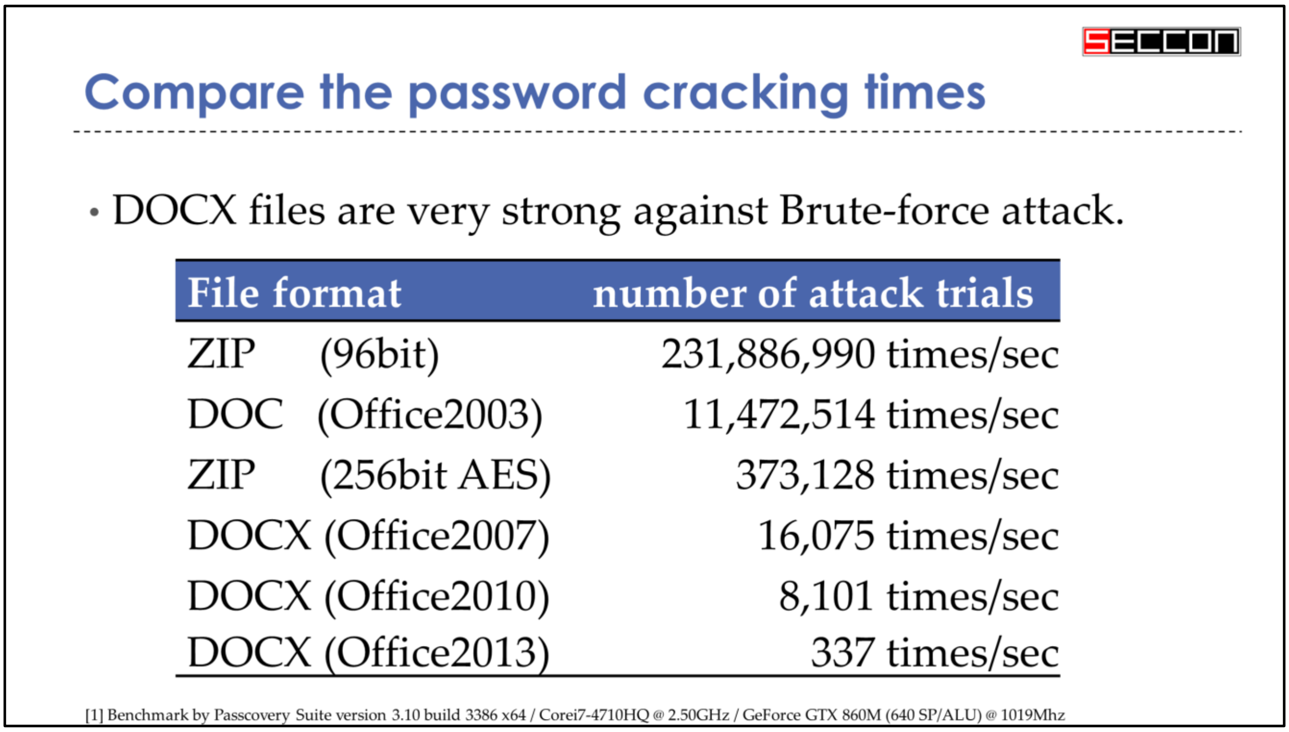


```
File: D:¥oox¥files¥xlsx¥demo1.xlsx
Type: Microsoft® Excel 2013
Protection: MS Office 2013, SHA512, 256-bit AES
Total passwords to process: 351,520
Password range is defined and attack has been started. If password belongs to this range it'll found otherwise you'll need to
extend password range taking into consideration practical limits based on password recovery speed.
Attack time limited to 30 minutes. Please consider purchasing a license to get rid of this limitation.
Total passwords processed: 13,312.

Attack was interrupted by user.

Total passwords to process: 186,843,979
Password range is defined and attack has been started. If password belongs to this range it'll found otherwise you'll need to
extend password range taking into consideration practical limits based on password recovery speed.
Total passwords processed: 34,048.
Passwords skipped due to length limitations: 3.

Attack was interrupted by user.

Total passwords to process: 456,976
Password range is defined and attack has been started. If password belongs to this range it'll found otherwise you'll need to
extend password range taking into consideration practical limits based on password recovery speed.
Attack time limited to 30 minutes. Please consider purchasing a license to get rid of this limitation.
```

Current password: btoa Estimated time: 14m 40s
Current speed: 431 Average speed: 484

© 2011-2015 Passcovery Co. Ltd. Demo Version

It's very simple graphic user interface. Only clicking.



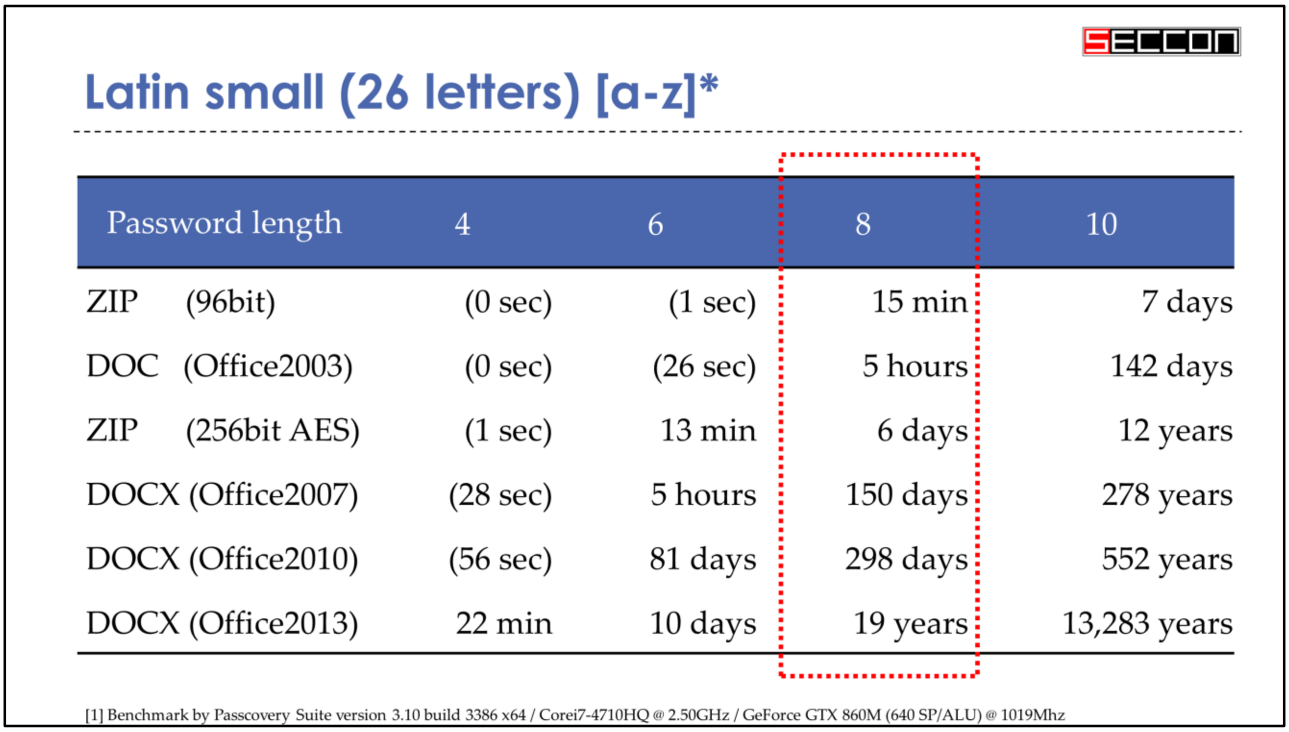
Compare the password cracking times

- DOCX files are very strong against Brute-force attack.

File format	number of attack trials
ZIP (96bit)	231,886,990 times/sec
DOC (Office2003)	11,472,514 times/sec
ZIP (256bit AES)	373,128 times/sec
DOCX (Office2007)	16,075 times/sec
DOCX (Office2010)	8,101 times/sec
DOCX (Office2013)	337 times/sec

[1] Benchmark by Passcovery Suite version 3.10 build 3386 x64 / Corei7-4710HQ @ 2.50GHz / GeForce GTX 860M (640 SP/ALU) @ 1019Mhz

I evaluated comparing the decryption time of password cracking. There are some encryption file format. Classic Zip and AES Zip nad old DOCFILE and new DOCX files.
DOCX files are very strong against Brute-force attack.

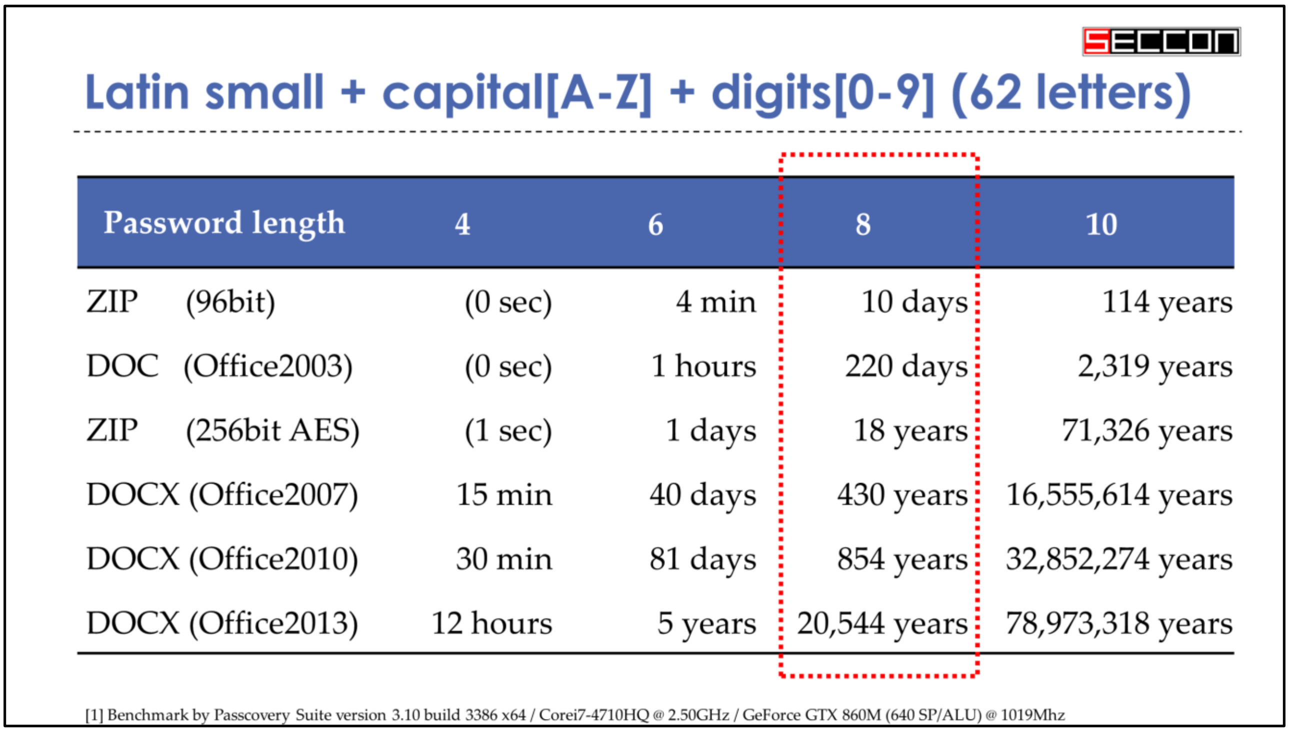


Latin small (26 letters) [a-z]*

Password length	4	6	8	10
ZIP (96bit)	(0 sec)	(1 sec)	15 min	7 days
DOC (Office2003)	(0 sec)	(26 sec)	5 hours	142 days
ZIP (256bit AES)	(1 sec)	13 min	6 days	12 years
DOCX (Office2007)	(28 sec)	5 hours	150 days	278 years
DOCX (Office2010)	(56 sec)	81 days	298 days	552 years
DOCX (Office2013)	22 min	10 days	19 years	13,283 years

[1] Benchmark by Passcovery Suite version 3.10 build 3386 x64 / Corei7-4710HQ @ 2.50GHz / GeForce GTX 860M (640 SP/ALU) @ 1019Mhz

Password consists of Latin small characters an Latin capital characters and digits and special symbols characters. If the password length is 8. Time required to decrypt the encrypted Classic ZIP file by brute force attack was 15 minutes. Its encryption key bit is only 96.
WinZIP have a long AES encryption key, Time required to decrypt the encrypted new WinZIP file by brute force attack was 6 days. Time required of brute force attacking has increased gradually with Office version is newer.



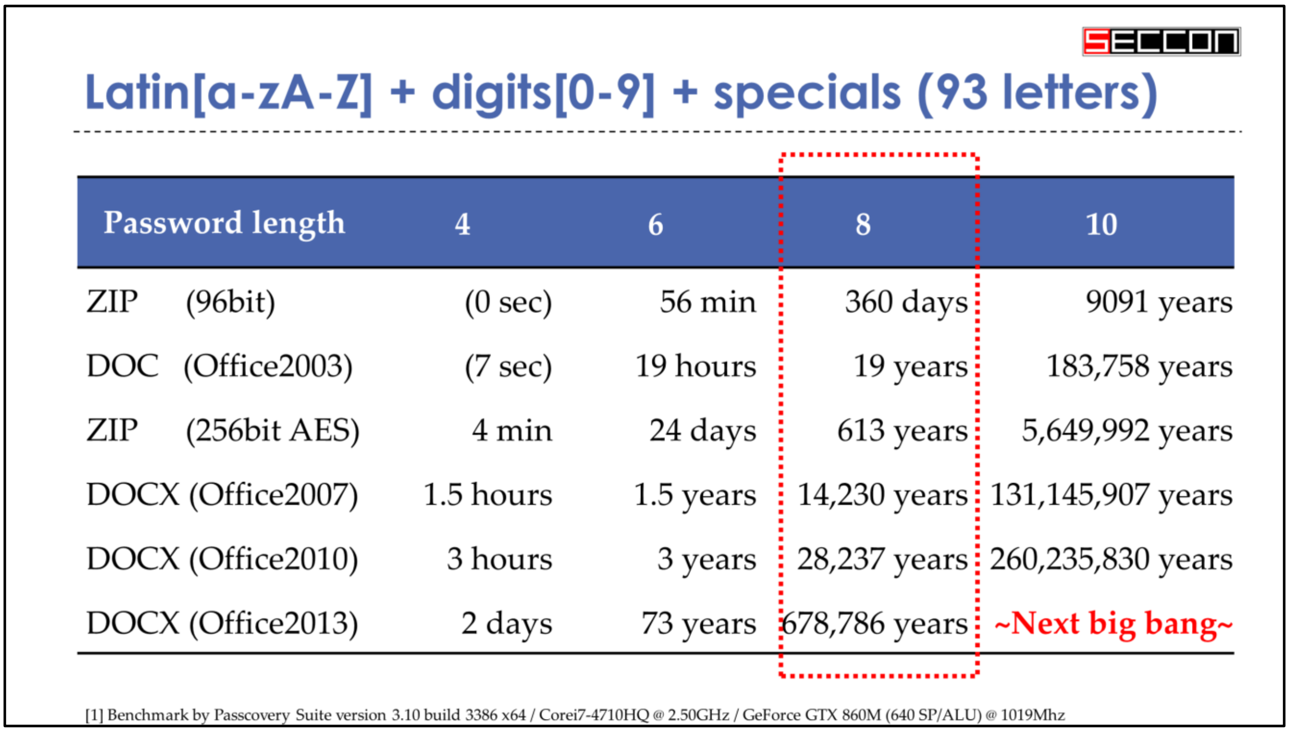
SECCON

Latin small + capital[A-Z] + digits[0-9] (62 letters)

Password length	4	6	8	10
ZIP (96bit)	(0 sec)	4 min	10 days	114 years
DOC (Office2003)	(0 sec)	1 hours	220 days	2,319 years
ZIP (256bit AES)	(1 sec)	1 days	18 years	71,326 years
DOCX (Office2007)	15 min	40 days	430 years	16,555,614 years
DOCX (Office2010)	30 min	81 days	854 years	32,852,274 years
DOCX (Office2013)	12 hours	5 years	20,544 years	78,973,318 years

[1] Benchmark by Passcovery Suite version 3.10 build 3386 x64 / Corei7-4710HQ @ 2.50GHz / GeForce GTX 860M (640 SP/ALU) @ 1019Mhz

Password consists of Latin small characters an Latin capital characters and digits characters. If the password length is 8. DOCX's time required of brute force attacking was about Twenty thousand years with Office version is twenty thirteen.



Latin[a-zA-Z] + digits[0-9] + specials (93 letters)

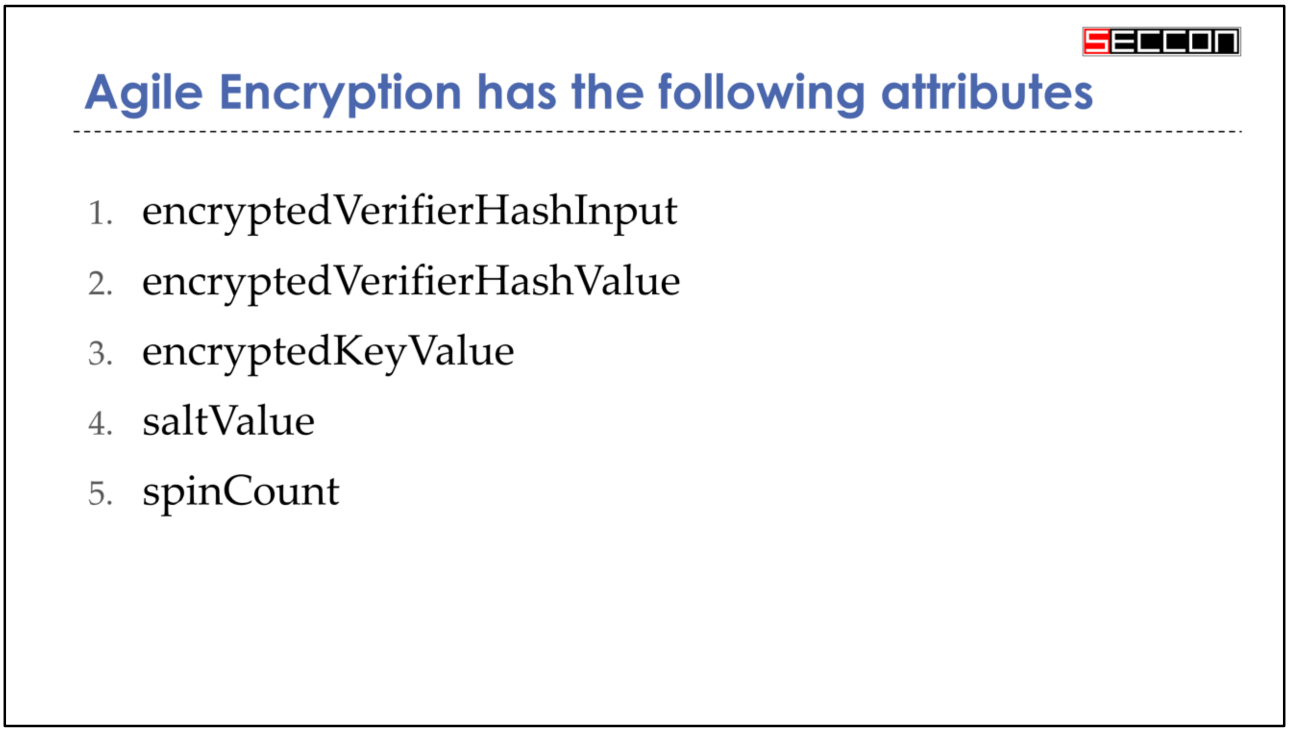
Password length	4	6	8	10
ZIP (96bit)	(0 sec)	56 min	360 days	9091 years
DOC (Office2003)	(7 sec)	19 hours	19 years	183,758 years
ZIP (256bit AES)	4 min	24 days	613 years	5,649,992 years
DOCX (Office2007)	1.5 hours	1.5 years	14,230 years	131,145,907 years
DOCX (Office2010)	3 hours	3 years	28,237 years	260,235,830 years
DOCX (Office2013)	2 days	73 years	678,786 years	~Next big bang~

[1] Benchmark by Passcovery Suite version 3.10 build 3386 x64 / Corei7-4710HQ @ 2.50GHz / GeForce GTX 860M (640 SP/ALU) @ 1019Mhz

Password consists of 93 letters which include Latin small characters and Latin capital characters and digits and special symbols characters. If the password length is 8. Office twenty thirteen DOCX's time required of brute force attacking was about Sixty-seven million years. If the password length is 10, you will not be able to decrypt even coming the next Big Bang.

EncryptionInfo (XML format) Office 2010~

```
<encryption>
  <keyData saltSize="16" blockSize="16" keyBits="256" hashSize="64"
    cipherAlgorithm="AES" cipherChaining="ChainingModeCBC" hashAlgorithm="SHA512"
    saltValue="..."/>
  <dataIntegrity encryptedHmacKey="..." encryptedHmacValue="..."/>
  <keyEncryptors>
    <keyEncryptor
uri="http://schemas.microsoft.com/office/2006/keyEncryptor/password">
      <p:encryptedKey spinCount="100000"
        saltSize="16" blockSize="16" keyBits="256" hashSize="64" cipherAlgorithm="AES"
        cipherChaining="ChainingModeCBC" hashAlgorithm="SHA512" saltValue="..."
        encryptedVerifierHashInput="..."
        encryptedVerifierHashValue="..."
        encryptedKeyValue="..."/>
    </keyEncryptor>
  </keyEncryptors>
</encryption>
```



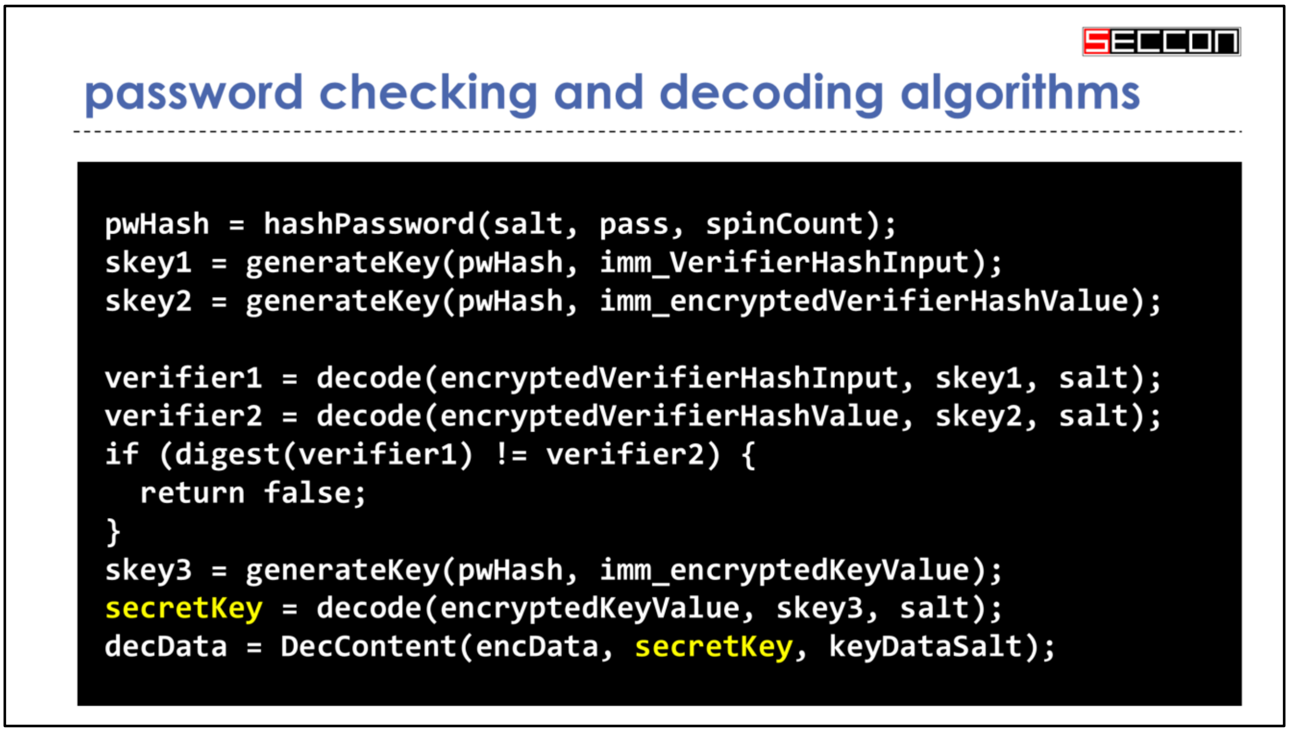
Agile Encryption has the following attributes

1. encryptedVerifierHashInput
2. encryptedVerifierHashValue
3. encryptedKeyValue
4. saltValue
5. spinCount

spinCount

1. Set this attribute to the number of times to iterate the password hash when creating the key used to encrypt the encryptedVerifierHashInput, encryptedVerifierHashValue, and encryptedKeyValue.
2. It MUST conform to a SpinCount type.

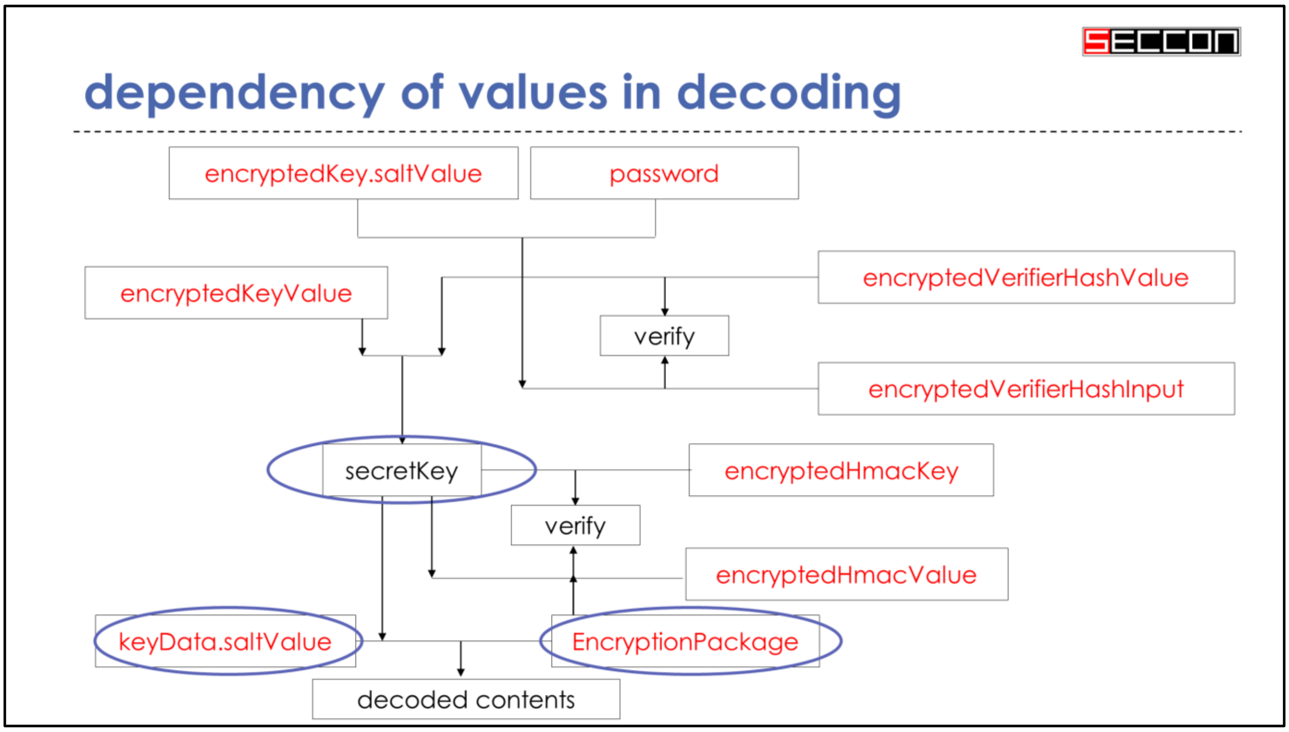
[1] https://msdn.microsoft.com/en-us/library/dd950165(v=office.12).aspx



password checking and decoding algorithms

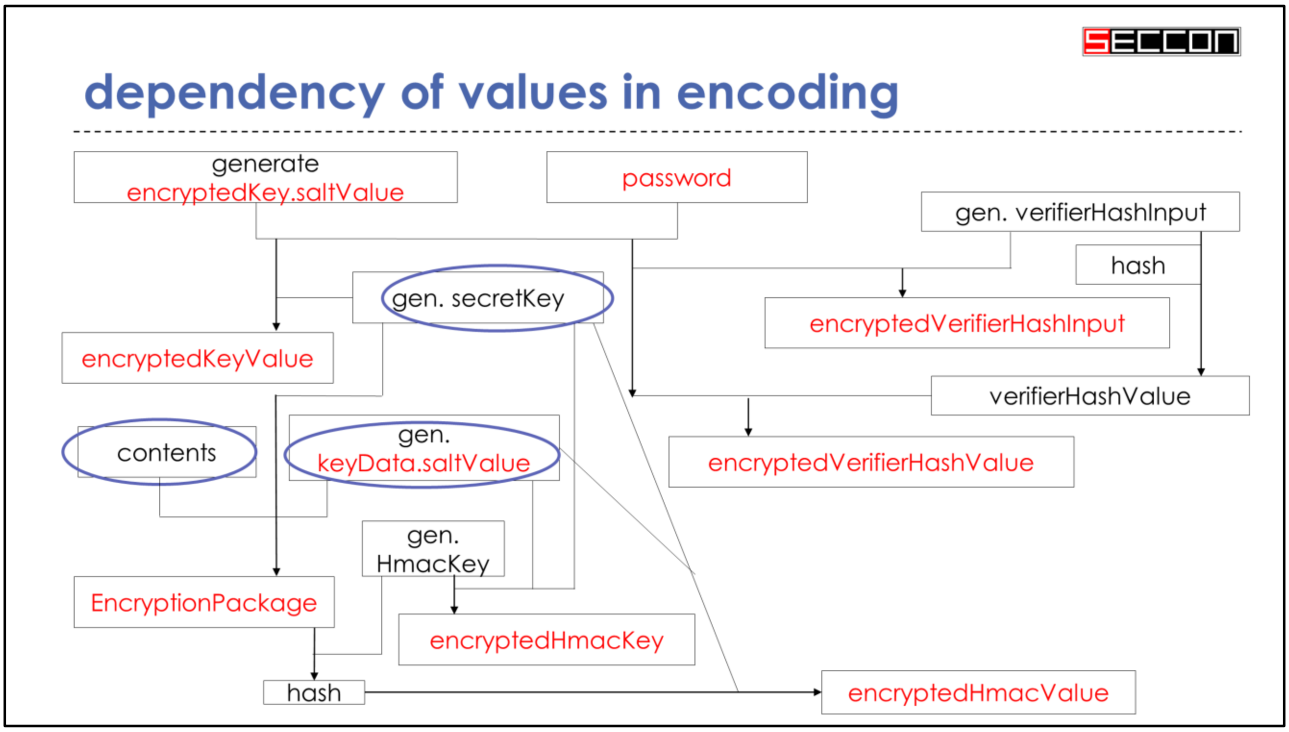
```
pwHash = hashPassword(salt, pass, spinCount);
skey1 = generateKey(pwHash, imm_VerifierHashInput);
skey2 = generateKey(pwHash, imm_encryptedVerifierHashValue);

verifier1 = decode(encryptedVerifierHashInput, skey1, salt);
verifier2 = decode(encryptedVerifierHashValue, skey2, salt);
if (digest(verifier1) != verifier2) {
  return false;
}
skey3 = generateKey(pwHash, imm_encryptedKeyValue);
secretKey = decode(encryptedKeyValue, skey3, salt);
decData = DecContent(encData, secretKey, keyDataSalt);
```

This program code is password checking and decoding algorithm. Please attention the line that the secretkey is used. decData is dependent on only secretkey and keydatasalt. It is not dependent on password.



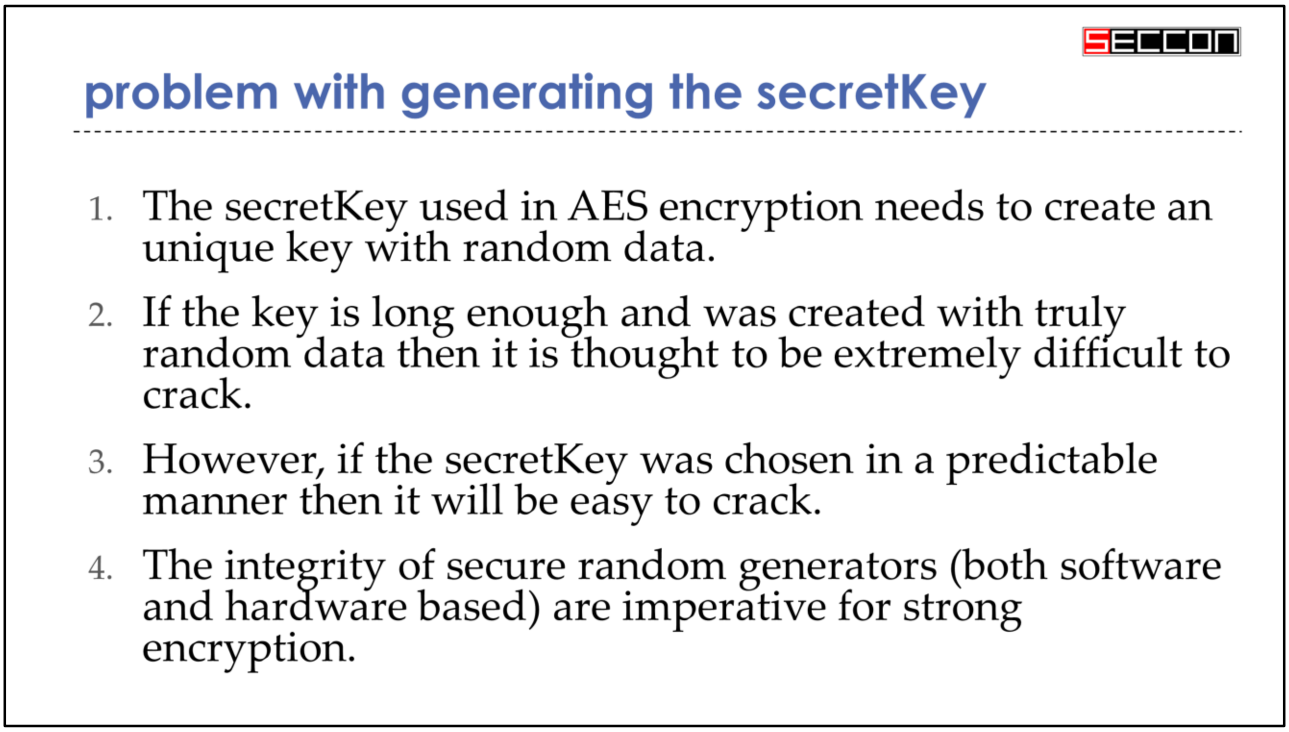
dependency of values in decoding

This is a figure of dependency of values in decoding.
Decoded contents is dependent on only secretkey and keydata.saltValuy. It is not dependent on password. I think that it is a problem.



dependency of values in encoding

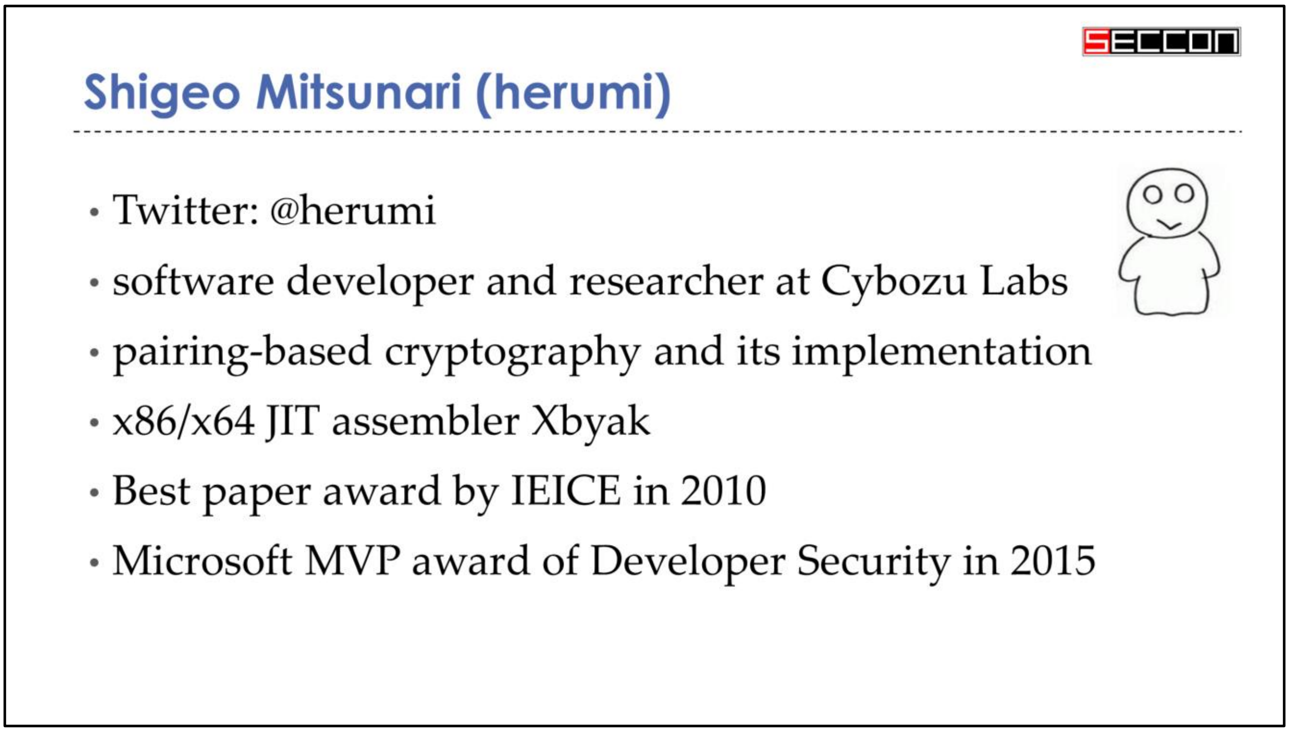
This is a figure of dependency of values in encoding.
There are problem with generating the secretKey.
The secretKey used in AES encryption needs to create an unique key with random data.



problem with generating the secretKey

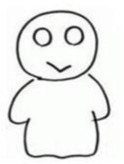
1. The secretKey used in AES encryption needs to create an unique key with random data.
2. If the key is long enough and was created with truly random data then it is thought to be extremely difficult to crack.
3. However, if the secretKey was chosen in a predictable manner then it will be easy to crack.
4. The integrity of secure random generators (both software and hardware based) are imperative for strong encryption.

If the key is long enough and was created with truly random data then it is thought to be extremely difficult to crack.
However, if the secretKey was chosen in a predictable manner then it will be easy to crack.
The integrity of secure random generators (both software and hardware based) are imperative for strong encryption.

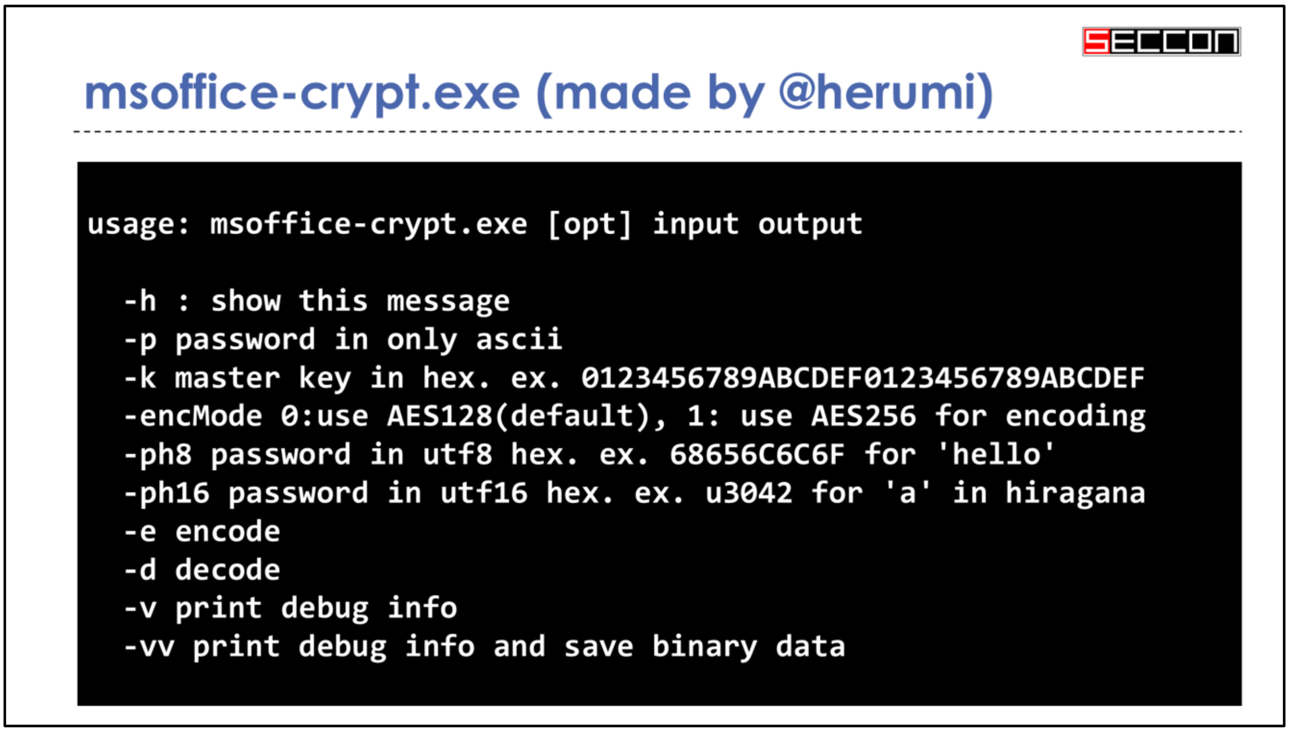


Shigeo Mitsunari (herumi)

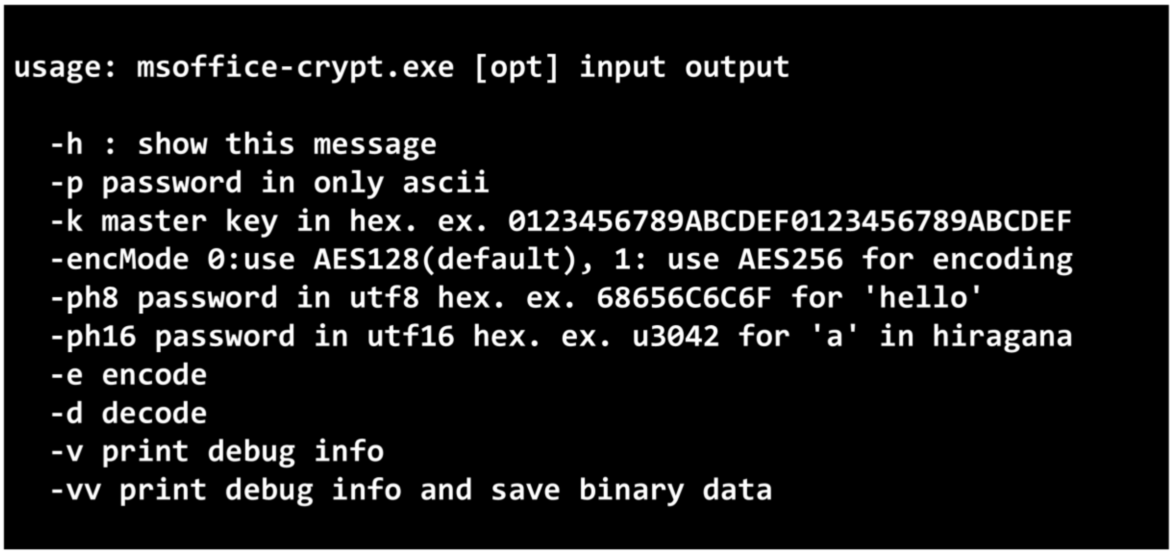
- Twitter: @herumi
- software developer and researcher at Cybozu Labs
- pairing-based cryptography and its implementation
- x86/x64 JIT assembler Xbyak
- Best paper award by IEICE in 2010
- Microsoft MVP award of Developer Security in 2015



I would like to introduce my friend. Shigeo Mitsunari is a software developer and researcher at Cybozu Labs company. He developed this msoffice-crypt.exe tools.
I was working together with him. He is a co-author of this paper.

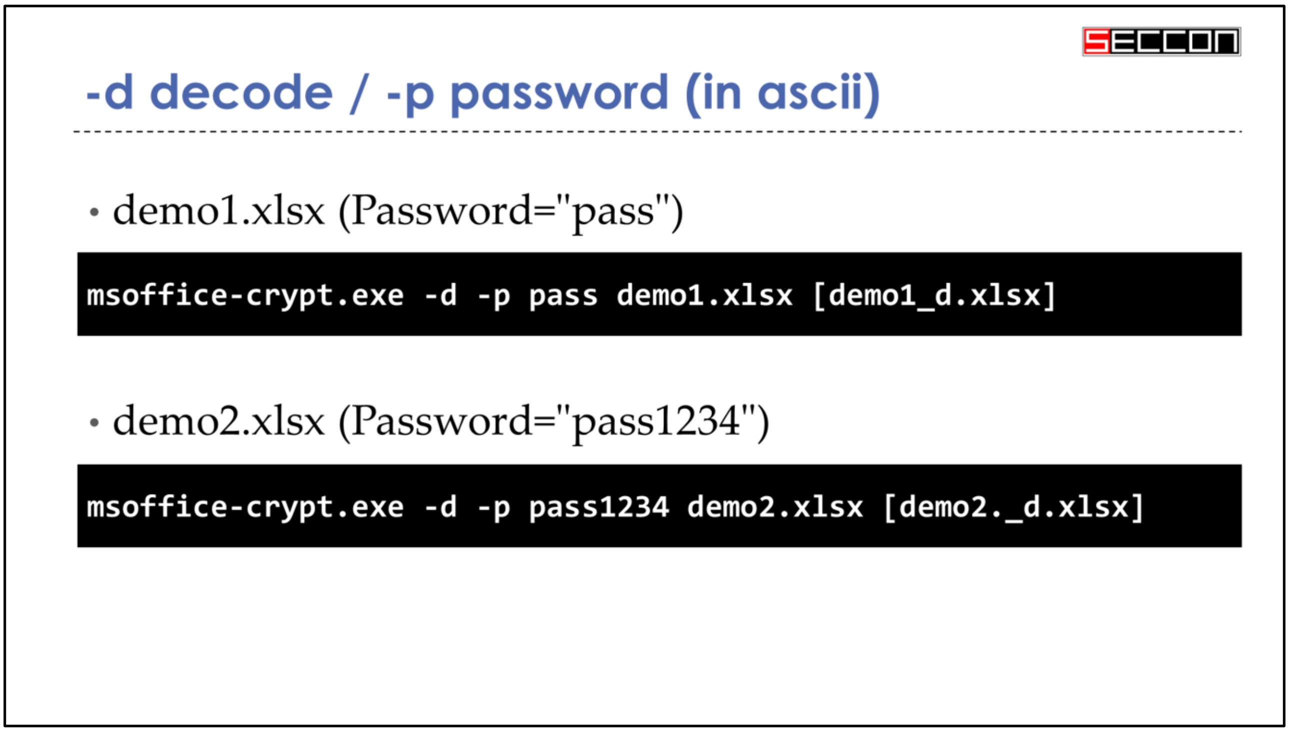


msoffice-crypt.exe (made by @herumi)



```
usage: msoffice-crypt.exe [opt] input output

  -h : show this message
  -p password in only ascii
  -k master key in hex. ex. 0123456789ABCDEF0123456789ABCDEF
  -encMode 0:use AES128(default), 1: use AES256 for encoding
  -ph8 password in utf8 hex. ex. 68656C6C6F for 'hello'
  -ph16 password in utf16 hex. ex. u3042 for 'a' in hiragana
  -e encode
  -d decode
  -v print debug info
  -vv print debug info and save binary data
```

We made the encryption and decryption tools for new Microsoft Office DOCX and XLSX and PPTX files.



-d decode / -p password (in ascii)

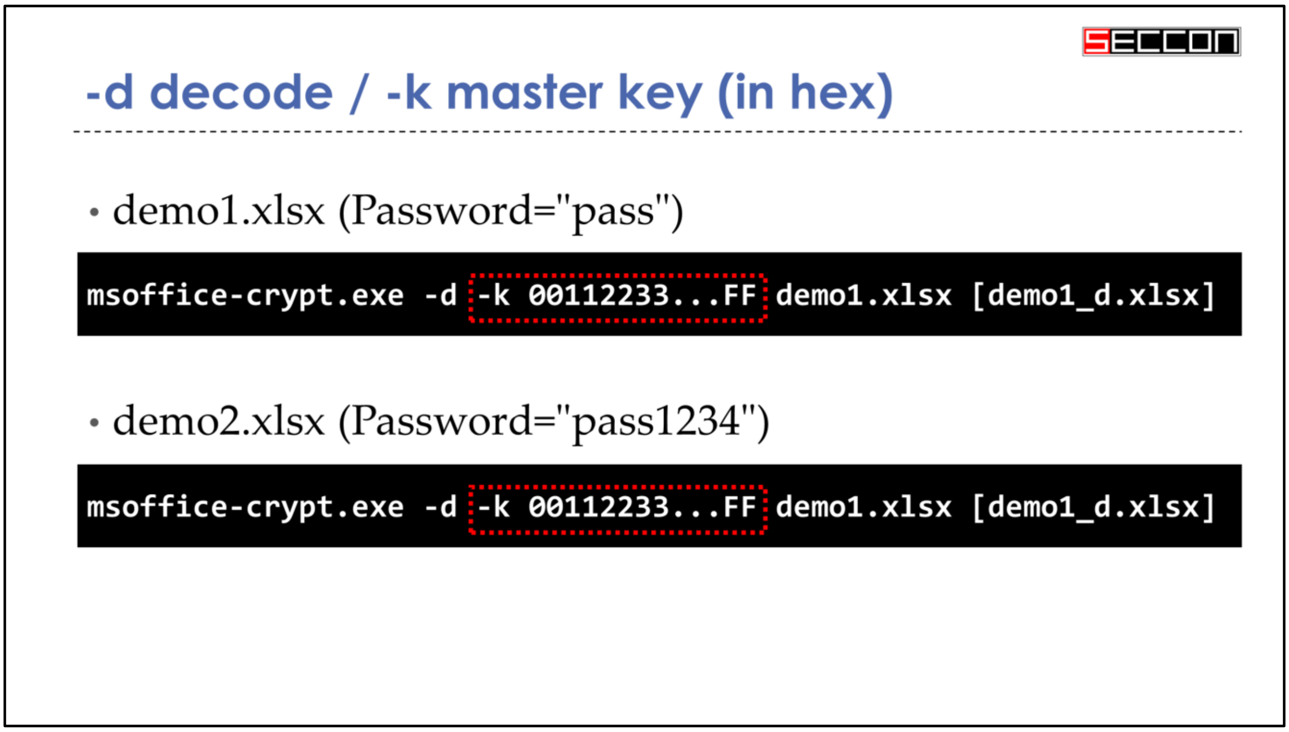
- demo1.xlsx (Password="pass")

```
msoffice-crypt.exe -d -p pass demo1.xlsx [demo1_d.xlsx]
```

- demo2.xlsx (Password="pass1234")

```
msoffice-crypt.exe -d -p pass1234 demo2.xlsx [demo2._d.xlsx]
```

msoffice-crypt command has a decode with inputed password options. It is –d and –p.



-d decode / -k master key (in hex)

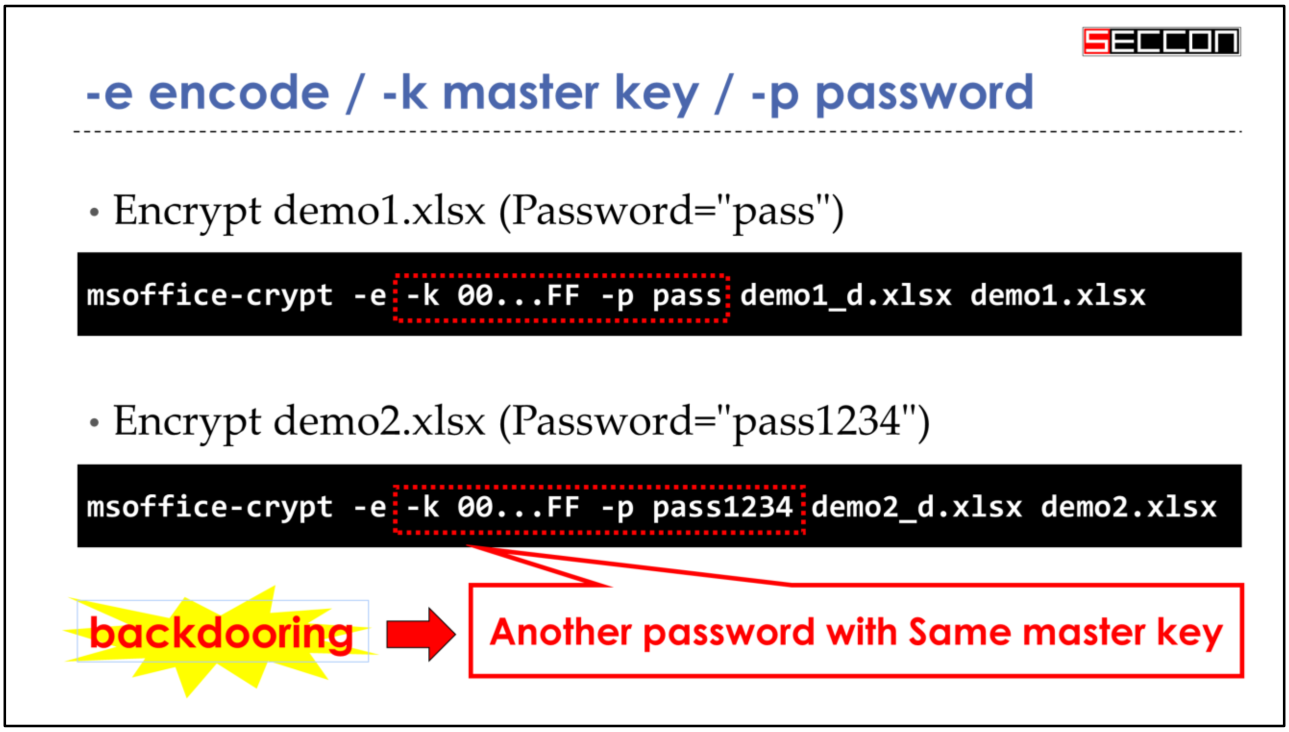
- demo1.xlsx (Password="pass")

```
msoffice-crypt.exe -d -k 00112233...FF demo1.xlsx [demo1_d.xlsx]
```

- demo2.xlsx (Password="pass1234")

```
msoffice-crypt.exe -d -k 00112233...FF demo1.xlsx [demo1_d.xlsx]
```

We made a decode with master key options. Is is –k.



-e encode / -k master key / -p password

- Encrypt demo1.xlsx (Password="pass")

```
msoffice-crypt -e -k 00...FF -p pass demo1_d.xlsx demo1.xlsx
```

- Encrypt demo2.xlsx (Password="pass1234")

```
msoffice-crypt -e -k 00...FF -p pass1234 demo2_d.xlsx demo2.xlsx
```

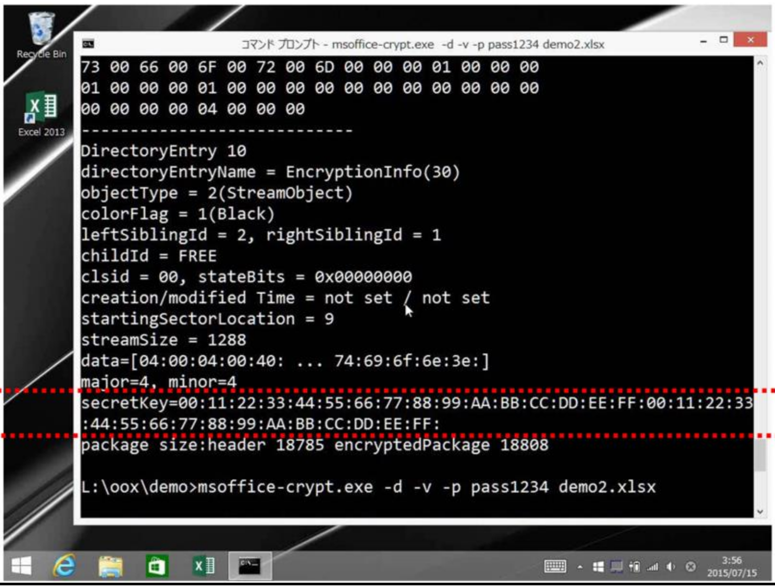
backdooring → Another password with Same master key

And we made a encode option. It is –e. Then we have –e and –k and –p options. We can make two encrypted files by another password with same master key. It is a backdoor.

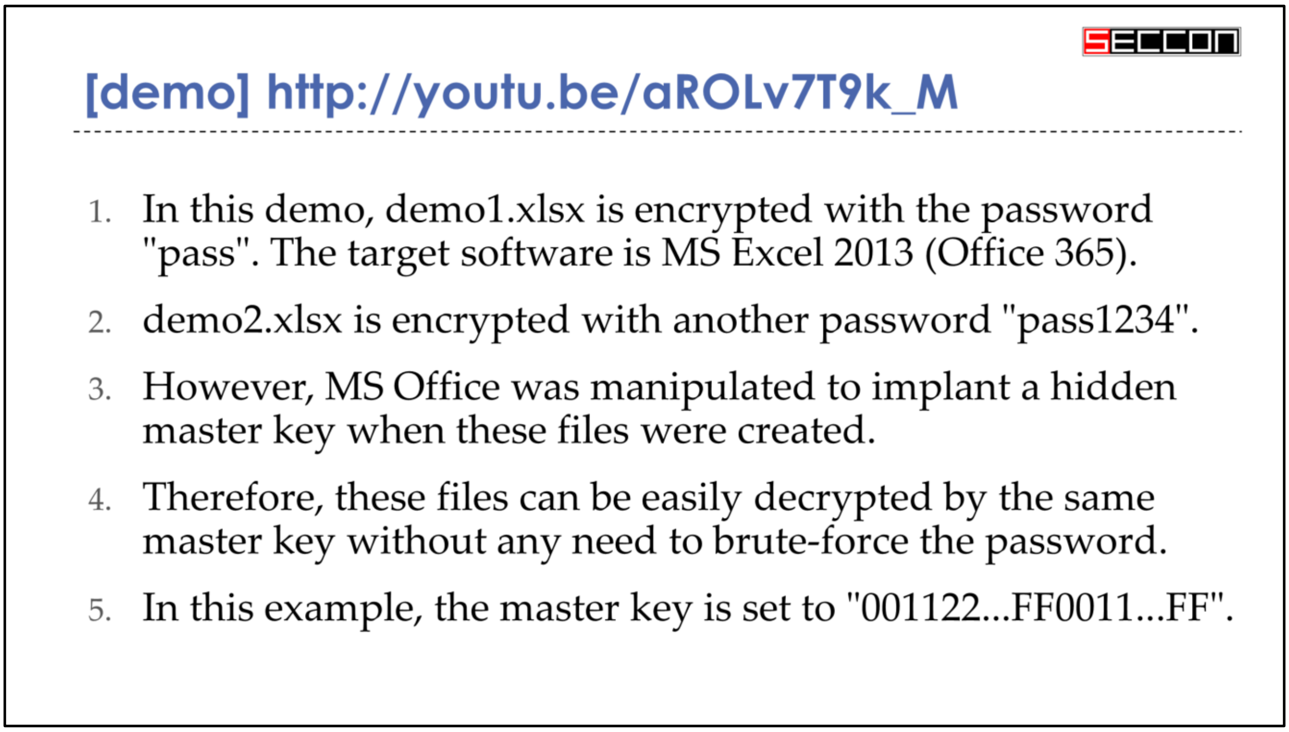
password checking and decoding algorithms

```
pwHash = hashPassword(salt, pass, spinCount);
skey1 = generateKey(pwHash, imm_VerifierHashInput);
skey2 = generateKey(pwHash, imm_encryptedVerifierHashValue);

verifier1 = decode(encryptedVerifierHashInput, skey1, salt);
verifier2 = decode(encryptedVerifierHashValue, skey2, salt);
if (digest(verifier1) != verifier2) {
  return false;
}
skey3 = generateKey(pwHash, imm_encryptedKeyValue);
secretKey = decode(encryptedKeyValue, skey3, salt);
decData = DecContent(encData, secretKey, keyDataSalt);
```

Proof of Concept


Recycle Bin
Excel 2013
コマンド プロンプト - msoffice-crypt.exe -d -v -p pass1234 demo2.xlsx
3:56
2015/07/15

```
73 00 66 00 6F 00 72 00 6D 00 00 00 01 00 00 00
01 00 00 00 01 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 04 00 00 00
----------------------------
DirectoryEntry 10
directoryEntryName = EncryptionInfo(30)
objectType = 2(StreamObject)
colorFlag = 1(Black)
leftSiblingId = 2, rightSiblingId = 1
childId = FREE
clsid = 00, stateBits = 0x00000000
creation/modified Time = not set / not set
startingSectorLocation = 9
streamSize = 1288
data=[04:00:04:00:40: ... 74:69:6f:6e:3e:]
major=4, minor=4
secretKey=00:11:22:33:44:55:66:77:88:99:AA:BB:CC:DD:EE:FF:00:11:22:33
:44:55:66:77:88:99:AA:BB:CC:DD:EE:FF:
package size:header 18785 encryptedPackage 18808

L:\oox\demo>msoffice-crypt.exe -d -v -p pass1234 demo2.xlsx
```

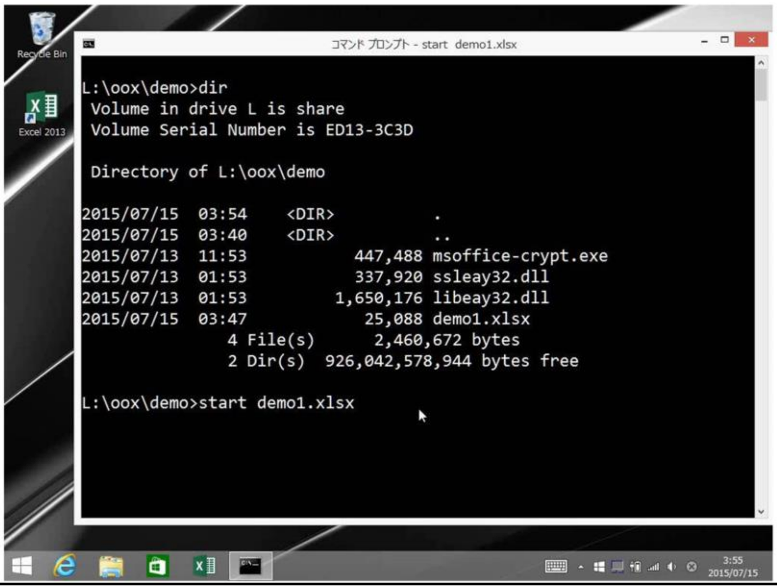


[demo] http://youtu.be/aROLv7T9k_M

1. In this demo, demo1.xlsx is encrypted with the password "pass". The target software is MS Excel 2013 (Office 365).
2. demo2.xlsx is encrypted with another password "pass1234".
3. However, MS Office was manipulated to implant a hidden master key when these files were created.
4. Therefore, these files can be easily decrypted by the same master key without any need to brute-force the password.
5. In this example, the master key is set to "001122...FF0011...FF".

1. In this demo, demo1.xlsx is encrypted with the password "pass". The target software is MS Excel twenty-thirteen.
2. demo2.xlsx is encrypted with another password "pass1234".
3. However, MS Office was manipulated to implant a hidden master key when these files were created.
4. Therefore, these files can be easily decrypted by the same master key without any need to brute-force the password.
5. In this example, the master key is set to "001122...FF0011...FF".

[demo] http://youtu.be/aROLv7T9k_M

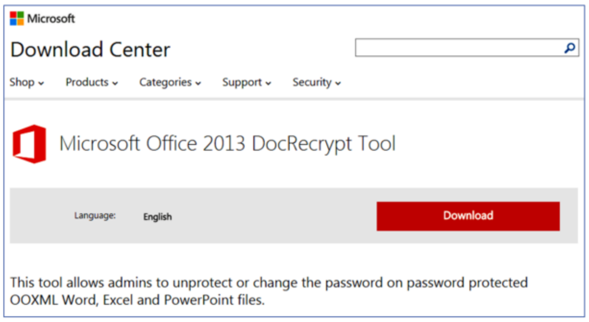


```
L:\oox\demo>dir
 Volume in drive L is share
 Volume Serial Number is ED13-3C3D

 Directory of L:\oox\demo

2015/07/15  03:54    <DIR>          .
2015/07/15  03:40    <DIR>          ..
2015/07/13  11:53           447,488 msoffice-crypt.exe
2015/07/13  01:53           337,920 ssleay32.dll
2015/07/13  01:53         1,650,176 libeay32.dll
2015/07/15  03:47            25,088 demo1.xlsx
               4 File(s)      2,460,672 bytes
               2 Dir(s)  926,042,578,944 bytes free

L:\oox\demo>start demo1.xlsx
```

SECCON

Microsoft Office 2013 DocRecrypt Tool (official)

- IT admin can "unlock" the password-protected OOXML Word, Excel and PowerPoint files for a user and then either leave the file without password protection! (it is official)

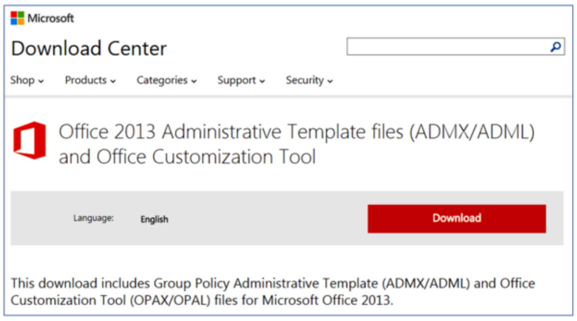


Microsoft Office 2013 DocRecrypt Tool

Language: English

Download

This tool allows admins to unprotect or change the password on password protected OOXML Word, Excel and PowerPoint files.



Office 2013 Administrative Template files (ADMX/ADML) and Office Customization Tool

Language: English

Download

This download includes Group Policy Administrative Template (ADMX/ADML) and Office Customization Tool (OPAX/OPAL) files for Microsoft Office 2013.

[1] https://www.microsoft.com/en-us/download/details.aspx?id=36443

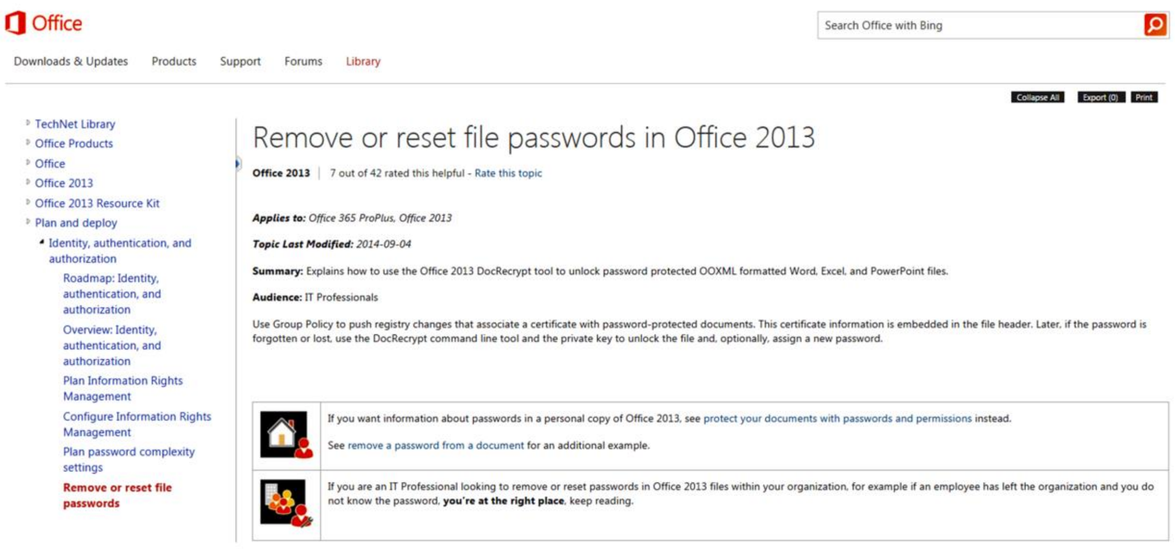
IT admin can "unlock" the password-protected OOXML Word, Excel and PowerPoint files for a user and then either leave the file without password protection! (it is official)

By using Office 2013 and an escrow key

1. You the IT admin, are the keeper of the **escrow key** which is generated from your company or organization's **private key** certificate store.
2. You can silently push the **public key** information to client computers one time through a registry key setting.
3. When a user later creates password-protected Office 2013 files, this public key is included in the file header.
4. IT admin can use the Office **DocRecrypt** tool to remove the password that is attached to the file by using your company's **private key**.

[1] https://technet.microsoft.com/en-US/library/jj923033.aspx

An attacker can exploit this IT admin's function


Office

Search Office with Bing

Downloads & Updates Products Support Forums Library

Collapse All Export (0) Print

- TechNet Library
- Office Products
- Office
- Office 2013
- Office 2013 Resource Kit
- Plan and deploy
  - Identity, authentication, and authorization
    - Roadmap: Identity, authentication, and authorization
    - Overview: Identity, authentication, and authorization
    - Plan Information Rights Management
    - Configure Information Rights Management
    - Plan password complexity settings
    - **Remove or reset file passwords**

Remove or reset file passwords in Office 2013

Office 2013 | 7 out of 42 rated this helpful - Rate this topic

Applies to: *Office 365 ProPlus, Office 2013*

Topic Last Modified: *2014-09-04*

Summary: Explains how to use the Office 2013 DocRecrypt tool to unlock password protected OOXML formatted Word, Excel, and PowerPoint files.

Audience: IT Professionals

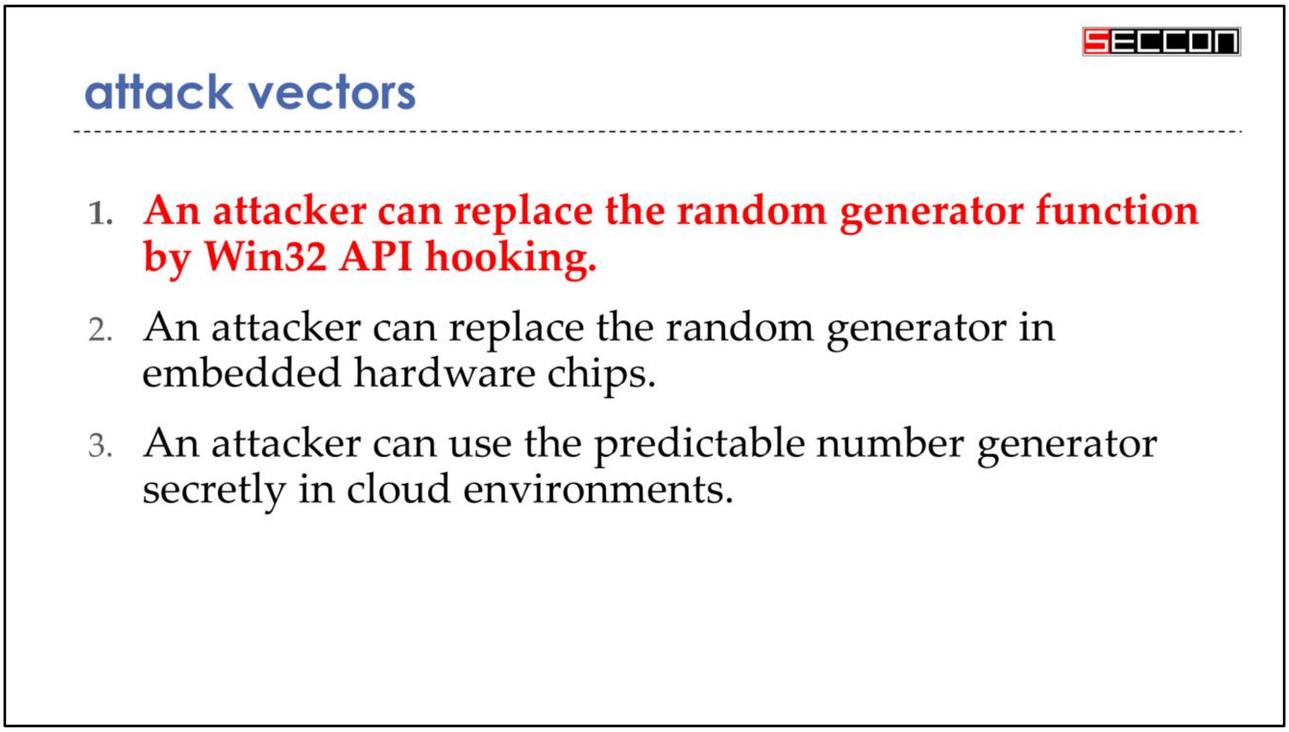
Use Group Policy to push registry changes that associate a certificate with password-protected documents. This certificate information is embedded in the file header. Later, if the password is forgotten or lost, use the DocRecrypt command line tool and the private key to unlock the file and, optionally, assign a new password.

If you want information about passwords in a personal copy of Office 2013, see protect your documents with passwords and permissions instead.

See remove a password from a document for an additional example.

If you are an IT Professional looking to remove or reset passwords in Office 2013 files within your organization, for example if an employee has left the organization and you do not know the password, **you're at the right place**, keep reading.

[1] https://technet.microsoft.com/en-US/library/jj923033.aspx

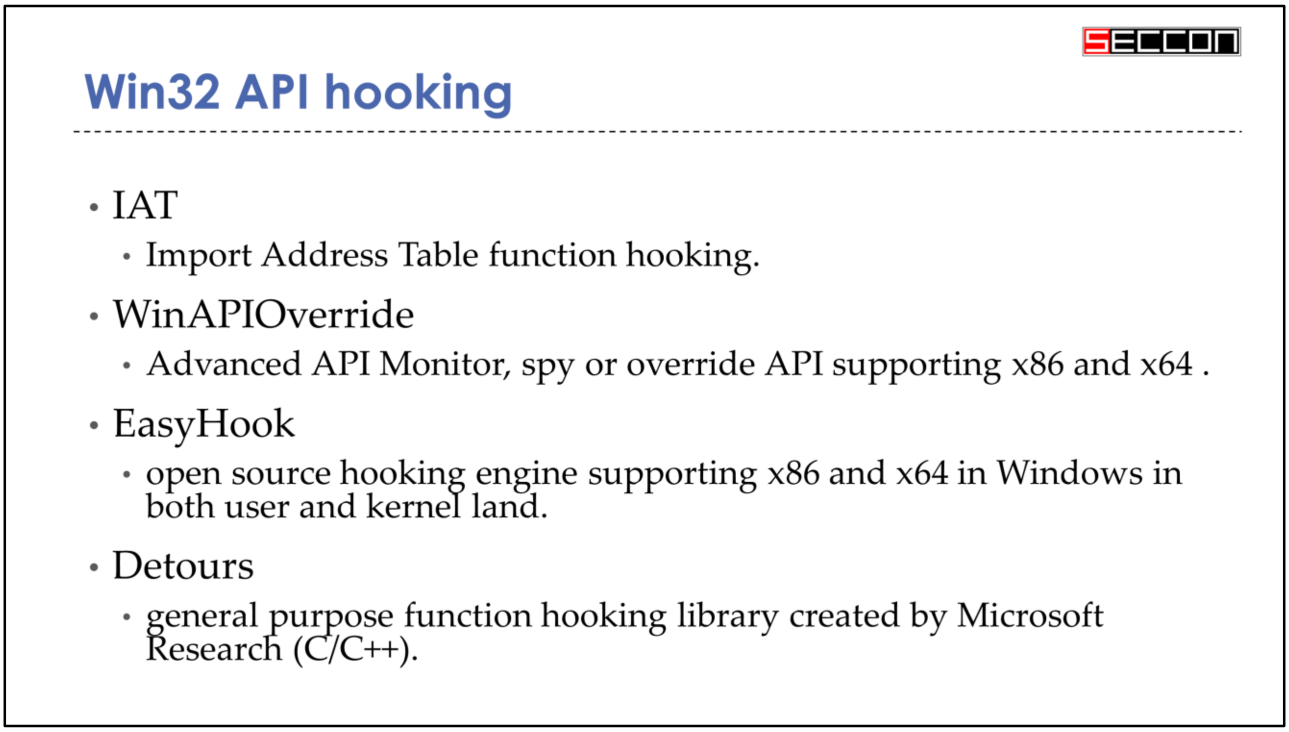


SECCON

attack vectors

1. **An attacker can replace the random generator function by Win32 API hooking.**
2. An attacker can replace the random generator in embedded hardware chips.
3. An attacker can use the predictable number generator secretly in cloud environments.

1st attack vector is that some attacker can replace the random generator function by Win32 API hooking.

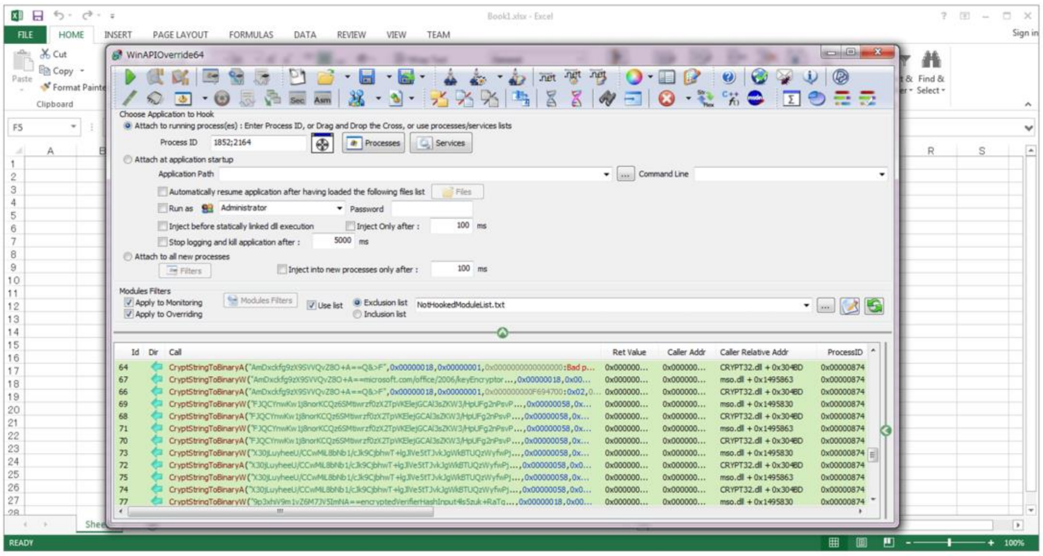

SECCON

Win32 API hooking

- IAT
  - Import Address Table function hooking.
- WinAPIOverride
  - Advanced API Monitor, spy or override API supporting x86 and x64 .
- EasyHook
  - open source hooking engine supporting x86 and x64 in Windows in both user and kernel land.
- Detours
  - general purpose function hooking library created by Microsoft Research (C/C++).

There are so many API hooking techniques. IAT Import Address Table function hooking is one of the famous Windows API hooking techniques.

WinAPIOverride32 / WinAPIOverride64



Choose Application to Hook

(●) Attach to running process(es) : Enter Process ID, or Drag and Drop the Cross, or use processes/services lists

Process ID: 1852;2164 | Processes | Services

( ) Attach at application startup

Application Path: | Command Line:

[ ] Automatically resume application after having loaded the following files list | Files

[ ] Run as: Administrator | Password

[ ] Inject before statically linked dll execution | [ ] Inject Only after : 100 ms

[ ] Stop logging and kill application after : 5000 ms

( ) Attach to all new processes | Filters | [ ] Inject into new processes only after : 100 ms

Modules Filters

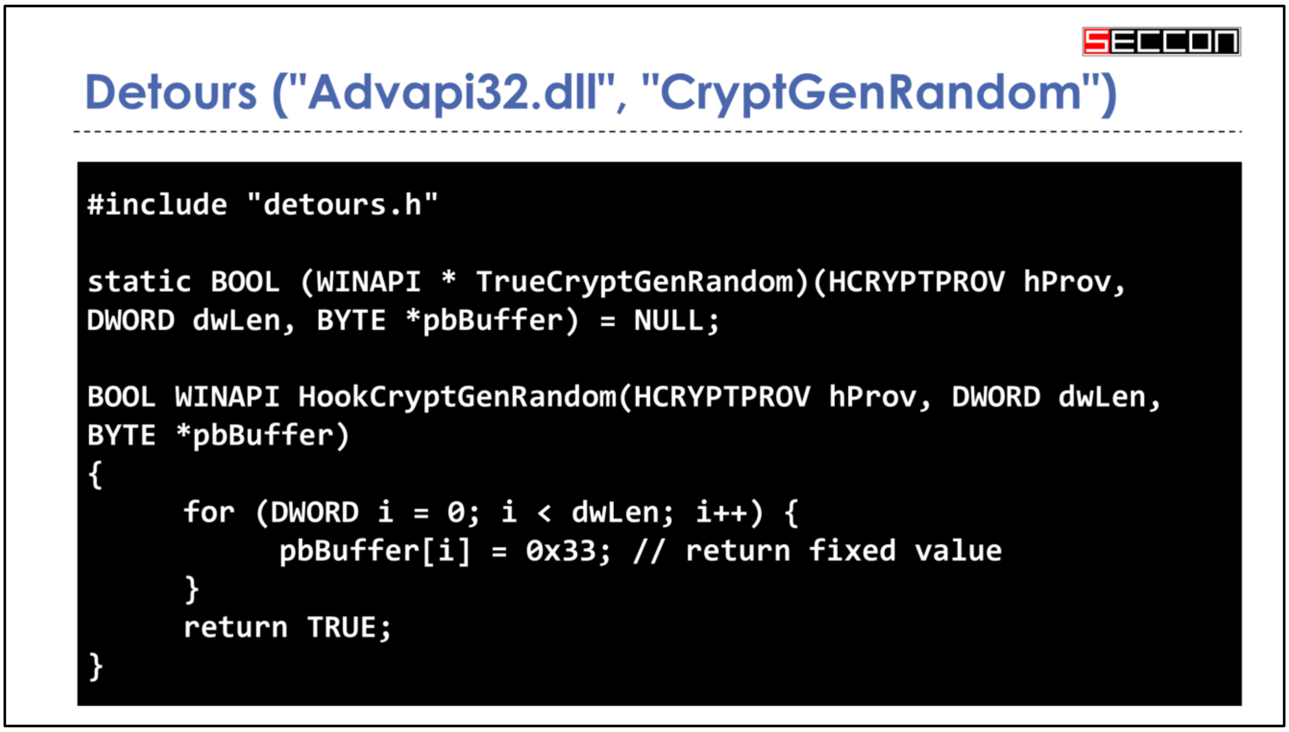
[✓] Apply to Monitoring | Modules Filters | [✓] Use list | (●) Exclusion list | NotHookedModuleList.txt

[✓] Apply to Overriding | ( ) Inclusion list

Id	Dir	Call	Ret Value	Caller Addr	Caller Relative Addr	ProcessID
64		CryptStringToBinaryA("AmDxckfg9zX9SVVQvZ8O+A==Q&>F",0x00000018,0x00000001,0x0000000000000000:Bad p...	0x000000...	0x000000...	CRYPT32.dll + 0x304BD	0x00000874
67		CryptStringToBinaryW("AmDxckfg9zX9SVVQvZ8O+A==microsoft.com/office/2006/keyEncryptor...,0x00000018,0x00...	0x000000...	0x000000...	mso.dll + 0x1495863	0x00000874
66		CryptStringToBinaryA("AmDxckfg9zX9SVVQvZ8O+A==Q&>F",0x00000018,0x00000001,0x000000000F694700:0x02,0...	0x000000...	0x000000...	CRYPT32.dll + 0x304BD	0x00000874
69		CryptStringToBinaryW("FJQCYnwKw1jBnorKCQz6SMtwrzf0zX2TpVKElejGCAl3sZKW3/HpUFg2nPsvP...,0x00000058,0x...	0x000000...	0x000000...	mso.dll + 0x1495830	0x00000874
68		CryptStringToBinaryA("FJQCYnwKw1jBnorKCQz6SMtwrzf0zX2TpVKElejGCAl3sZKW3/HpUFg2nPsvP...,0x00000058,0x...	0x000000...	0x000000...	CRYPT32.dll + 0x304BD	0x00000874
71		CryptStringToBinaryW("FJQCYnwKw1jBnorKCQz6SMtwrzf0zX2TpVKElejGCAl3sZKW3/HpUFg2nPsvP...,0x00000058,0x...	0x000000...	0x000000...	mso.dll + 0x1495863	0x00000874
70		CryptStringToBinaryA("FJQCYnwKw1jBnorKCQz6SMtwrzf0zX2TpVKElejGCAl3sZKW3/HpUFg2nPsvP...,0x00000058,0x...	0x000000...	0x000000...	CRYPT32.dll + 0x304BD	0x00000874
73		CryptStringToBinaryW("X30jLuyheeU/CCwMiL8bNb1/cJk9CjbhwT+lgJlVe5tTJvkJgWkBTUQzWyfwPj...,0x00000058,0x...	0x000000...	0x000000...	mso.dll + 0x1495830	0x00000874
72		CryptStringToBinaryA("X30jLuyheeU/CCwMiL8bNb1/cJk9CjbhwT+lgJlVe5tTJvkJgWkBTUQzWyfwPj...,0x00000058,0x0...	0x000000...	0x000000...	CRYPT32.dll + 0x304BD	0x00000874
75		CryptStringToBinaryW("X30jLuyheeU/CCwMiL8bNb1/cJk9CjbhwT+lgJlVe5tTJvkJgWkBTUQzWyfwPj...,0x00000058,0x...	0x000000...	0x000000...	mso.dll + 0x1495863	0x00000874
74		CryptStringToBinaryA("X30jLuyheeU/CCwMiL8bNb1/cJk9CjbhwT+lgJlVe5tTJvkJgWkBTUQzWyfwPj...,0x00000058,0x0...	0x000000...	0x000000...	CRYPT32.dll + 0x304BD	0x00000874
77		CryptStringToBinaryW("9p3xhiV9m1vZ6M7JV5ImNA==encryptedVerifierHashInput4is5zuk+RaTq...,0x00000018,0x00...	0x000000...	0x000000...	mso.dll + 0x1495830	0x00000874

[1] http://jacquelin.potier.free.fr/winapioverride32/

And more over, there are WinAPIOverride thirty-two and sixty-four application. It's very nice software. I like it.

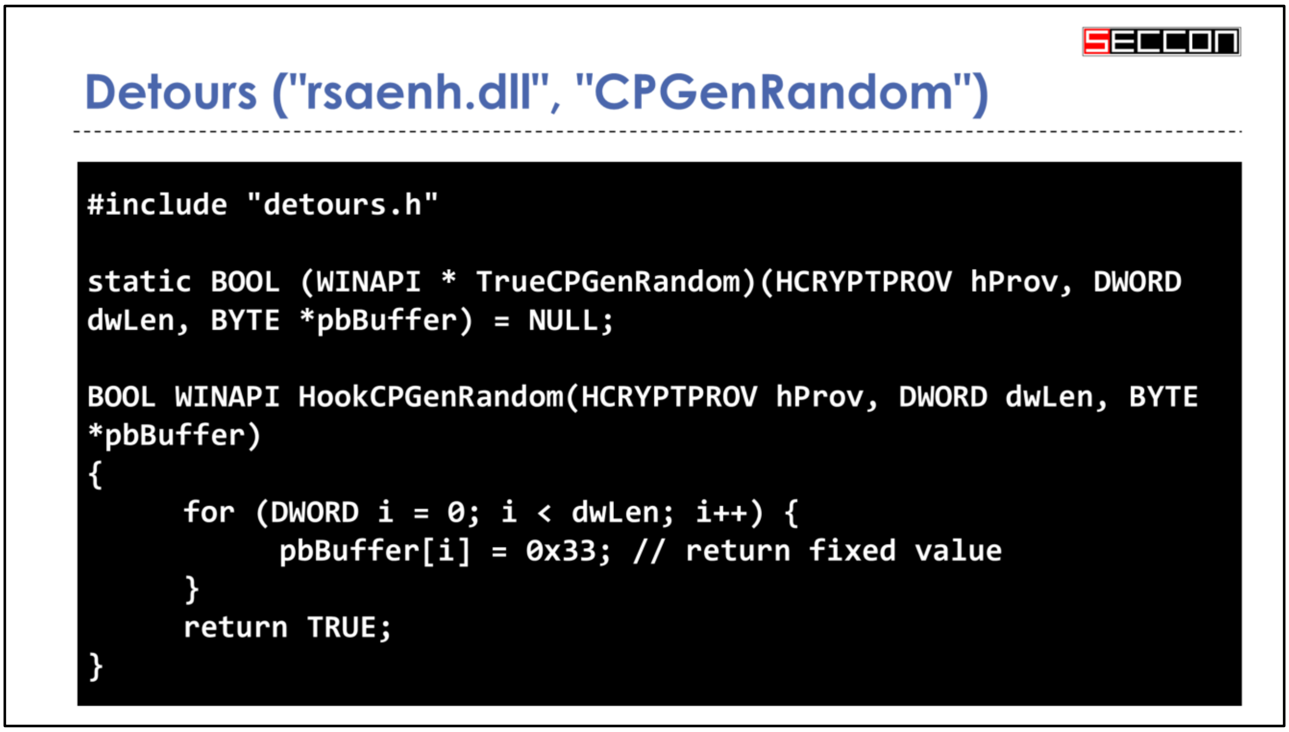


SECCON

Detours ("Advapi32.dll", "CryptGenRandom")

```
#include "detours.h"

static BOOL (WINAPI * TrueCryptGenRandom)(HCRYPTPROV hProv,
DWORD dwLen, BYTE *pbBuffer) = NULL;

BOOL WINAPI HookCryptGenRandom(HCRYPTPROV hProv, DWORD dwLen,
BYTE *pbBuffer)
{
      for (DWORD i = 0; i < dwLen; i++) {
            pbBuffer[i] = 0x33; // return fixed value
      }
      return TRUE;
}
```

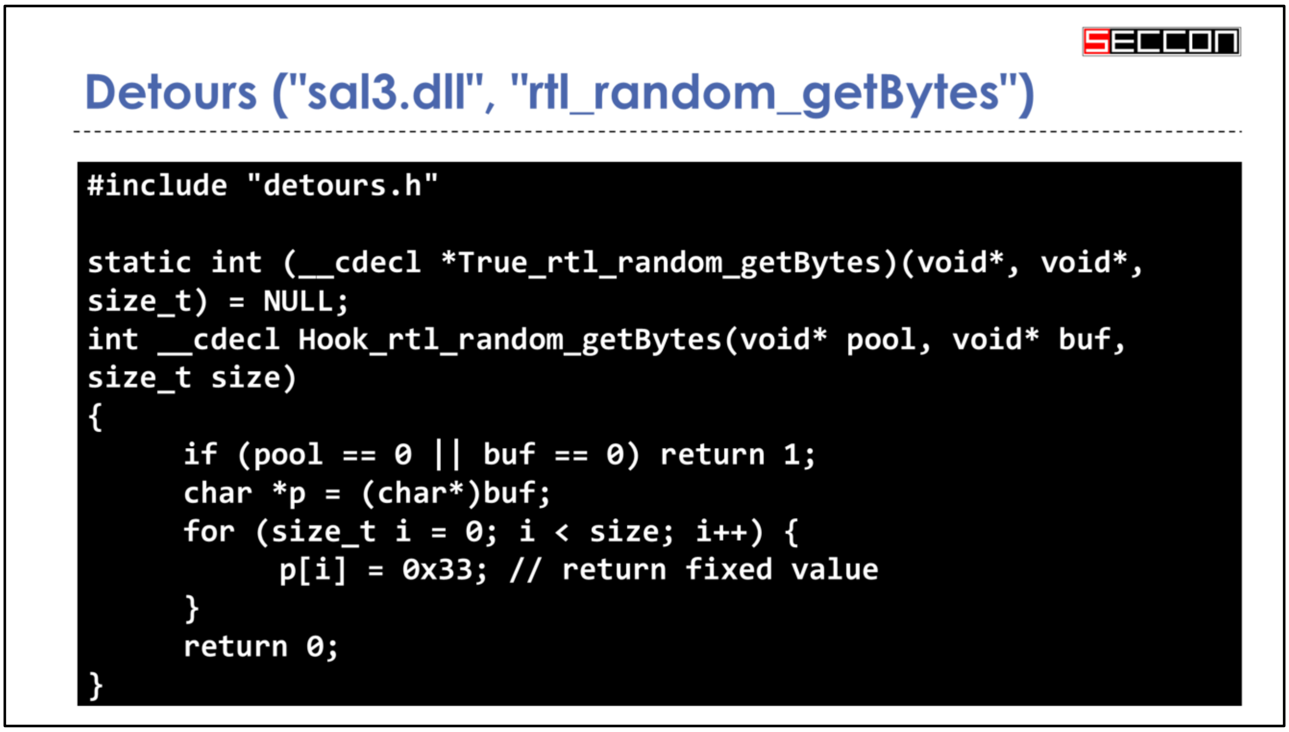
Microsoft Research created general purpose function hooking library “Detours”.
It can easily hook the application by DLL injection.
In this case, I can hook the CryptGenRandom function on Advapi32.dll. Then hooked CryptGenRandom function always return the fixed value. 0x33



Detours ("rsaenh.dll", "CPGenRandom")

```
#include "detours.h"

static BOOL (WINAPI * TrueCPGenRandom)(HCRYPTPROV hProv, DWORD
dwLen, BYTE *pbBuffer) = NULL;

BOOL WINAPI HookCPGenRandom(HCRYPTPROV hProv, DWORD dwLen, BYTE
*pbBuffer)
{
	for (DWORD i = 0; i < dwLen; i++) {
		pbBuffer[i] = 0x33; // return fixed value
	}
	return TRUE;
}
```

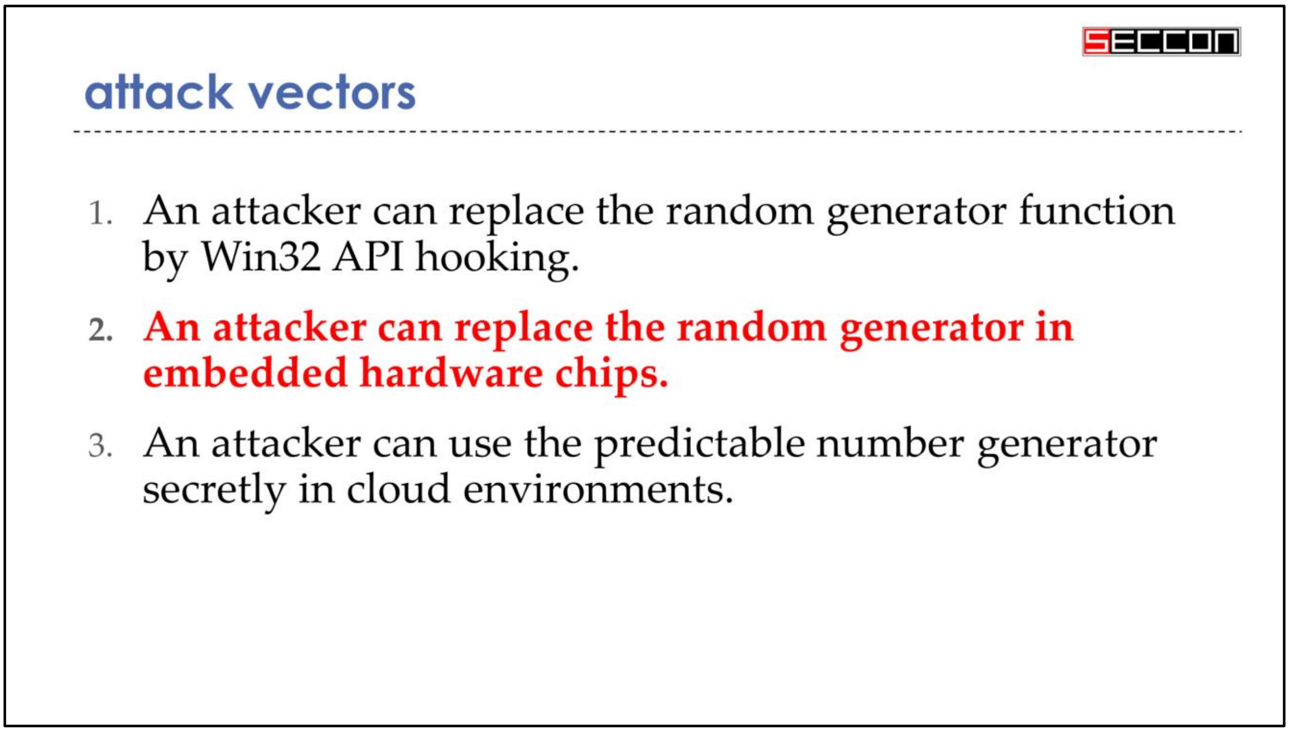
In other case, I can hook the CPGenRandom function on old Windows API. Hooked CPGenRandom function always return the fixed value which is not random value.



Detours ("sal3.dll", "rtl_random_getBytes")

```
#include "detours.h"

static int (__cdecl *True_rtl_random_getBytes)(void*, void*,
size_t) = NULL;
int __cdecl Hook_rtl_random_getBytes(void* pool, void* buf,
size_t size)
{
      if (pool == 0 || buf == 0) return 1;
      char *p = (char*)buf;
      for (size_t i = 0; i < size; i++) {
            p[i] = 0x33; // return fixed value
      }
      return 0;
}
```

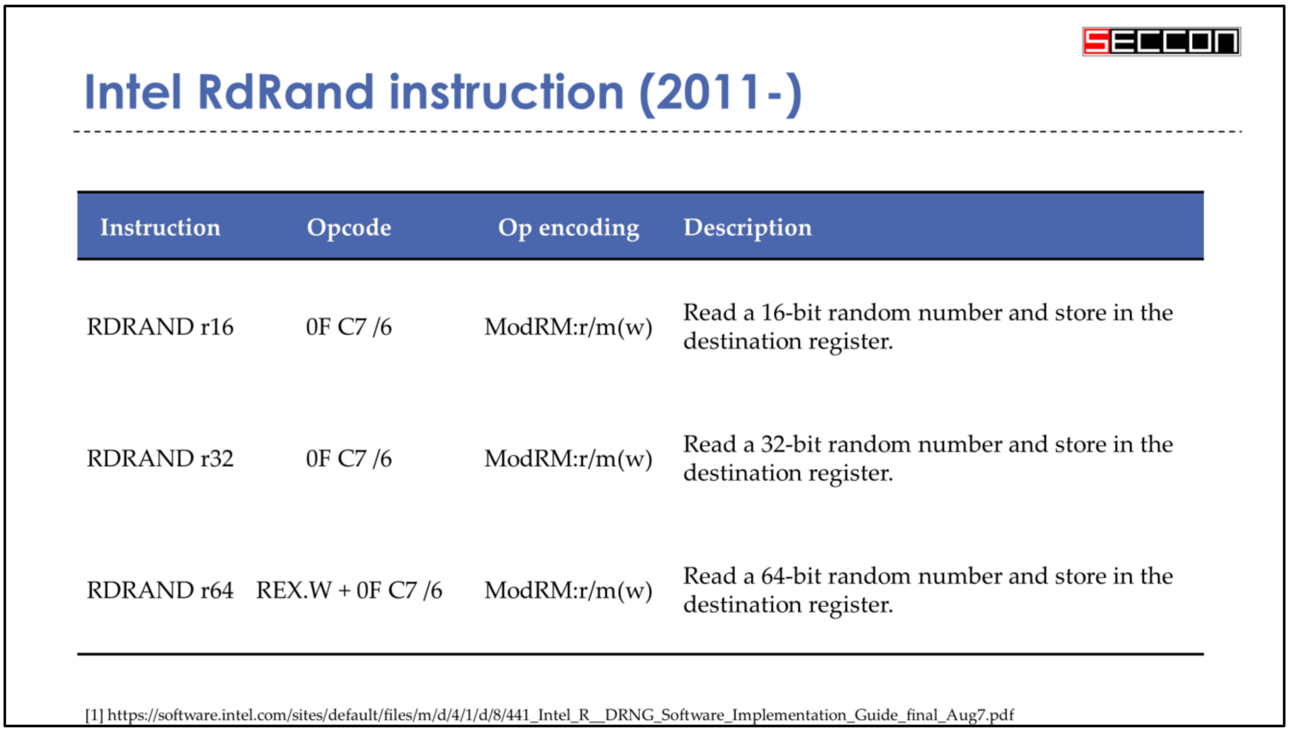
In another case, I can hook the rtl_random_getBytes function on sal3.dll which is used by LibreOffice application. I can control the randomness on my own computer.



attack vectors

1. An attacker can replace the random generator function by Win32 API hooking.
2. **An attacker can replace the random generator in embedded hardware chips.**
3. An attacker can use the predictable number generator secretly in cloud environments.

2nd attack vector is that some attackers can replace the random generator in embedded hardware chips.

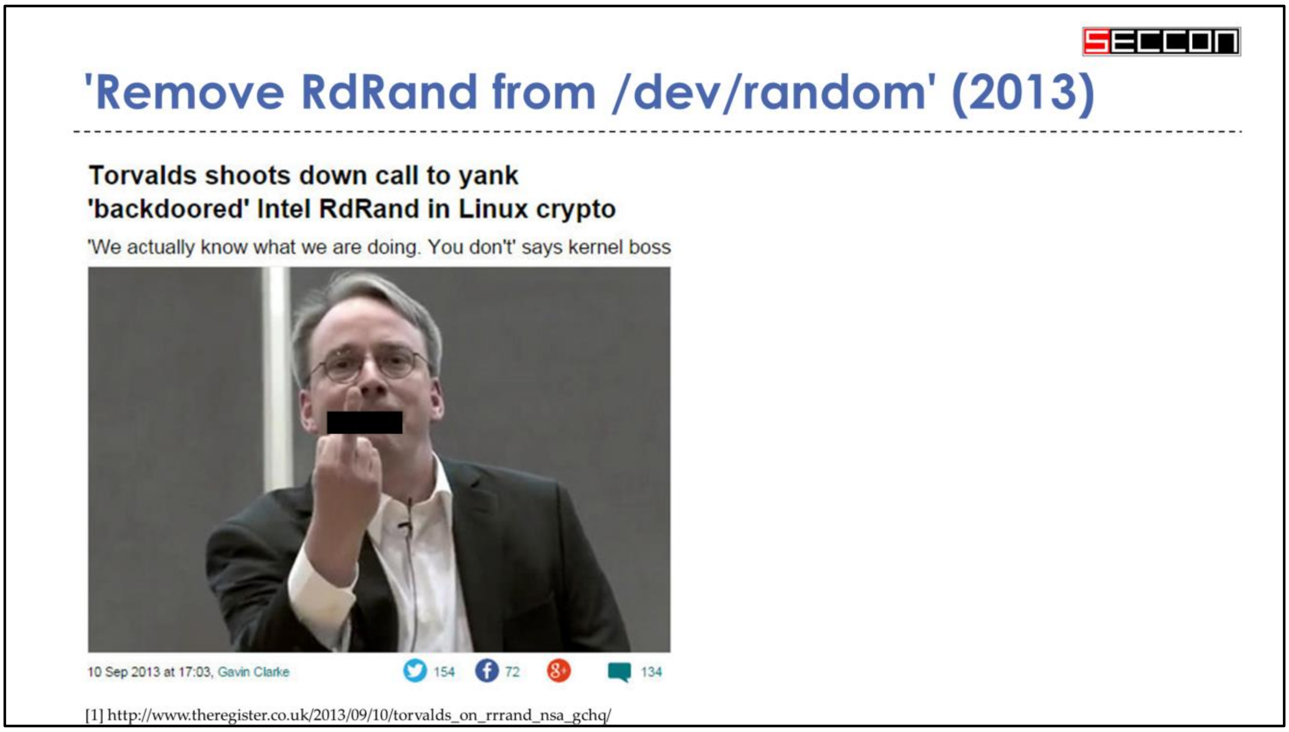


Intel RdRand instruction (2011-)

Instruction	Opcode	Op encoding	Description
RDRAND r16	0F C7 /6	ModRM:r/m(w)	Read a 16-bit random number and store in the destination register.
RDRAND r32	0F C7 /6	ModRM:r/m(w)	Read a 32-bit random number and store in the destination register.
RDRAND r64	REX.W + 0F C7 /6	ModRM:r/m(w)	Read a 64-bit random number and store in the destination register.

[1] https://software.intel.com/sites/default/files/m/d/4/1/d/8/441_Intel_R__DRNG_Software_Implementation_Guide_final_Aug7.pdf

Intel developed the RdRand instruction in the hardware chip. Core i7 so on. It generate truly random by Intel's new hardware chips.



'Remove RdRand from /dev/random' (2013)

Torvalds shoots down call to yank 'backdoored' Intel RdRand in Linux crypto

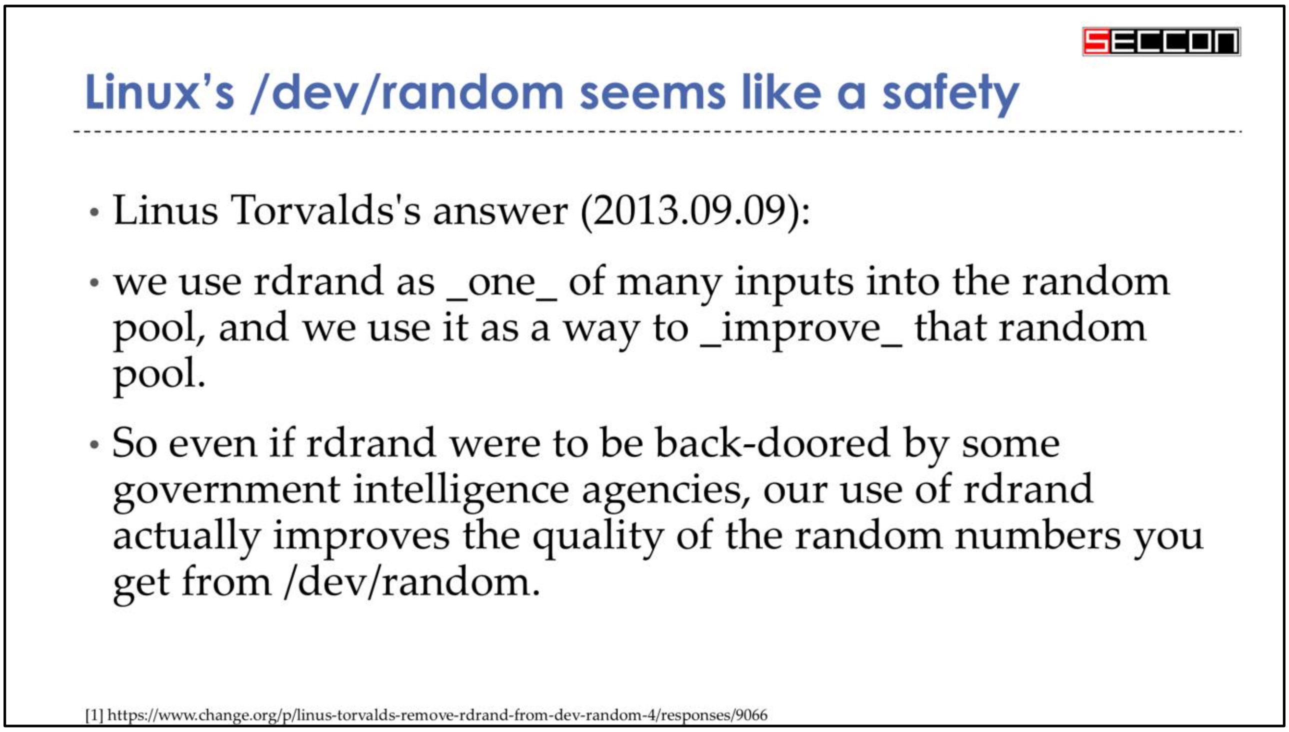
'We actually know what we are doing. You don't' says kernel boss

[redacted]

10 Sep 2013 at 17:03, Gavin Clarke 154 72 134

[1] http://www.theregister.co.uk/2013/09/10/torvalds_on_rrrand_nsa_gchq/

The pseudo-device /dev/random generates a virtually endless stream of random numbers on GNU/Linux systems. RdRand is an instruction found in modern Intel CPU chips that stashes a "high-quality and high-performance entropy" generated random number in a given CPU register. These, hopefully, unpredictable values are vital in producing secure session keys, new public-private keys and padding in modern encryption technology.
If some government intelligence agencies have managed to persuade Intel to hobble that instruction. The strength of encryption algorithms will be weak on that random data.

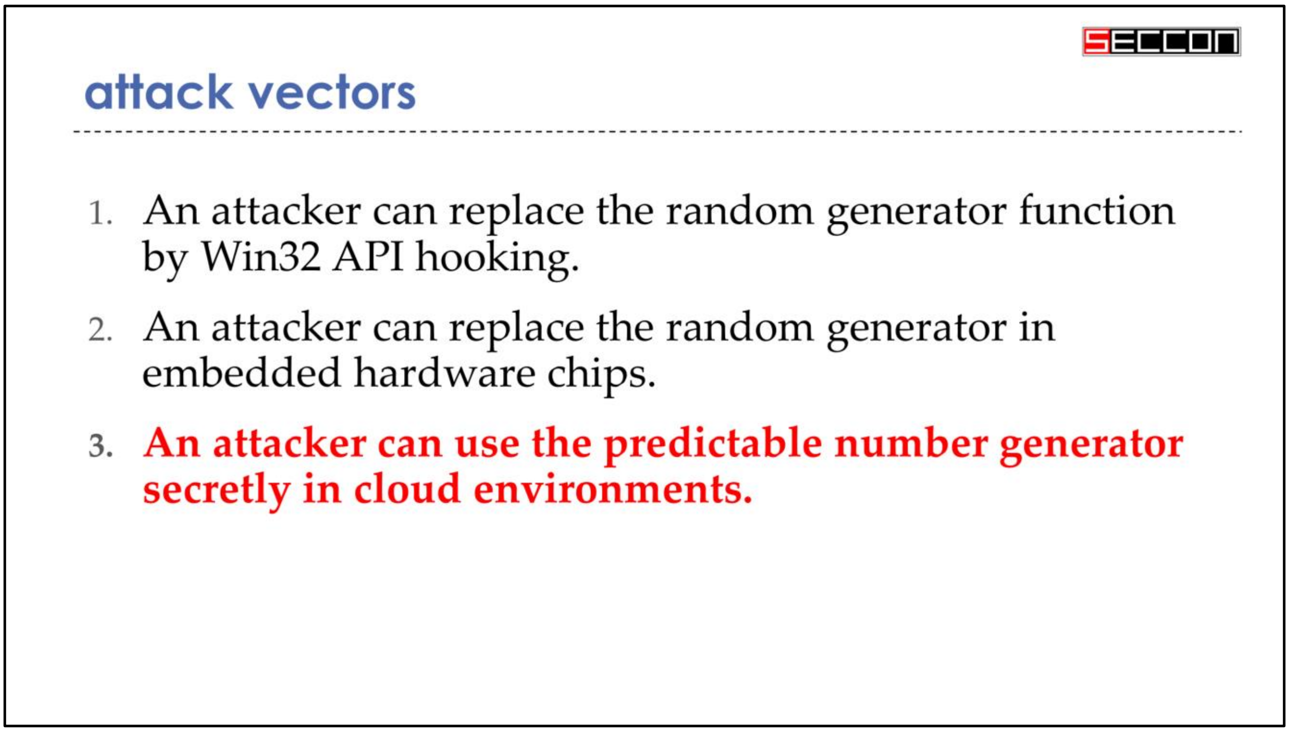


Linux's /dev/random seems like a safety

- Linus Torvalds's answer (2013.09.09):
- we use rdrand as _one_ of many inputs into the random pool, and we use it as a way to _improve_ that random pool.
- So even if rdrand were to be back-doored by some government intelligence agencies, our use of rdrand actually improves the quality of the random numbers you get from /dev/random.

[1] https://www.change.org/p/linus-torvalds-remove-rdrand-from-dev-random-4/responses/9066

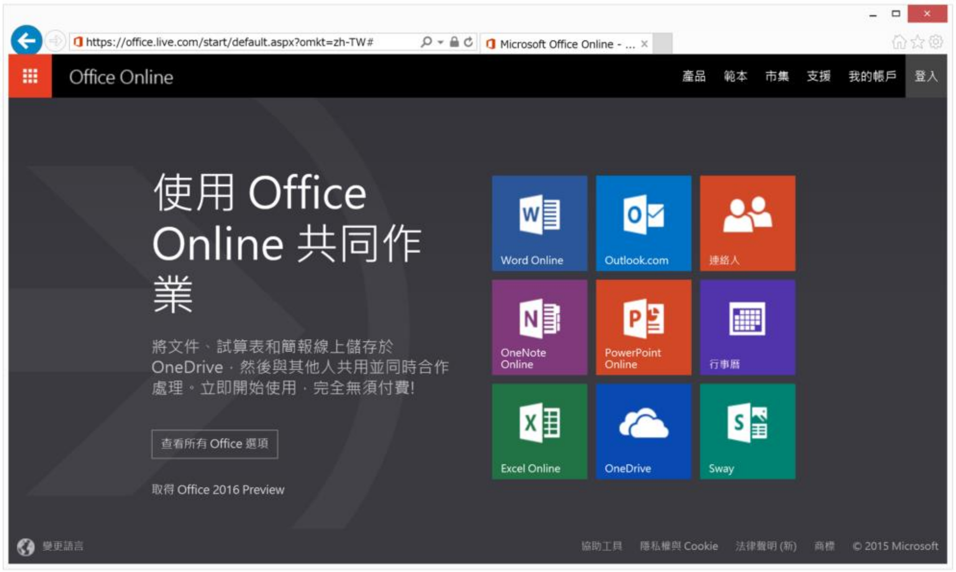
Linus Torvalds's answer is very simple.
we use rdrand as one of many inputs into the random pool, and we use it as a way to improve that random pool. So even if rdrand were to be back-doored by some government intelligence agencies, our use of rdrand actually improves the quality of the random numbers you get from /dev/random.
We can get the source code of Linux. And This is because it can be verified binaries on your own machines. It's very important that Linux is a open source software.


attack vectors

1. An attacker can replace the random generator function by Win32 API hooking.
2. An attacker can replace the random generator in embedded hardware chips.
3. **An attacker can use the predictable number generator secretly in cloud environments.**

However, what is in the cloud environments? 3rd attack vector is that some attackers can use the predictable number generator secretly in cloud environments.

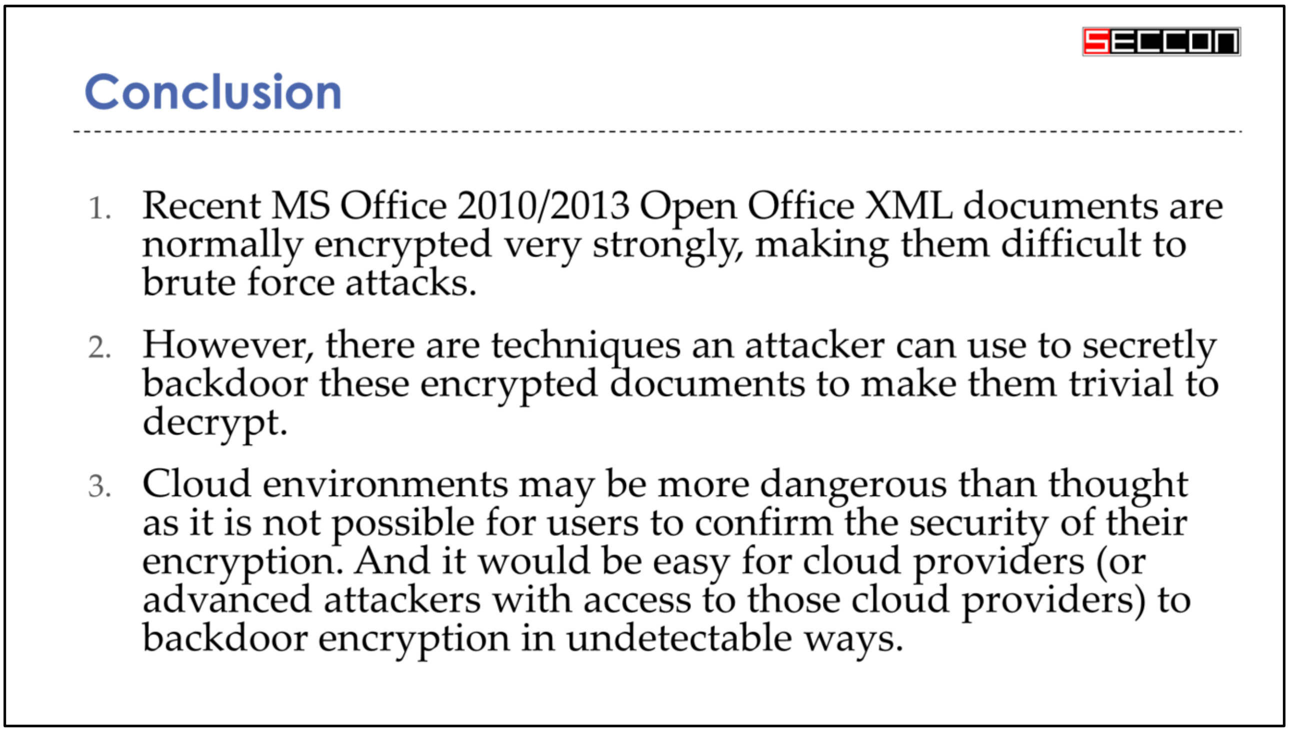
Office Online is coming soon.



使用 Office Online 共同作業

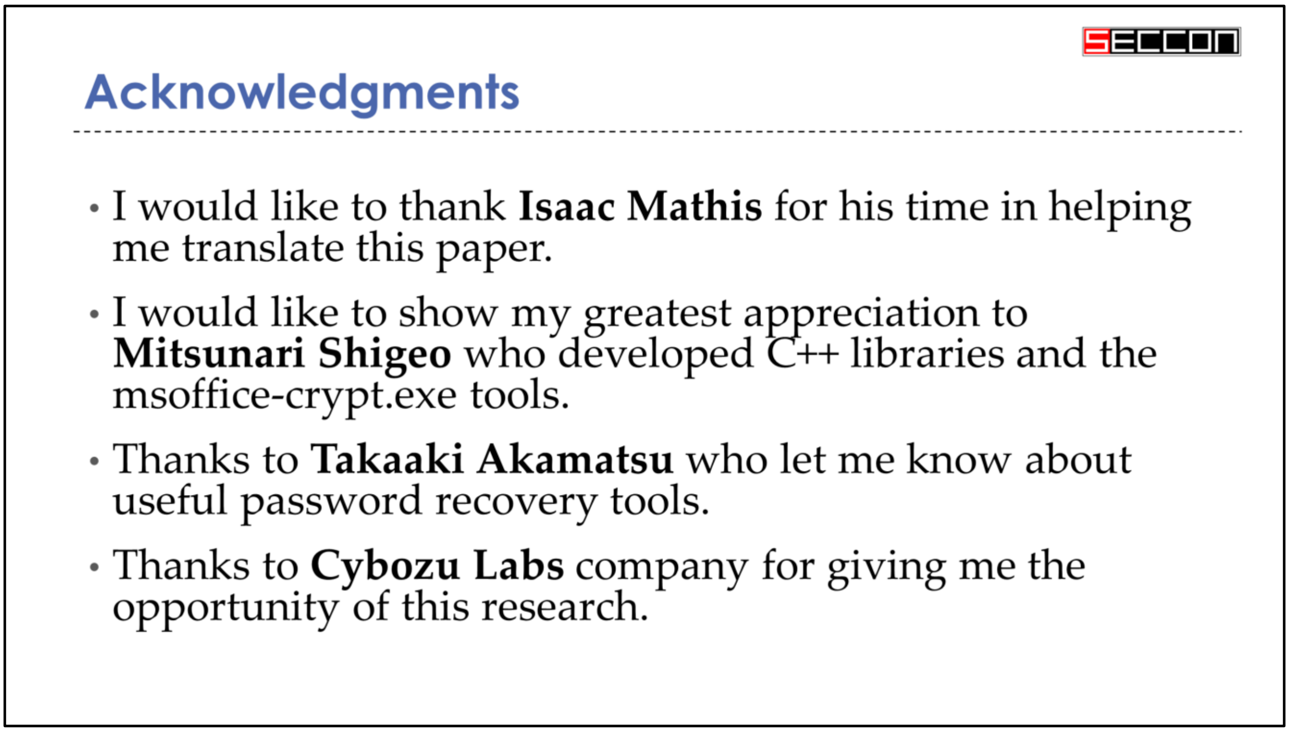
將文件、試算表和簡報線上儲存於 OneDrive，然後與其他人共用並同時合作處理。立即開始使用，完全無須付費!

Recently, Microsoft released an Office online. You can try this one as Office twenty-sixteen preview edition. The Office application will be on the Microsoft's cloud system. I think that we can not stop these cloud system movements now. We should check how the cloud encryption algorithm and encryption system is safety. Some industry companies become to have an interest in safety encryption system. I think that it is important things. Linux is a open source, but Microsoft product is closed source.


Conclusion

1. Recent MS Office 2010/2013 Open Office XML documents are normally encrypted very strongly, making them difficult to brute force attacks.
2. However, there are techniques an attacker can use to secretly backdoor these encrypted documents to make them trivial to decrypt.
3. Cloud environments may be more dangerous than thought as it is not possible for users to confirm the security of their encryption. And it would be easy for cloud providers (or advanced attackers with access to those cloud providers) to backdoor encryption in undetectable ways.

Recent MS Office twenty-ten or later version's documents are normally encrypted very strongly, making them difficult to brute force attacks. However, there are techniques some attacker can use to secretly backdoor these encrypted documents to make them trivial to decrypt. Cloud environments may be more dangerous than thought as it is not possible for users to confirm the security of their encryption. And it would be easy for cloud providers to backdoor encryption in undetectable ways.
If advanced attackers can access to those cloud providers, it will become a serious problem.



Acknowledgments

- I would like to thank **Isaac Mathis** for his time in helping me translate this paper.
- I would like to show my greatest appreciation to **Mitsunari Shigeo** who developed C++ libraries and the msoffice-crypt.exe tools.
- Thanks to **Takaaki Akamatsu** who let me know about useful password recovery tools.
- Thanks to **Cybozu Labs** company for giving me the opportunity of this research.

Thank you for your attention. And I want to say thanks for some supported members. That's all.