```
by@ubuntu:~/dump$ sigfind -b 4096 504b  03_dump_txt.img
Block size: 4096  Offset: 0  Signature: 504B
Block: 33192 (-)
Block: 35328 (+2136)
error reading bytes 507619
```

⇨ 일단 .zip으로 변경후 xml 분석.

```
ion/vnd.openxmlformats-
PartName="/xl/worksheets/
```

라고 나온다.

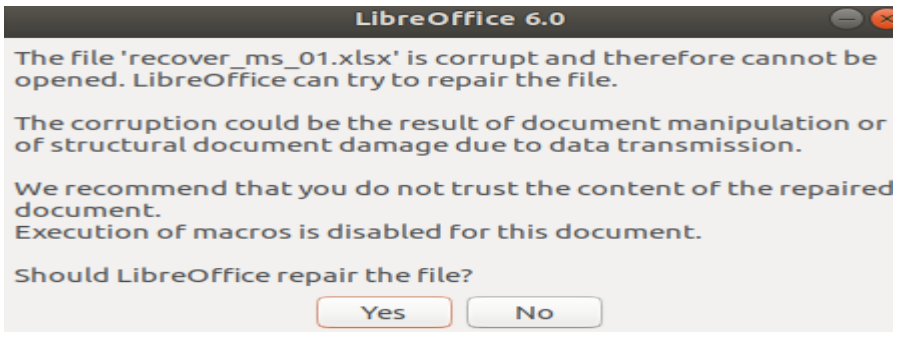
50 4B 03 04 14 00 06 00	PK.....
DOCX, PPTX, XLSX	Microsoft Office Open XML Format (OOXML) Document **NOTE:** There is no subheader for MS OOXML files as there is with DOC, PPT, and XLS files. To better understand the format of these files, rename any OOXML file to have a .ZIP extension and then unZIP the file; look at the resultant file named *[Content_Types].xml* to see the content types. In particular, look for the *<Override PartName=* tag, where you will find *word*, *ppt*, or *xl*, respectively. **Trailer:** Look for 50 4B 05 06 (PK..) followed by 18 additional bytes at the end of the file.

⇨ 위의 설명을 보면 xl, ppt, word 라고 써있어서 파일을 구별가능하다.

⇨ 결론은 sigfind 로 504b 를 찾아 dd 로 zip 파일을 만든 후 zip 파일 내부에 있는 xml 파일에서 이름 태그를 찾아서 파일을 구분하면 된다.

⇨ 문제 : 이 안에 들어가는 쓰래기 값.

```
by@ubuntu:~/dump$ cat recover_ms_01 >recover_ms_01.xlsx
```



⇨ 불필요한 값이 들어가서 복구 할 수 없다. => 끝을 알아서 잘라야 함.

< 다음 방법을 써볼까 함>

⇨ **파일 구조체 카빙**

⇨ 파일의 미리보기를 지원하는 파일의 경우 파일 포맷 내부에 실제 내용 이외에 썸네일(Thumbnail)을 포함한다. 이 경우 썸네일도 해당 파일의 포맷과 동일한 구조로 되어 있기 때문에 전체 파일 포맷에 시그니처가 둘 이상이 존재하게 된다. 파일 구조체 카빙 기법은 푸터 시그니처가 존재하지 않거나 파일 포맷 내부에 여러 개의 시그니처가 존재하는 경우에 효과적인 방법으로 파일의 구조를 분석하여 카빙하는 기법이다. 파일 구조체 카빙 기법은 다시 파일 크기 획득 방법과 파일 구조 검증 방법으로 나눌 수 있다.

⇨ **파일 크기 획득 방법**

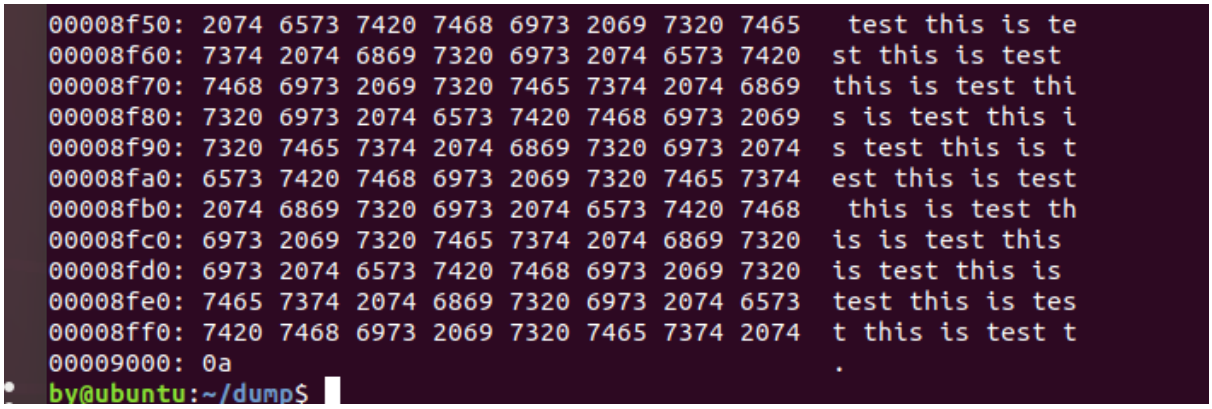
⇨ 대부분의 파일은 해당 데이터 표현을 위해 파일의 앞부분에 파일구조체가 위치한다. 파일구조체 내부에는 파일시스템의 파일메타정보와 유사하게 해당 파일의 크기, 형식 등의 정보가 포함된 메타정보가 저장된다. 파일 크기 획득 방법은 파일구조체 내부에서 파일 크기 정보를 획득하여 카빙하는 방법이다. 파일의 시작위치와 파일 크기에 대한 정보를 알고 있다면 비교적 쉽게 해당 파일을 카빙할 수 있다. 아래 표는 파일 크기 획득 방법을 적용할 수 있는 파일 포맷과 시그니처의 예이다.

파일 크기 획득 방법을 적용할 수 있는 파일 포맷

파일 포멧	시그니처	
	헤더(Hex)	푸터(Hex)
BMP	42 4D ("BM")	-
EXE	4D 5A ("PE")	-
DLL		-

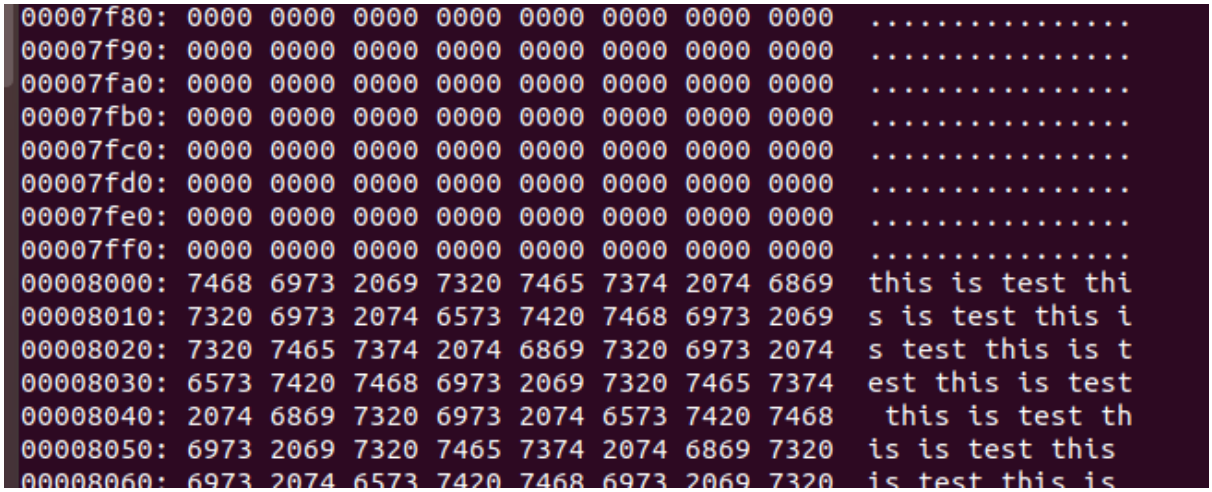
⇨ 파일 구조체를 찾아서 크기를 알아내 footer 가 없는 파일을 카빙해본다. 구조체가 없을 가능성이 큼.

⇨ Xxd 로 recover_ms_01 을 열어봄.



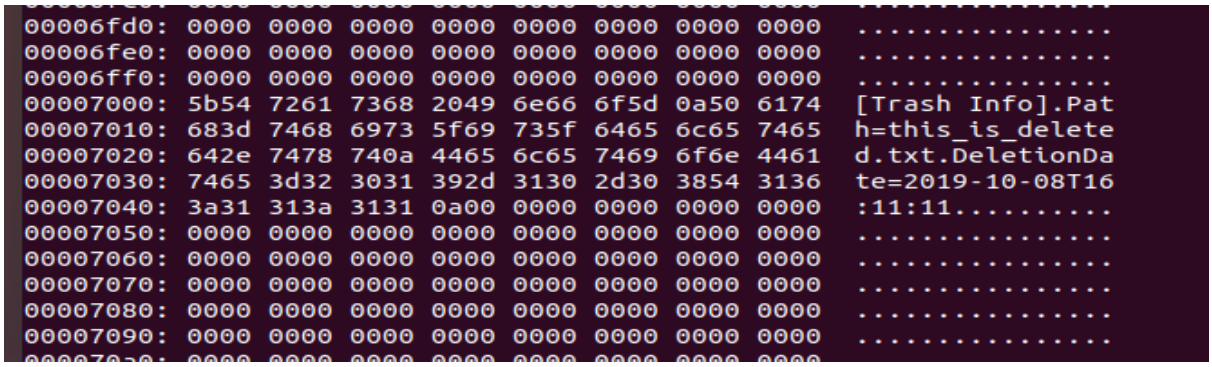
⇨ 끝부분에 header 도 footer 도 없는 txt 파일이 섞여 들어간걸 볼 수 있음.

⇨ Txt 의 시작 부분은 그냥 시작됨. Txt 앞은 0으로 패딩 됨.



⇨ 파일을 보다가 다음과 txt 파일이 같이 삭제되었다는 기록을 발견함.

⇨ [Trash Info] = 5b54 7261 7368 2049 6e66 6f5d 0a50 를 인식하는 것은 어떤가 제안.

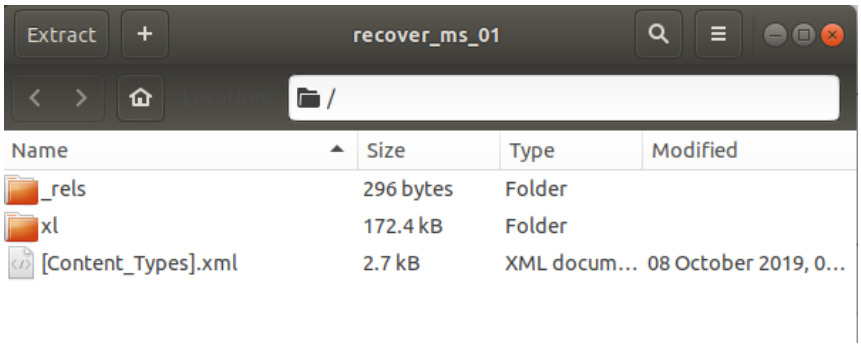


⇨ Xlsx 파일의 끝을 발견함.

```
00006c90: 6573 2e78 6d6c 504b 0102 1400 1400 0808  es.xmlPK........
00006ca0: 0800 d804 484f 2cc2 1697 d101 0000 eb04  ....HO,.........
00006cb0: 0000 0f00 0000 0000 0000 0000 0000 0000  ................
00006cc0: 3d5f 0000 786c 2f77 6f72 6b62 6f6f 6b2e  =_..xl/workbook.
00006cd0: 786d 6c50 4b01 0214 0014 0008 0808 00d8  xmlPK...........
00006ce0: 0448 4fc4 283b dd07 0100 0011 0600 001a  .HO.(;..........
00006cf0: 0000 0000 0000 0000 0000 0000 004b 6100  .............Ka.
00006d00: 0078 6c2f 5f72 656c 732f 776f 726b 626f  .xl/_rels/workbo
00006d10: 6f6b 2e78 6d6c 2e72 656c 7350 4b01 0214  ok.xml.relsPK...
00006d20: 0014 0008 0808 00d8 0448 4fa4 6fa1 20b2  .........HO.o. .
00006d30: 0000 0028 0100 000b 0000 0000 0000 0000  ...(............
00006d40: 0000 0000 009a 6200 005f 7265 6c73 2f2e  ......b._rels/.
00006d50: 7265 6c73 504b 0102 1400 1400 0808 0800  relsPK..........
00006d60: d804 484f bea8 0400 6b01 0000 a80a 0000  ..HO....k.......
00006d70: 1300 0000 0000 0000 0000 0000 0000 8563  ...............c
00006d80: 0000 5b43 6f6e 7465 6e74 5f54 7970 6573  ..[Content_Types
00006d90: 5d2e 786d 6c50 4b05 0600 0000 001e 001e  ].xmlPK.........
00006da0: 0064 0800 0031 6500 0000 0000 0000 0000  .d...1e.........
00006db0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00006dc0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
```

⇨ 여기서 끝이 [Content_Type].xmlPK 부분이랑 근접하다는 것을 알 수 있음.

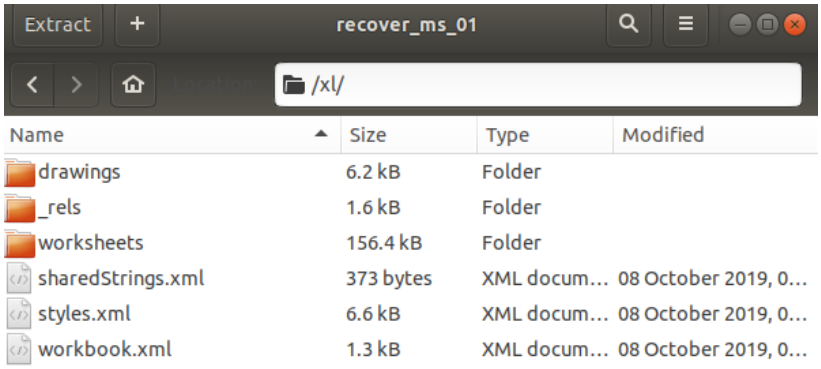
⇨ Recover_ms_01. 의 구조는 다음과 같음



⇨ 이번에는 Recover_ms_01의 시작 부분으로 이동함.

```
by@ubuntu:~/dump$ xxd  recover_ms_01
00000000: 504b 0304 1400 0808 0800 d804 484f 0000  PK..........HO..
00000010: 0000 0000 0000 0000 1800 0000 786c       ............xl
00000020: 2f64 7261 7769 6e67 732f 6472 6177 696e  /drawings/drawin
00000030: 6731 2e78 6d6c 9dd0 5d6e c230 0c07 f013  g1.xml..]n.0....
00000040: ec0e 55de 695a 1813 4314 5ed0 4e30 0ee0  ..U.iZ..C.^.N0..
00000050: 256e 1b91 8fca 0ea3 dc7e d14a 3669 7b01  %n.......~.J6i{.
00000060: 1e6d cb3f f9ef cd6e 74b6 f844 6213 7c23  .m.?...nt..Db.|#
00000070: eab2 1205 7a15 b4f1 5d23 0eef 6fb3 9528  ....z...]#..o..(
00000080: 3882 d760 83c7 465c 90c5 6efb b419 35ad  8..`..F\..n...5.
00000090: cfbc a722 ed7b 5ea7 b211 7d8c c35a 4a56  ..."{^...}..ZJV
```

⇨ 크기에 대한 정보는 딱히 없는듯...



⇨ 이번에는 recover_ms_02 를 열어보았음.

```
by@ubuntu:~/dump$ xxd  recover_ms_02
00000000: 504b 0304 1400 0808 0800 f604 484f 0000  PK..........HO..
00000010: 0000 0000 0000 0000 0000 1000 0000 776f  ..............wo
00000020: 7264 2f66 6f6f 7465 7231 2e78 6d6c a595  rd/footer1.xml..
00000030: db6e 9c30 1086 9fa0 ef80 7cbf 0b44 490f  .n.0......|..DI.
00000040: 68d9 5c74 d5a8 522f 5669 fb00 1363 c08a  h.\t..R/Vi...c..
```

⇨ 다음과 같은 내용에서 이 파일은 word 파일이라는 것을 알 수 있음. (파일이 손상이 심해서 zip 을 열지 못함.

```
00008fc0: 7985 38d9 6294 05f5 0567 b223 49ac 9b15  y.8.b....g.#I...
00008fd0: a6b6 4937 0e25 68a9 bfe4 a5a9 d0ab cf06  ..I7.%h.........
00008fe0: 09a3 9b6f 1d4c 94cd 52f0 7830 9266 e37d  ...o.L..R.x0.f.}
00008ff0: c56b c42a 9abe eb7c 4bdd 84a8 6f92 e4e7  .k.*...|K...o...
by@ubuntu:~/dump$
```

⇨ 끝까지 내용이 차있음. 이 파일 같은 경우는 쓰래기 값이 들어간 것이 아니라 파일을 전부 가져오지 못한것 아닐까 추측.

⇨ 끝 위치를 찾아야 될 필요가 있음.

⇨ 결론: sigfind 로 대충 위치를 찾고

- Header 와 footer 가 있는 경우 (jpeg) : 원래 방식으로
- Header 만 있는 경우는 (docx, xlsx 등) : 헤더로 위치를 찾아낸 뒤, 첫 번째 부분

```
by@ubuntu:~/dump$ xxd  recover_ms_02
00000000: 504b 0304 1400 0808 0800 f604 484f 0000  PK..........HO..
00000010: 0000 0000 0000 0000 0000 1000 0000 776f  ..............wo
00000020: 7264 2f66 6f6f 7465 7231 2e78 6d6c a595  rd/footer1.xml..
```


```
by@ubuntu:~/dump$ xxd  recover_ms_02
00000000: 504b 0304 1400 0808 0800 f604 484f 0000  PK..........HO..
00000010: 0000 0000 0000 0000 0000 1000 0000 776f  ..............wo
00000020: 7264 2f66 6f6f 7465 7231 2e78 6d6c a595  rd/footer1.xml..
```


  를 통해 xl(xlsx), ppt(ppt), word(docx) 를 구분함. 0 padding 으로 구별하는 것이 가장 좋은 방법일 듯 함.

- Header 도 footer 도 없는 경우 (txt) : 는 삭제되었을 때 생긴 앞의 [Trash Info] = 5b54 7261 7368 2049 6e66 6f5d 0a50 를 인식하는 것은 어떤가 제안.

< 참고 >

1) Zip 시그니처 특징.

50 4B 03 04	PK..
ZIP	PKZIP archive file (Ref. 1 | Ref. 2)
	Trailer: *filename* 50 4B *17 characters* 00 00 00
	Trailer: (*filename* PK *17 characters* . . .)
	Note: PK are the initals of Phil Katz, co-creator of the ZIP fil format and author of PKZIP.
ZIP	Apple Mac OS X Dashboard Widget, Aston Shell theme, Oolite eXpansion Pack, Opera Widget, Pivot Style Template, Rockbox Theme package, Simple Machines Forums theme, SubEthaEdit Mode, Trillian zipped skin, Virtual Skipper skin
JAR	Java archive; compressed file package for classes and data
KMZ	Google Earth saved working session file

출처 :

http://forensic.korea.ac.kr/DFWIKI/index.php/%EB%8D%B0%EC%9D%B4%ED%84%B0_%EB%B3%B5

%EA%B5%AC

https://www.garykessler.net/library/file_sigs.html