

# Codegate 2013 Forensic Write-ups

---

*Deok9*

*DDeok9@gmail.com*

*<http://deok9.org>*





1. Forensic 100

2. Forensic 200

3. Forensic 300

4. Forensic 400

5. Forensic 500

# Forensic 100

- It is so easy
- Cloud application(Dropbox)
- I-Phone



## 지문 확인

- 기업보안감사

A회사 보안팀은 내부직원 PC 자체보안감사 중 특정직원 PC에서 인터넷을 통해서 내부분서를 외부로 업로드한 흔적을 발견하였다. 보안팀은 보안 위반 흔적을 더 찾기 위해 직원 스마트폰도 임의 제출을 받아 추가 흔적을 조사하였다. 내부분서의 정보를 찾아 정답을 입력하시오.

correct answer ex)

```
upload date&time(UTC+9:00)_modified date&time(UTC+9:00)_filename.extention_filesize(logicalfilesize)
2013-03-01 21:00:00_2013-04-03 10:00:50_sample.docx_100MB
```

### ■ 지문에서 유추할 수 있는 내용

- 스마트폰을 추가적으로 제공 받은 것으로 보아, 스마트폰 관련된 **Artifacts** 문제임을 유추
- PC에서 내부분서를 업로드 하였다고 해당 PC 이미지를 문제 파일로 제공한 것이 아니라, 스마트폰 증거를 분석 중인 PC의 이미지를 문제 파일로 제공한 것

지문 때문에 접근에 혼동을  
가진 참여자가 존재



## 이미지 파일 확인

Name ^	Ext.	Size	Created	Modified	Accessed	Attr.	1st sector
\$Extend		448 B	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...	SH	66238
\$RECYCLE.BIN	BIN	224 B	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...	SH	66286
(Root directory)		4.1 KB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...	SH	352
17FCA978-3FF4-4FAD-A6CB-C346EAA816A2		440 B	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...		69960
3BF5888C-B2FE-4E31-9FC2-480DEA405331		4.1 KB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...		91008
3E068DCB-B90B-43DC-B1A9-D083B5BBABE7		4.1 KB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...		49640
53C38695-70F0-42A0-9C59-34606A7802A8		4.1 KB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...		175984
5BB3AF5D-01CC-45D9-947D-977DB30DD439		4.1 KB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...		118768
6BCC19E4-31A9-4381-AABA-88069F3A763F		456 B	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...		69146
6F667589-637A-45E3-92AC-E421C00FF657		4.1 KB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...		122112
7C0D5811-A2CC-4E45-A0D1-2600B299C54D		4.1 KB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...		130624
7C10A720-1D64-4B77-AB4E-136AA429EBFF		4.1 KB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...		140712
9A0ABEAF-D4E5-4BA6-8E8D-6FEB1D685B74		4.1 KB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...		146760
\$AttrDef		2.5 KB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...	SH	63136
...\$BadClus		0 B	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...	SH	
\$Bitmap		3.0 KB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...	SH	66192
\$Boot		8.0 KB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...	SH	0
\$LogFile		2.0 MB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...	SH	58000
...\$MFT		4.2 MB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...	SH	66216
\$MFTMirr		4.0 KB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...	SH	16
...\$Secure		0 B	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...	SH	
...\$UpCase		128 KB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...	SH	24
\$Volume		0 B	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...	SH	
Free space		4.1 MB					

### ■ 대략적인 이미지 구조 확인 결과

- NTFS 파일 시스템 + 10개의 폴더 존재
- 각각 폴더 내부에는 /\*\*\*.app/\*\*\*/.plist 파일이 존재

아이폰 앱??



## 접근점 파악

15BB3AF5D-01CC-45D9-947D-977DB30DD439							
Name ^	Ext.	Size	Created	Modified	Accessed	Attr.	1st sector
..							
Documents		496 B	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...		67396
Dropbox.app	app	192 KB	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...		49848
Library		456 B	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...		69068
tmp		272 B	2013-02-16 12:4...	2013-02-16 12:4...	2013-02-16 12:4...		69140
iTunesArtwork		30.2 KB	2013-02-16 12:4...	2012-12-27 17:5...	2013-02-16 12:4...	A	49648
iTunesMetadata.plist	plist	3.8 KB	2013-02-16 12:4...	2012-12-27 17:5...	2013-02-16 12:4...	A	49712

- 업로드와 관련된 앱은 Dropbox가 가장 유력
- 시나리오
  - PC에서 내부 문서를 Dropbox를 통해 업로드
  - 아이폰의 Dropbox 앱은 이를 동기화 하였고, 해당 정보는 Cache에 남음
- 접근점
  - Dropbox의 앱 Cache는 **~/Library/Caches/Cache.db**로 저장

맥의 경우 대체로  
~/Library/Caches에 Cache  
저장

## 접근 데이터 확인

SQLite Database Browser - /Users/Deok9/Desktop/cache.db

Database Structure Browse Data Execute SQL

Table: data\_cache

	key	data
1	dbmetadata	<?xml version="1.0" encoding="UTF-8"?>...
2	dbmetadata/카메라 업로드	<?xml version="1.0" encoding="UTF-8"?>...
3	dbmetadata/photos	<?xml version="1.0" encoding="UTF-8"?>...
4	dbmetadata/tim_folder	<?xml version="1.0" encoding="UTF-8"?>...
5	dbmetadata/photos/sample albi	<?xml version="1.0" encoding="UTF-8"?>...

### ■ SQLite Browser를 통해 내부 데이터 확인

- tim\_folder가 의심스러움
- tim\_folder 의 data는 base64 인코딩된 값

exported\_tim\_folder\_data.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```

<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE plist PUBLIC "-//Apple//DTD
PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd"><plist
version="1.0"><data>
YnBsaXN0MDUAAEAAgADAAQABQAI AhYCF1QkdG9w#CRvYmpIY3RzWCR2ZXJzaW9uSRhcmNoaXZl
ctEABgAHVHJyb3SAAa8QBgAJAAoAKgAuADQA0AA9AEQAXABfAGIAZQBoAGsAbgBxAHUahACHAloA
JQCQAjMAIlgCZAKgAqCuALEAtAC3ALoAvQDMAMBAGsDYANgA2wDeAOEABDzAPYA+QDBAPBBAgEF
ARQBFwEaAROBIAEjASVBKQE4ATsBPgFBAUQBFRwFKAUOBXAFfAWIBZQFoA#sBbFxAyABgWGGAYkE
JAGPAZ1BIQGkAacBqGtAbABsWg2ABkByQHMAc8B0gHVAdgB2wHeAeQBBAHzAfYB+QBAfBCAgIH
AgocDQIQAhNYJG51bGZlEBAAcWAMAAQADgAPABAAEQASABMAFAAYABYAFwAYABkAGsABABwAHQAE
ABBAIAAhACIAHwAKACUAJgAdACcAKAApU3JldlRwYXRhbnRvdGFsQnI0ZXNvY29udGVudHNFaE9G
aHwtYm5ha#xFeG1zdHNuA#NvbI8QEGxhc3Rnb2R2ZmIIZERhdGVUcm9vdF1pcORlbGV0ZWpBbY2xp
ZW50TXRpbWVYcmV2aXNpb25baXNEAkJIY3RvcnIac3RyZWftYVJsZVYkY2xhc3NUaGFzaF8QEWhtI
bWFuUmYhZGFibGVTaXpIlg2ABBAaAYIlgYyAAoBrC1AAEAoJgA+AAyBqOgAYACsALAAATVOSTLnRb
bWwAAyNBtoP1dgAAANIAAwADEAMlgY2xhc3Nlc1okY2xhc3NuYV11ogAYADNWTINEYXRIVE5T
T2JqZWNOOgAYADUANGA3WU5TLnN0cmIuZ4AFWY90aW1fZm9sZGVyOgAvADAQAQABowA6ADsAM18C
DOSTTXVOYwJsZVNOcmIuZ1h0U1N0cmIuZ18QDOSTTXVOYwJsZVNOcmIuZ91AGAA+AD8AQFpOUy5v
YmpIY3RzGisAEAAQgBDAEQARQBGAEcASABJAeAoASwBmAgAEIAYgCCAKIAwDIAQIBIgfCAwI1Bg
3xAAAAsADAANA4ADwAQABEAEGATABQAFQA#ABcAGAZABoAtgBPABQAJAAfAFIAUwBIAAAYgBX

```

## 복호화 후 확인

- 복호화를 진행하면 아래와 같은 Plist 파일이 추출됨

Key	Type	Value
▼ Item 85	Dictionary	(1 item)
NS.string	String	dropbox
▼ Item 86	Dictionary	(1 item)
NS.string	String	page_white_acrobat
▼ Item 87	Dictionary	(1 item)
NS.string	String	1c0c8960aa
▼ Item 88	Dictionary	(6 items)
totalBytes	Number	2249192
revision	Number	33
isDeleted	Boolean	NO
streamable	Number	0
thumbnailExists	Boolean	NO
isDirectory	Boolean	NO
▼ Item 89	Dictionary	(1 item)
NS.time	Number	378291354
▼ Item 90	Dictionary	(1 item)
NS.time	Number	357554798
▼ Item 91	Dictionary	(1 item)
NS.string	String	/tim_folder/S-Companysecurity.pdf
▼ Item 92	Dictionary	(1 item)
NS.string	String	2.1 MB

수정 시간이  
업로드 시간보다 이전

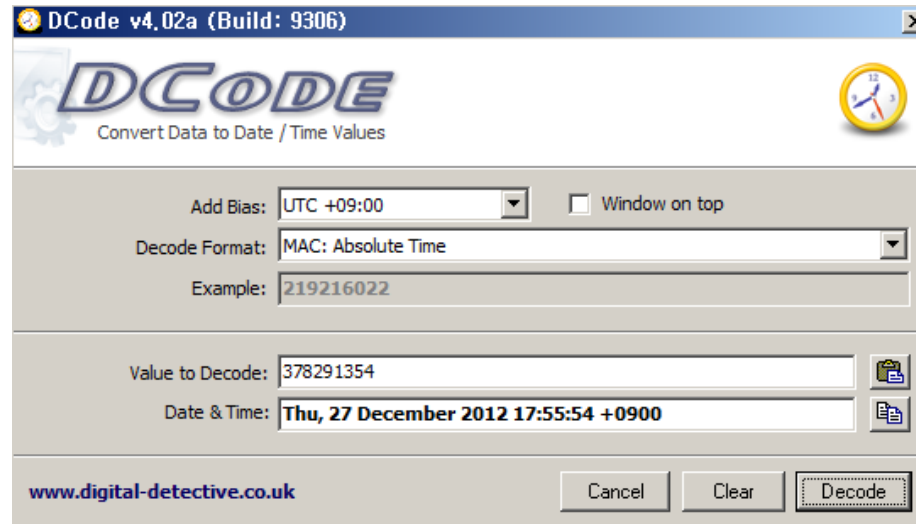
- 파일명이 S-Companysecurity.pdf 인 것으로 보아 업로드 한 내부 문서라 판단
- 첫 번째 NS.time은 업로드 시간, 두 번째 NS.time은 수정 시간





## 인증

- 각 시간 값을 MAC Absolute Time으로 변환



- Bias는 UTC +09:00 이라고 지문에 명시되어 있었음
- 인증 키 값

**2012-12-27 17:55:54 \_2012-05-01 17:46:38 \_S-Companysecurity.pdf\_2.1MB**

# Forensic 200

- It is so easy too
- Torrent Artifacts
- But, I solved this problem using \$Logfile
- Thanks to BlueAngel



## 지문 확인

경찰청은 최근 아동 성폭력과 관련된 범죄를 소탕하기 위해 대대적인 계획을 세운다.  
경찰청은 아동 성폭력 범죄들의 공통점이 아동에 대한 성적 내용이 들어간 동영상 또는 유사한 자료물에서 그 동기가 비롯된다는 것을 발견했다.  
경찰청은 인터넷에 떠돌아 다니는 아동과 관련된 음란물을 대대적으로 수사하기 시작했고, 아동 음란물을 다운로드 받는 다운로드들을 일제히 검거/구속하기 시작했다.

어느날, 다운로드의 집을 급습하여 검거를 하였는데, 나중에 다운로드의 컴퓨터와 디지털 저장매체를 분석해 보니 아동 음란물은 완전삭제되어 있었다. 그래서 아동 음란물을 받은 흔적을 찾기 위해 증거들을 모두 분석하였지만 아동 음란물 다운로드에 대한 흔적은 존재하지 않았다.

경찰청은 분명 다운로드가 아동 음란물을 받는 것을 트랙픽 모니터링을 통해 확인 하였고, 해당 트래픽 또한 증거로 가지고 있으나 결정적인 증거가 없어 해당 다운로드를 기소하지 못하고 있다.

그래서 경찰청은 그대들에게 다음과 같이 요청한다.  
"다운로더를 기소할 수 있는 결정적인 증거를 찾아주세요!"

```
Key format : SHA1("md5(Evidence File)_Download Time")  
  
Download Time : KST, YYYY/MM/DD_HH:MM:SS
```

### ■ 지문에서 유추할 수 있는 내용

- 다운로드 된 동영상 파일은 이미 삭제 되었고, 파일 다운로드 흔적을 찾으라는 문제
- 의문점 1. 다운로드 흔적을 찾은 후, 해당 파일이 아동 음란물인지 여부는 어떻게 판별?

문제 풀이 후 알았지만 토렌트  
파일은 1개 뿐



## 이미지 파일 확인

f200 f200, P1							
\Users\Administrator\AppData\Local							
Name ^	Ext	Size	Created	Modified	Accessed	Attr	1st sector
Application Data		48 B	2012-12-24 13:4...	2009-07-14 13:5...	2012-12-24 13:4...		347572
History		48 B	2012-12-24 13:4...	2009-07-14 13:5...	2012-12-24 13:4...		347574
Microsoft		4.1 KB	2012-12-24 13:4...	2012-12-24 13:4...	2012-12-24 13:4...		344
Temp		4.1 KB	2012-12-24 13:4...	2012-12-24 13:4...	2012-12-24 13:4...		347112
Temporary Internet Files		48 B	2012-12-24 13:4...	2009-07-14 13:5...	2012-12-24 13:4...		347788
uTorrent		144 B	2012-12-24 13:4...	2012-12-24 13:4...	2012-12-24 13:4...		347790
GDIPFONTCACHEV1.DAT	DAT	32.4 KB	2012-12-24 13:4...	2012-12-24 13:4...	2012-12-24 13:4...	A	343008

- 설치된 파일을 확인하기 위해 AppData를 확인하니, uTorrent 존재
  - uTorrent가 범죄자의 다운로드 프로그램임을 추측 가능
  - uTorrent를 통해 어떤 파일을 다운로드 하고 삭제 하였는지 확인하기 위해 \$Logfile 분석





## \$Logfile 확인

uTorrent.Ink	WUsers#Public#Desktop#uTorrent.Ink
Desktop.ini	WUsers#CodeGate_Forensic#AppData#Roaming#Microsoft#Windows#Start Menu#Programs#Accesso
dht.dat	WUsers#Administrator#AppData#Roaming#uTorrent#dht.dat
resume.dat	WUsers#Administrator#AppData#Roaming#uTorrent#resume.dat
rss.dat	WUsers#Administrator#AppData#Roaming#uTorrent#rss.dat
settings.dat	WUsers#Administrator#AppData#Roaming#uTorrent#settings.dat
settings.dat.old	WUsers#Administrator#AppData#Roaming#uTorrent#settings.dat.old
10E6FBE4D921B475FA5FEC6E9A535A540D6FEED1	WUsers#Administrator#AppData#Roaming#uTorrent#dlimagecache#10E6FBE4D921B475FA5FEC6E9A53
utorrent.Ing	WUsers#Administrator#AppData#Roaming#uTorrent#utorrent.Ing
3609FC884502A1DF0AA5D9D160C827BB1BD51FC	WUsers#Administrator#AppData#Roaming#uTorrent#apps#3609FC884502A1DF0AA5D9D160C827BB1B
featuredContent.btapp	WUsers#Administrator#AppData#Roaming#uTorrent#apps#featuredContent.btapp
plus.btapp	WUsers#Administrator#AppData#Roaming#uTorrent#apps#plus.btapp
player.btapp	WUsers#Administrator#AppData#Roaming#uTorrent#apps#player.btapp
2D78C93EC367E6C1D9894103FA04B3BE5B20A84E	WUsers#Administrator#AppData#Roaming#uTorrent#dlimagecache#2D78C93EC367E6C1D9894103FA0
whatsnew-ut.btapp	WUsers#Administrator#AppData#Roaming#uTorrent#apps#whatsnew-ut.btapp
83DD5C860D7C31A1D3588629CA65A66BEA75689	WUsers#Administrator#AppData#Roaming#uTorrent#dlimagecache#83DD5C860D7C31A1D3588629CA
32F529521A3DEC709F97F761F192AABF29BDC408	WUsers#Administrator#AppData#Roaming#uTorrent#dlimagecache#32F529521A3DEC709F97F761F192
BBEEC0395D21A2A7F91889D7C7509F3D5D46FC05	WUsers#Administrator#AppData#Roaming#uTorrent#dlimagecache#BBEEC0395D21A2A7F91889D7C75
98E3ED7A3B1D58C3E51BA AFC15A3D9876B4396B7	WUsers#Administrator#AppData#Roaming#uTorrent#dlimagecache#98E3ED7A3B1D58C3E51BA AFC15
C5BED7C03B5061C637102B5BB2299385699ABDDC	WUsers#Administrator#AppData#Roaming#uTorrent#dlimagecache#C5BED7C03B5061C637102B5BB22
File Creation 052b585f1808716e1d12eb55aa646fc4984bc862	WUsers#CodeGate_Forensic#AppData#Roaming#Microsoft#Windows#Start Menu#Programs#Startup
Startup	WUsers#CodeGate_Forensic#AppData#Roaming#Microsoft#Windows#Start Menu#Programs#Startup
052b585f1808716e1d12eb55aa646fc4984bc862	WUsers#CodeGate_Forensic#AppData#Roaming#Microsoft#Windows#Start Menu#Programs#Startup

### ■ uTorrent 실행 후 파일 생성 행위 발견

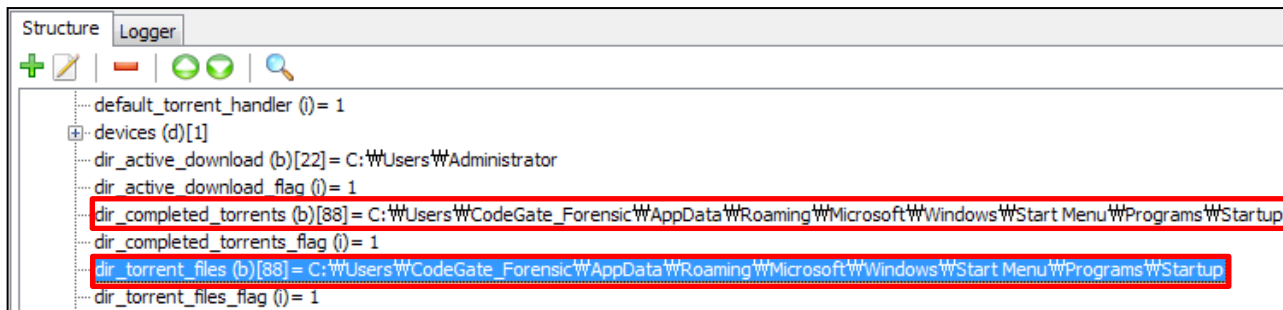
- 왜 저 위치에 파일이 생성되는 것일까?



## 경로 설정 확인

- uTorrent 폴더의 settings.dat 파일을 Bencode Editor로 확인

- dir\_completed\_torrents / dir\_torrent\_files 경로 확인



- 해당 경로는 \$Logfile에서 발견한 uTorrent 실행 후 생성된 파일의 경로와 일치

- 052b585f1808716e1d12eb55aa646fc4984bc862 파일이 증거 파일임을 짐작



## 052b585f1808716e1d12eb55aa646fc4984bc862 파일 확인

- 해당 파일의 첫 부분에 TorrentRG.com 문자열 존재

f200	f200, P1						
Users\CodeGate_Forensic\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup							
Name ^	Ext.	Size	Created	Modified	Accessed	Attr.	1st sector
052b585f1808716e1d12eb55aa646fc4984bc862		125 KB	2012-12-24 13:4...	2012-12-24 13:3...	2012-12-24 13:4...	A	133392
desktop.ini	ini	174 B	2012-12-24 13:4...	2012-12-25 04:2...	2012-12-24 13:4...	SHA	348634

[f200], ...85% free	Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
File system NTFS	04122000	3A	E2	98	85	E2	98	85	E2	98	85	20	ED	86	A0	EB	A0	:â  â  â   i  ë
Volume forensic	04122010	8C	ED	8A	B8	EC	95	8C	EC	A7	80	20	54	6F	72	72	65	ii,i  i\$  Torre
Default File Mode	04122020	6E	74	52	47	2E	63	6F	6D	20	E2	98	85	E2	98	85	E2	ntRG.com â  â  â

- 해당 파일은 uTorrent를 실행 후 settings.dat 에 설정된 위 경로에 생성된 Seed 파일

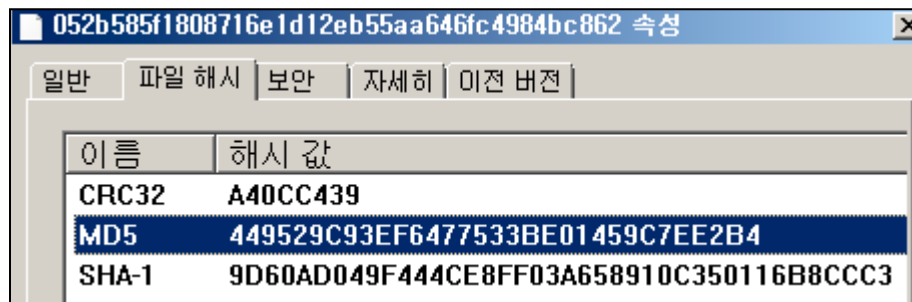
※ dir\_completed\_torrents 경로는 다운로드 완료 시 원본 Seed 파일이 생성되는 경로

이때, 생성 시각이 해당 다운로드 완료 시간



## 인증

- 해당 파일 복구 후 MD5 Hash 값 확인



- 인증 키 값

**449529C93EF6477533BE01459C7EE2B4\_2012/12/24\_13:45:43**



# Forensic 300

- It is so surprise to me
- ooXML Steganography



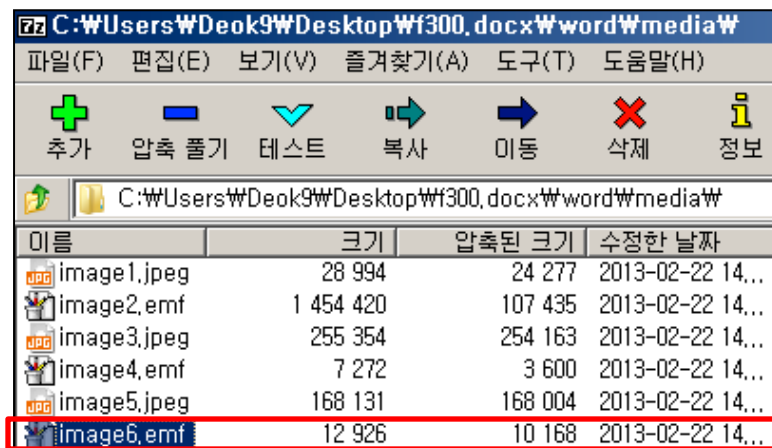
## 지문 확인

- Find Key . . .
- 문제 파일은 docx 파일 1개
  - docx 파일이 문제로 주어질 경우 대부분 ooXML 포맷의 특징을 이용한 Data Hidden 문제
- Documen.xml.rels 파일을 통해 파일 관계 확인

```
<?xml version="1.0" encoding="UTF-8" standalone="true"?>
<Relationships xmlns="http://schemas.openxmlformats.org/officeDocument/2006/relationships">
  <Relationship Target="media/image4.emf" Type="http://schemas.microsoft.com/office/2006/relationships/image" />
  <Relationship Target="webSettings.xml" Type="http://schemas.microsoft.com/office/2006/relationships/webSettings" />
  <Relationship Target="media/image3.jpeg" Type="http://schemas.microsoft.com/office/2006/relationships/image" />
  <Relationship Target="theme/theme1.xml" Type="http://schemas.microsoft.com/office/2006/relationships/theme" />
  <Relationship Target="settings.xml" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/settings" />
  <Relationship Target="styles.xml" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/styles" />
  <Relationship Target="embeddings/oleObject1.bin" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/embeddings" />
  <Relationship Target="fontTable.xml" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/fontTable" />
  <Relationship Target="media/image2.emf" Type="http://schemas.microsoft.com/office/2006/relationships/image" />
  <Relationship Target="media/image5.jpeg" Type="http://schemas.microsoft.com/office/2006/relationships/image" />
  <Relationship Target="media/image1.jpeg" Type="http://schemas.microsoft.com/office/2006/relationships/image" />
  <Relationship Target="embeddings/oleObject2.bin" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/embeddings" />
</Relationships>
```

## 수상한 파일 발견

- 실제 media 안에는 image6.emf 파일이 숨겨져 있음



- 해당 파일 확인 결과 doc 파일이며, 문서 내용은 "2013" 이라는 붉은 문자열

2013



## 힌트가 주어짐

### ▪ “compare the docx”

- Compare 기법을 이용한 것들은 주로 Steganography와 연관
  - ✓ 데이터가 숨겨진 파일의 특징과 숨겨지지 않은 파일의 특징을 비교

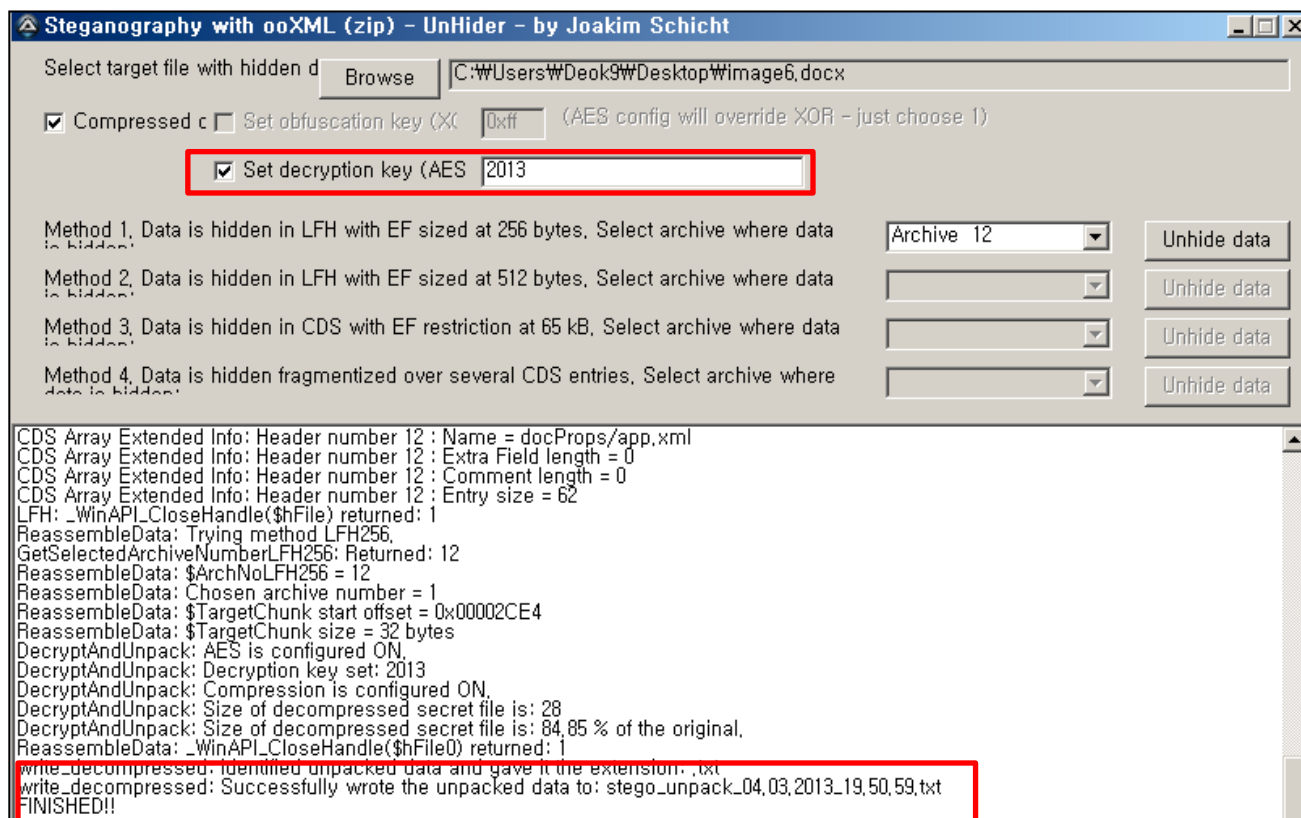
### ▪ Google에 ooXML Steganography 검색

- “Steganography with ooXML ”이라는 유명한 Tool 존재
- 고려대에서 ooXML Steganography 논문을 번역한 PPT 문서도 발견
  - ✓ **New Steganographic Techniques for the OOXML File Format** – 18기 윤지혜
  - ✓ 해당 PPT 및 논문은 OOXML에 관한 다양한 Steganography 기법들을 언급



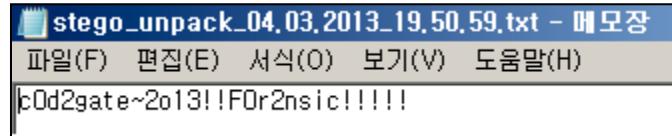
## Tool 사용

- “2013”은 Tool에서 지원하는 AES Decryption Key 임을 Guessing



## 인증

- Decompressed 된 “stego\_unpack\_04.03.2013\_19.50.59.txt” 확인



- 인증 키 값

**c0d2gate~2o13!!F0r2nsic!!!!**

- 2번째 힌트 “Extra field entry”가 인증 후 나옴

- 이를 통해 첫번째 hint가 docx 파일 2개의 **Extra field**를 비교하란 것을 확인
- Image 6의 Extra field를 원본 문제 파일(f300.docx)와 비교하니 아래와 같은 AES 값 발견

5E 23 32 A0 09 8B 7F 00 00 00 FF FF 03 00 50 4B	^#2 .<....ÿÿ..PK
03 04 14 00 06 00 08 00 00 00 21 00 98 60 B0 89	.....!.~`%`
7B 01 00 00 CE 02 00 00 10 00 08 01 64 6F 63 50	{...Ï.....docP
72 6F 70 73 2F 61 70 70 2E 78 6D 6C 20 A2 04 01	rops/app.xml <..
01 00 20 00 59 ED 33 14 66 32 D8 7B A3 10 02 D3	..ÿi3.f20{f...0
F2 8D 49 69 16 B4 9E F4 37 BC 63 C7 C9 78 25 C1	ô.Ii.'žô74cÇÊx%Ä
E5 8C EC 0C 00 00 00 00 00 00 00 00 00 00 00	â@i.....

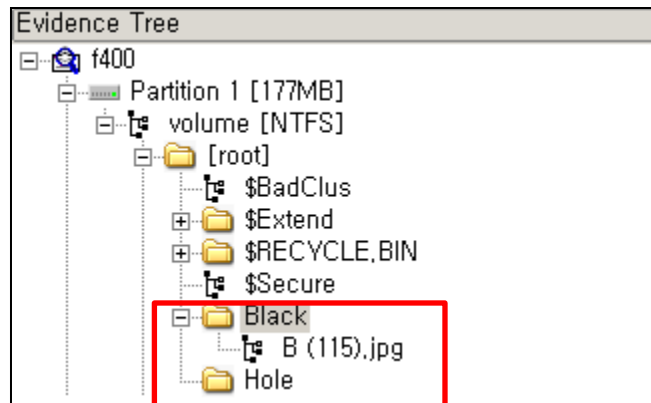
# Forensic 400

- So So
- ADS, Fileslack
- Fuzzyhash
- Truecrypt



## 지문 확인

- Find Key . . .
- 문제 파일은 Disk Image 파일 이며 FTK Imager를 통해 확인



- Black & Hole 폴더가 존재하며, B(115).jpg 에 ADS로 데이터가 숨겨져 있음을 확인





## ADS 파일 확인

- ADS에 특정 파일을 숨겨 놓았다는 것은 문제 풀이에 결정적인 파일일 가능성 높음



- Byte 분포도를 살펴본 결과 차이 없이 거의 균등한 분포를 띄고 있음
  - 해킹대회에서 이럴 경우 대부분 Truecrypt 이미지일 가능성 **90%(자주 출제됨)**
- **Truecrypt Image를 마운트 하기 위한 Key 값은?**
    - Black 과 Hole을 적절히 이용해 보자

## Black 폴더 확인

### 시간 순으로 정렬

B (18).jpg	776 KB	Regular File	2013-02-21 오전 5:36:29
B (46).jpg	776 KB	Regular File	2013-02-21 오전 9:47:46
B (59).jpg	772 KB	Regular File	2013-02-21 오전 9:47:46
B (68).jpg	772 KB	Regular File	2013-02-21 오전 9:47:46
B (113).jpg	772 KB	Regular File	2013-02-21 오전 9:47:47
B (137).jpg	692 KB	Regular File	2013-02-21 오전 9:47:47
B (20).jpg	776 KB	Regular File	2013-02-21 오전 9:47:48
B (28).jpg	780 KB	Regular File	2013-02-21 오전 9:47:48
B (42).jpg	776 KB	Regular File	2013-02-21 오전 9:47:48
B (71).jpg	776 KB	Regular File	2013-02-21 오전 9:47:48
B (103).jpg	776 KB	Regular File	2013-02-21 오전 9:47:48
B (108).jpg	772 KB	Regular File	2013-02-21 오전 9:47:48
B (118).jpg	772 KB	Regular File	2013-02-21 오전 9:47:48
B (19).jpg	776 KB	Regular File	2013-02-21 오전 9:47:48
B (55).jpg	776 KB	Regular File	2013-02-21 오전 9:47:48
B (58).jpg	776 KB	Regular File	2013-02-21 오전 9:47:48
B (78).jpg	776 KB	Regular File	2013-02-21 오전 9:47:48

00000	FF D8 FF E1 21 AD 45 78-69 66 00 00 49 49 2A 00	27
00010	08 00 00 00 00 09 00 00 01-04 00 01 00 00 00 C0 0C	
00020	32 37 01 01 04 00 01 00-00 00 90 09 00 00 0F 01	

특정 순서를 나타낸  
파일로 짐작됨

- 오전 5시대 영역과 9시대 영역으로 확연히 구분 가능하며, 9시대 영역의 파일은 총 27개
- 9시대 영역 JPG 파일들에는 Offset 0x20 ~ 0x21 지점에 순서를 나타내는 숫자가 표기

## Hole 폴더 확인

- 시간 순으로 정렬

Name	Size	Type	Date Modified
x6bym670hWSO2718,...	772 KB	Regular File	2013-02-21 오전 5:36:05
CXOal816j3nd8604.jpg	776 KB	Regular File	2013-02-21 오전 9:46:18
SeOPj839ibKR3093.jpg	772 KB	Regular File	2013-02-21 오전 9:46:19
cxU607gLOjCh5020.jpg	776 KB	Regular File	2013-02-21 오전 9:46:21
36CDskL88fYm3811,j...	776 KB	Regular File	2013-02-21 오전 9:46:21
ER00DmSSz5068005	776 KB	Regular File	2013-02-21 오전 9:46:21

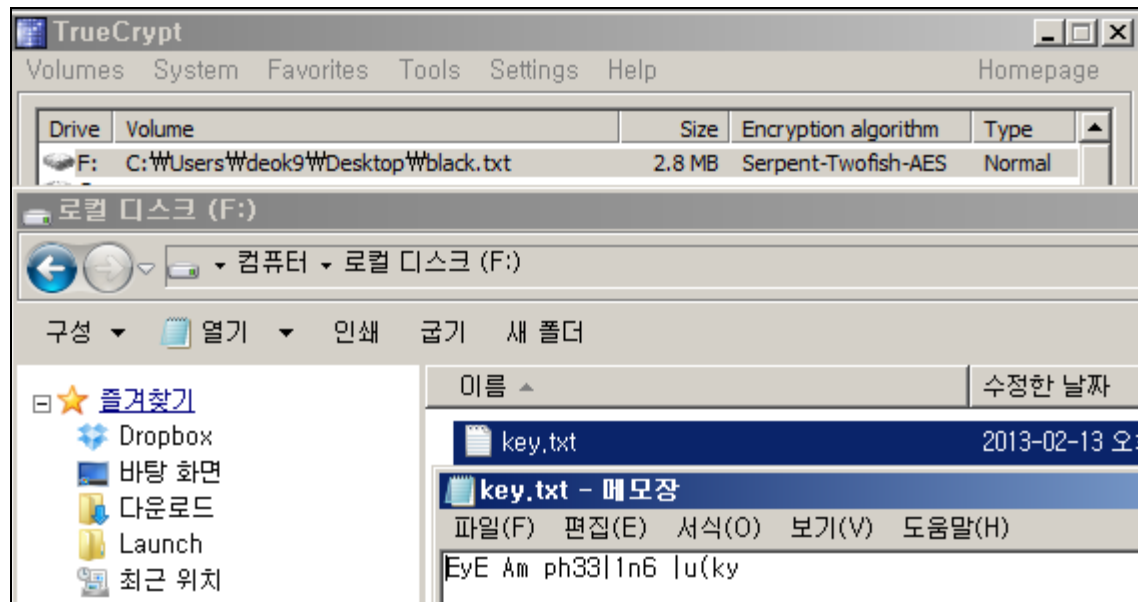
c1b50	D0 9A BB 77 37 96 82	...
c1b60	7C DF E1 14 87 CB 92 52-CE	...
c1b70	1C AD F2 1E 82 A0 B0	...
c1b80	FE 1C FA 51 22 ED 6C 0A-00	...
c1b90	0E A3 E5 27 02 80 B1 17	...
c1ba0	E4 D0 49 38 45 55 CB 72-6A 17 C6	...
c1bb0	6A 10 B8 0D D3 BD 5B B8-16 F1 9D	...
c1bc0	B2 9C 93 17 62 4F 6A 89-46 F6 24 76 E7 AD 34 EC	...
c1bd0	4B 2C 5B DD 88 98 19 17-78 1D 89 AB 06 E4 DC B8	...
c1be0	08 98 51 E9 DA 93 62 4A-E7 FF D9 00 00 00 00	77 ...QéŮ·bJçŸŮ... w

특정 문자를 나타낸  
파일로 짐작됨

- 9시대 영역의 파일은 총 27개로 동일
- 9시대 영역 JPG 파일들에는 모두 Footer 뒤의 Slack에 1Byte 문자가 존재
- 1Byte 값 27개 = 7y\_5n4\_10221CuPw7R73\_rrK9Pd

## 확인

- Fuzzyhash 값 비교로 모든 글자의 순서를 맞춤
  - "7ru3CyP7\_P422w0Rd\_57r1n9\_K1" 라는 문자열이 나오게 됨
  - l33t 언어이며, Truecrypt\_Password\_String\_Key 로 번역 가능
- Truecrypt에서 ADS에서 추출된 Truecrypt 이미지를 해당 키 값으로 마운트





## 인증

- 인증키 값은 마운트된 Truecrypt 이미지에 존재하는 key.txt 파일 내용

**EyE Am ph33l1n6 lu(ky**

- 의견

- 지문 없이 단순 이미지만 가지고 문제 풀기에는 어려움이 있었음
- 기술적인 점만 너무 고려하지 않았나 생각함
  - ✓ 그러나 기술적으로 특이한 것도 없었음
- Black 폴더의 그림에서 순서라는 것을 정황상 찾게 하도록 의도 했다면 더 좋았을 듯함
- 대부분의 팀들이 Fuzzyhash 가 아닌 문자열 계싱으로 Truecrypt 키 값을 추출

# Forensic 500

- What The F...
- MFT Data runs
- Fragment file carving











## 지문 확인

### Explain

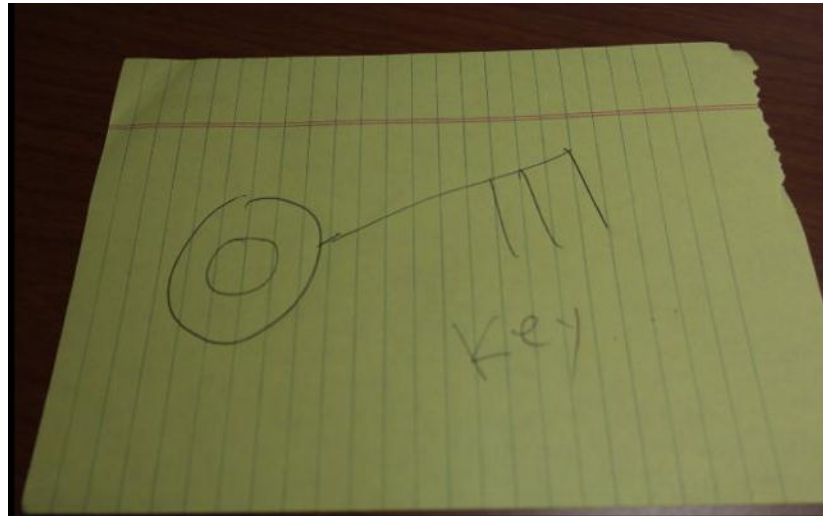
Recovered key image file Logical  
file hash value ?(MD5)

- 무슨 말일까 ...
  - 복구된 키 이미지 파일 논리적 파일 Hash 값?
- Key image file을 먼저 찾아보자

f500	Unpartitioned space						
Partitioning style: MBR							
Name ^	Ext.	Size	Created	Modified	Accessed	Attr.	1st sector
 Partition 1	NT...	30.0 MB					128
 Partition 2	FAT...	10.0 MB					61568
 Partition 3	NT...	15.0 MB					82048
 Partition 4	FAT...	21.0 MB					215424
 Start sectors		64.0 KB					0
 Unpartitionable space		64.0 KB					266112
 Unpartitioned space		50.1 MB					112768
 Unpartitioned space		3.8 MB					258432

## Data Carving 수행

- Key 파일 발견



- R-Studio 로 복구한 파일의 경우 추출된 파일의 Hash값으로 인증 가능
- 500점 문제 인데 몇 분만에 20개 이상의 팀이 인증함
  - ✓ 참가자들이 문제 푸는 방식을 고려하지 않아 난이도 조절에 실패





## Logical File Hash?

### ▪ MFT 에서 k3y2013.jpg 파일 확인

46 49 4C 45 30 00 03 00	97 74 28 00 00 00 00 00	FILE0	It(
02 00 01 00 38 00 01 00	60 01 00 00 00 04 00 00	8	,
00 00 00 00 00 00 00 00	03 00 00 00 32 00 00 00	2	
02 00 00 00 00 00 00 00	10 00 00 00 60 00 00 00		
00 00 00 00 00 00 00 00	48 00 00 00 18 00 00 00	H	
20 F3 4C 14 1A 08 CE 01	00 99 10 DF 17 08 CE 01	óL î ß î	
CF EF 5A 87 18 08 CE 01	20 F3 4C 14 1A 08 CE 01	ïZ î óL î	
20 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00		
00 00 00 00 05 01 00 00	00 00 00 00 00 00 00 00		
00 00 00 00 00 00 00 00	30 00 00 00 70 00 00 00	0 p	
00 00 00 00 00 00 02 00	58 00 00 00 18 00 01 00	X	
05 00 00 00 00 00 05 00	20 F3 4C 14 1A 08 CE 01	óL î	
20 F3 4C 14 1A 08 CE 01	20 F3 4C 14 1A 08 CE 01	óL î óL î	
20 F3 4C 14 1A 08 CE 01	00 40 49 00 00 00 00 00	óL î @I	
00 00 00 00 00 00 00 00	20 00 00 00 00 00 00 00		
0B 00 6B 00 33 00 79 00	32 00 30 00 31 00 33 00	k 3 y 2 0 1 3	
2E 00 4A 00 50 00 47 00	80 00 00 00 50 00 00 00	. J P G P	
01 00 00 00 00 00 01 00	00 00 00 00 00 00 00 00		
93 04 00 00 00 00 00 00	40 00 00 00 00 00 00 00	@	
00 40 49 00 00 00 00 00	0E 3D 49 00 00 00 00 00	@I =I	
0E 3D 49 00 00 00 00 00	22 3D 02 B7 06 22 DF 00	=I "= . "ß	
F3 28 22 78 01 56 E4 00	FF FF FF FF 82 79 47 11	ó("x Vã yyylyG	

- Non-resident 속성과 이에 따른 Cluster Runs 존재
- Logical File 이란, Cluster Runs를 따라가서 복구한 파일을 의미?



## Cluster Runs 확인

### ▪ K3y2013.jpg Cluster Runs

값	구조	설명
22 3D 02 B7 06	2Byte Length, 2Byte Offset	Length 1 : 0x023D * Cluster Size Offset 1 : 0x06B7 * Cluster Size + Start Offset
22 DF 00 F3 28	2Byte Length, 2Byte Offset	Length 2 : 0x00DF * Cluster Size Offset 2 : 0x28F3 * Cluster Size + Offset 1
22 78 01 56 E4	2Byte Length, 2Byte Offset	Length 3 : 0x0178 * Cluster Size Offset 3 : 0xE456 * Cluster Size + Offset 2

- Offset 3의 0xE456은 음수를 나타냄(-0x1BAA)

### ▪ Cluster Size/Start Offset 확인

00010000	EB 52 90 4E 54 46 53 20 20 20 20 00 02 08 00 00	ëR NTFS
00010010	00 00 00 00 00 F8 00 00 3F 00 FF 00 80 00 00 00	ø ? ý I

- Cluster Size :  $0x0200 * 0x0008 = 0x1000$
- Start Offset : 0x10000



## 복구 시작

- **Offset 1 :  $0x06B7 * \text{Cluster Size}(0x1000) + \text{Start Offset}(0x10000) = 0x6C7000$**
- **Length 1 :  $0x023D * \text{Cluster Size}(0x1000) = 0x23D000$** 
  - 0x6C7000 ~ 0x904000 지점 까지 1번째 Fragmented Data

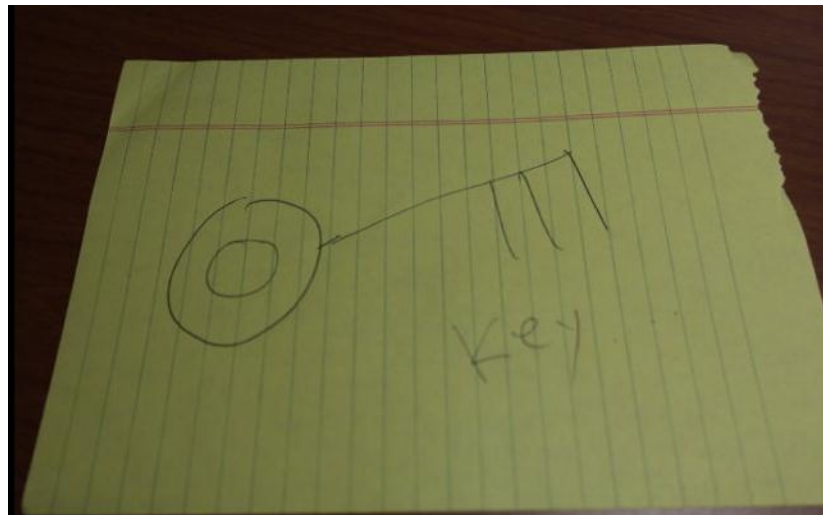
006C7000	FF D8 FF E1 DB 0D 45 78 69 66 00 00 4D 4D 00 2A	ÿøÿáÛ Exif MM *
006C7010	00 00 00 08 00 0E 01 0E 00 02 00 00 00 0C 00 00	



00903FF0	B6 D4 2E 61 0D 19 B9 19 5F 35 FC AD E1 B0 AD 92	00000000_5ü-á°-
00904000	FF D8 FF E1 36 98 45 78 69 66 00 00 49 49 2A 00	v0vâ6 Exif II*

- 단순 반복 하여 파일 생성

## 복구된 파일 확인



- 해당 파일의 Hash 값을 인증하면 인증이 되지 않음
  - Cluster 단위로 단편화된 데이터를 합쳤기 때문에 쓸데 없는 **0x00** 값이 들어갔기 때문



## 수정 후 인증

- JPG 파일의 Footer인 FF D9 까지 잘라냄

00493CF0	95 78 E6 A3 64 03 A5 2B 8C 85 F2 07 BD 07 3E 5D	!xæfd ₩+!!ò ½ >]
00493D00	00 47 26 D2 47 B5 40 EE 01 C0 A0 67 FF D9	G&ÒGμ@i À gyŮ

- 해당 파일 MD5 Hash 값 확인

이름	해시 값
CRC32	A563A938
MD5	597EB84759C836CF9889E07770FFACF7
SHA-1	F27EA164DBC3D557F62EB4F9D75D7679E110E54B

- 인증 키 값

**597EB84759C836CF9889E07770FFACF7**

- 의문점

- 원래 이렇게 복구된 파일을 Logical File이라 부르는지?
- 이렇게 구조를 따라가서 복구해 주는 Tool에는 어떤 것이 있는지?

