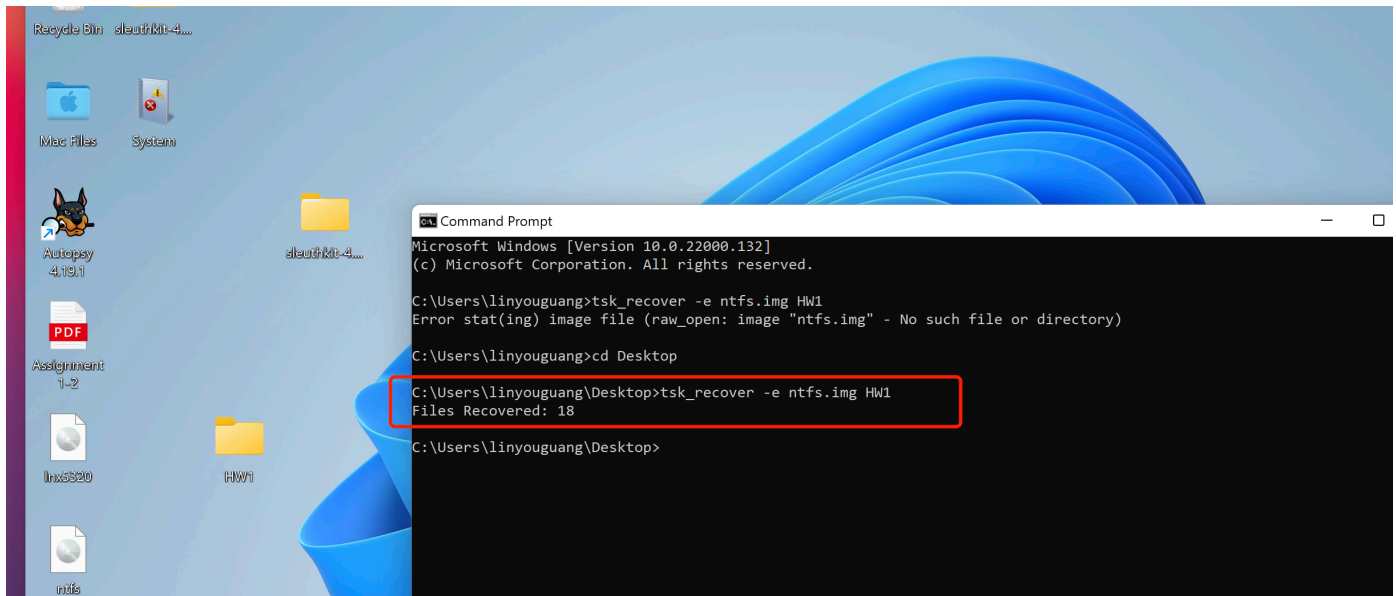


IERG5320, Fall 2021 Assignment1

- Name : Youguang Lin - Student ID : 1155169171

Q1.1

Firstly, I use command "tsk_recover -e ntfs.img HW1" to recover the img files.



Then I use command "exiftool .\System.evtx" to check the Windows event log file with its meta information, which includes filename, file size and the last modified date and time.

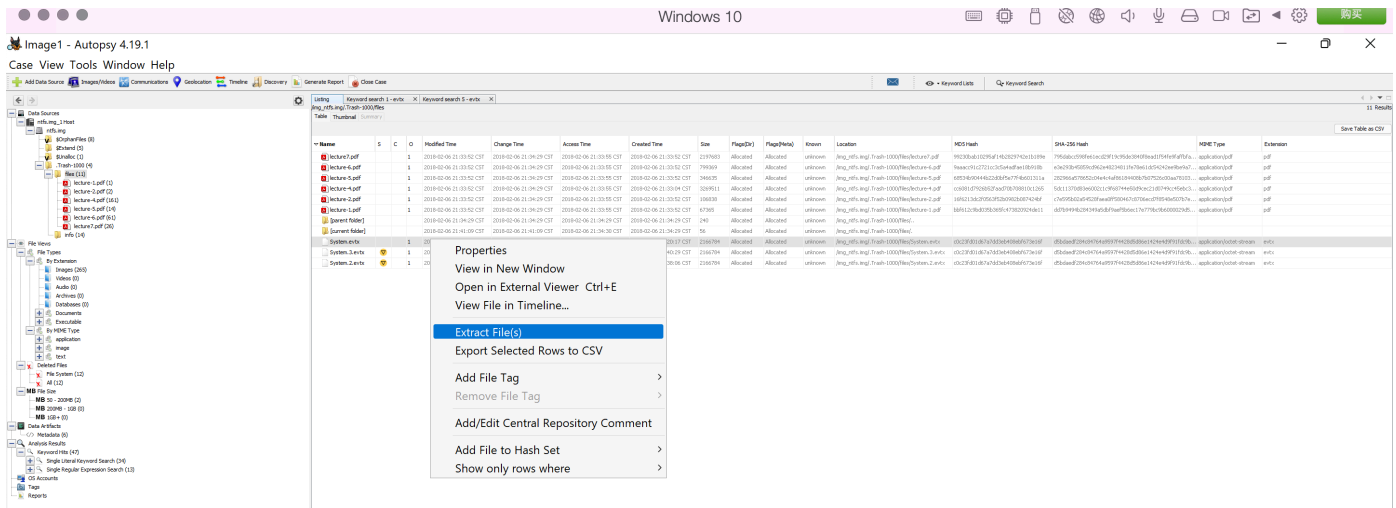
```
C:\Users\linyonguang\Desktop\HW1\.Trash-1000>cd files

C:\Users\linyonguang\Desktop\HW1\.Trash-1000\files>exiftool .\System.evtx
ExifTool Version Number      : 12.34
File Name                    : System.evtx
Directory                   : .
File Size                    : 2.1 MiB
File Modification Date/Time  : 2021:11:02 19:53:11+08:00
File Access Date/Time       : 2021:11:02 19:53:12+08:00
File Creation Date/Time     : 2021:11:02 19:53:11+08:00
File Permissions             : -rw-rw-rw-
Error                        : Unknown file type

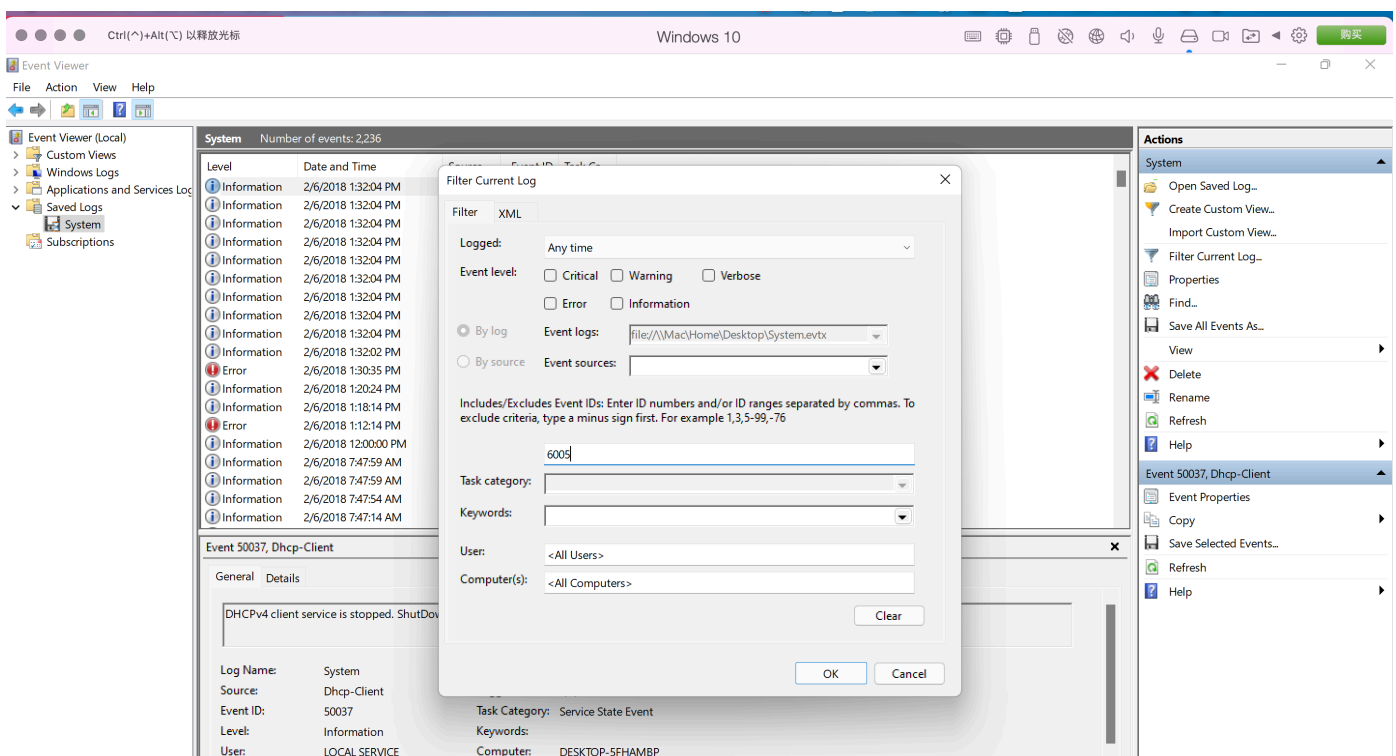
C:\Users\linyonguang\Desktop\HW1\.Trash-1000\files>
```

Q1.2

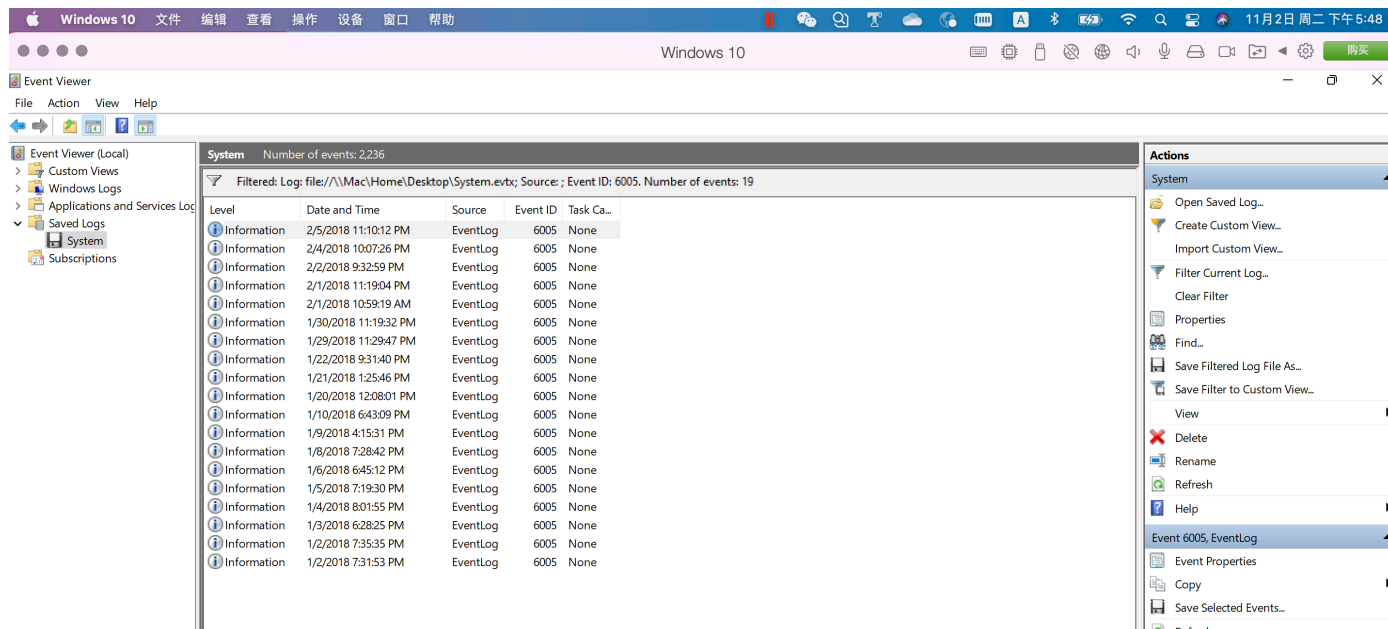
Then I use slueuthkit tool to extract the system.ectx to my disk.



Then I open this file on Window and filter the log by their Id. 6005 is the ID when Windows starts.



Finally, you can see the result.



Question 2.1

First recover the delete file and get file name ---- ierg5320.

```
PS C:\Users\linyonguang\Desktop> tsk_recover -e .\lnx5320.img HW2
Files Recovered: 1
PS C:\Users\linyonguang\Desktop> cd HW2
PS C:\Users\linyonguang\Desktop\HW2> ls

Directory: C:\Users\linyonguang\Desktop\HW2

Mode                LastWriteTime         Length Name
----                -
d-----           11/3/2021    7:54 PM             ierg5320

PS C:\Users\linyonguang\Desktop\HW2> |
```

We can also see the file name by using autopsy.

Autopsy 4.19.1 interface showing the File System view. The left pane displays the Data Sources tree, including File Types (By Extension, By MIME Type, Deleted Files, MB File Size, Data Artifacts) and Analysis Results (Keyword Hits, Single Literal Keyword Search, Single Regular Expression Search, Email Addresses, OS Accounts, Tags, Reports).

The right pane shows the File System table with columns: Name, S, C, O, Modified Time, Change Time, Access Time, Created Time, Size, Flags(Dir), Flags(Meta), Known, Location, and MD5H. The table lists files in the file system, including /img_jm5320_img/leg5320/flag.txt.

The bottom pane displays the Metadata for the selected file, /img_jm5320_img/leg5320/flag.txt. The metadata includes:

- Name: /img_jm5320_img/leg5320/flag.txt
- Type: File System
- None Type: 0x0000000000000000
- Size: 0
- File Name Allocation: Unallocated
- Metadata Allocation: Unallocated
- Modified: 2021-10-20 10:13:57 CST
- Accessed: 2021-10-20 10:11:29 CST
- Created: 0000-00-00 00:00:00
- Changed: 2021-10-20 10:13:57 CST
- MD5: d41d8cd98f00b204e9800998ecf8427e
- SHA-256: e3b0c44298fc1c149afebf4c9996fb92427ae41e4649b534ea495991b7852b855
- Hash Lookup Results: UNMATCHED
- Internal ID: 14

From The Sleuth Kit stat tool:

```
inode: 4099
Not Allocated
Group: 2
Ownership Id: 1507769356
uid / gid: 0 / 0
mode: r--r--r--
size: 0
```

Question 2.2

See what really happens on an Ext3 file system.

```

PS C:\Users\linyonguang\Desktop> ils -r .\lnx5320.img
class|host|device|start_time
ils|unknown||1635926692
st_ino|st_alloc|st_uid|st_gid|st_mtime|st_atime|st_ctime|st_crtime|st_mode|st_nlink|st_size
4098|f|0|0|1634696037|1634695889|1634696037|0|644|0|0
PS C:\Users\linyonguang\Desktop> istat .\lnx5320.img 4098
inode: 4098
Not Allocated
Group: 2
Generation Id: 1507768356
uid / gid: 0 / 0
mode: rrw-r--r--
size: 0
num of links: 0

Inode Times:
Accessed:      2021-10-20 10:11:29 (China Standard Time)
File Modified: 2021-10-20 10:13:57 (China Standard Time)
Inode Modified: 2021-10-20 10:13:57 (China Standard Time)
Deleted:       2021-10-20 10:13:57 (China Standard Time)

Direct Blocks:
PS C:\Users\linyonguang\Desktop> |

```

Look the stats of block group 2:

```
PS C:\Users\linyonguang\Desktop> fsstat .\lnx5320.img
FILE SYSTEM INFORMATION
-----
File System Type: Ext3
Volume Name: LNXExt3
Volume ID: b018e4395ef976b8ac426fb8cddcd373

Last Written at: 2021-10-20 10:13:49 (China Standard Time)
Last Checked at: 2021-10-20 09:58:28 (China Standard Time)

Last Mounted at: 2021-10-20 10:13:49 (China Standard Time)
Unmounted properly
Last mounted on: /mnt

Source OS: Linux
Dynamic Structure
Compat Features: Journal, Ext Attributes, Resize Inode, Dir Index
InCompat Features: Filetype, Needs Recovery,
Read Only Compat Features: Sparse Super, Large File,

Journal ID: 00
Journal Inode: 8

METADATA INFORMATION
-----
Inode Range: 1 - 16385
Root Directory: 2
Free Inodes: 16370

CONTENT INFORMATION
-----
Block Range: 0 - 65535
Block Size: 1024
Reserved Blocks Before Block Groups: 1
Free Blocks: 56008

BLOCK GROUP INFORMATION
-----
Number of Block Groups: 8
```

Take a look closely to the information of group 2

Group: 2:

Inode Range: 4097 – 6144

Block Range: 16385 – 24576

Layout:

Data bitmap: 16385 – 16385

Inode bitmap: 16386 – 16386

Inode Table: 16387 – 16898

Data Blocks: 16387 – 16386, 16899 – 0

Free Inodes: 2046 (0%)

Free Blocks: 7676 (0%)

Total Directories: 1

We can see that for this group there are 2048 nodes (Inode range: 4097 - 6144) and the inode table has a size of 512 blocks (Inode Table: 16387 - 16898). Each block of the inode table has 4 nodes (2048 divided by 512), thus inode 4098 it's the 2nd entry in the table and its content is located in the first block of the inode table.

Checking the output from jls, we find out that there are several references to 16387(the first block of the inode table). The output shows that block 2 of the journal contains information regarding an operation on the inode table of group 2 and since the journal at least records copies of the metadata (default journal mode is ordered) that has been modified; we can look for a copy of inode 4098 within the journal. There are cases that checking each instance of a particular block in the journal must be analyzed, but in this case we just want to recover the earliest one.

```
Windows PowerShell
PS C:\Users\linyonguang\Desktop> jls .\lnx5320.img
JBlk    Description
0:      Superblock (seq: 0)
sb version: 4
sb version: 4
sb feature_compat flags 0x00000000
sb feature_incompat flags 0x00000000
sb feature_ro_incompat flags 0x00000000
1:      Allocated Descriptor Block (seq: 10)
2:      Allocated FS Block 16387
3:      Allocated FS Block 16899
4:      Allocated FS Block 1
5:      Allocated FS Block 16385
6:      Allocated FS Block 2
7:      Allocated FS Block 16386
8:      Allocated Commit Block (seq: 10, sec: 1634696039.4184827648)
9:      Unallocated Commit Block (seq: 5, sec: 1634695618.2196018432)
10:     Unallocated Descriptor Block (seq: 6)
11:     Unallocated FS Block 16386
12:     Unallocated FS Block 2
13:     Unallocated FS Block 16387
14:     Unallocated FS Block 16899
15:     Unallocated FS Block 1
16:     Unallocated FS Block 16385
17:     Unallocated Commit Block (seq: 6, sec: 1634695893.3068617472)
18:     Unallocated Descriptor Block (seq: 7)
19:     Unallocated FS Block 16386
20:     Unallocated FS Block 2
21:     Unallocated FS Block 16387
22:     Unallocated FS Block 16899
23:     Unallocated FS Block 16385
24:     Unallocated Commit Block (seq: 7, sec: 1634695930.3043835648)
25:     Unallocated FS Block Unknown
26:     Unallocated FS Block Unknown
27:     Unallocated FS Block Unknown
28:     Unallocated FS Block Unknown
29:     Unallocated FS Block Unknown
30:     Unallocated FS Block Unknown
31:     Unallocated FS Block Unknown
```

We found there are three rows contain 16387, this is the start of the Inode Table of group 2. Because we want to recover the final one. The first entry, I think it is to create the file because it is the action of allocate. So the unallocated means delete. I use jcat to extract the copy of inode 4098. I use hex format.


```
PS C:\Users\linyonguang\Desktop> jcat lnx5320.img 21 | Format-Hex
```

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000000	3F	41	00	00	00	04	00	00	3F	7A	6F	61	3F	7A	6F	61	?A.....?zoa?zoa
00000010	3F	7A	6F	61	00	00	00	00	00	00	02	00	02	00	00	00	?zoa.....
00000020	00	00	00	00	03	00	00	00	03	42	00	00	00	00	00	00B.....
00000030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000060	00	00	00	00	3F	3F	3F	25	00	00	00	00	00	00	00	00???.%.....
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000080	20	00	00	00	38	77	30	7D	38	77	30	7D	3F	19	1F	4B	...8w0}8w0}?...K
00000090	3F	79	6F	61	3F	3F	3F	03	00	00	00	00	00	00	00	00	?yoa???.....
000000A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000100	3F	3F	00	00	47	00	00	00	3F	7A	6F	61	3F	7A	6F	61	??..G...?zoa?zoa
00000110	3F	7A	6F	61	00	00	00	00	00	00	01	00	02	00	00	00	?zoa.....
00000120	00	00	00	00	01	00	00	00	01	44	00	00	00	00	00	00D.....
00000130	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000140	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000150	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000160	00	00	00	00	24	3F	3F	59	00	00	00	00	00	00	00	00\$??Y.....
00000170	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000180	20	00	00	00	3F	3F	5B	27	3F	3F	5B	27	3F	29	7D	3F	...??['??['?)}?
00000190	3F	7A	6F	61	3F	3F	5B	27	00	00	00	00	00	00	00	00	?zoa??['.....
000001A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000001B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000001C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000001D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000001E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000001F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000200	3F	3F	00	00	3C	00	00	00	3F	7A	6F	61	3F	7A	6F	61	??..<...?zoa?zoa
00000210	3F	7A	6F	61	00	00	00	00	00	00	01	00	02	00	00	00	?zoa.....
00000220	00	00	00	00	01	00	00	00	01	46	00	00	00	00	00	00F.....
00000230	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

According to the i-node data structure, I found three single indirect pointer.

```
Windows PowerShell
000001B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000001C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000001D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000001E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000001F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000200 3F 3F 00 00 3C 00 00 00 3F 7A 6F 61 3F 7A 6F 61 ??..<...?zoa?zoa
00000210 3F 7A 6F 61 00 00 00 00 00 00 01 00 02 00 00 00 ?zoa.....
00000220 00 00 00 00 01 00 00 00 01 46 00 00 00 00 00 00 .....F.....
00000230 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000240 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000250 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000260 00 00 00 00 3A 3F 2E 3F 00 00 00 00 00 00 00 00 .....:?.?.....
00000270 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000280 20 00 00 00 38 77 30 7D 38 77 30 7D 38 77 30 7D ...8w0}8w0}8w0}
00000290 3F 7A 6F 61 38 77 30 7D 00 00 00 00 00 00 00 00 ?zoa8w0}.....
000002A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000002B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000002C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000002D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000002E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000002F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000300 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000310 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000320 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000330 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000340 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000350 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000360 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000370 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000380 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000390 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000003A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000003B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000003C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000003D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000003E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000003F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....

PS C:\Users\linyonguang\Desktop>
```

In linux, it use Little Endian. For the first one, i can see the 0342, so it is 4203 in hex, i convert it into decimal is 16899 block. That is the 4097 inode. But I want to find 4098.

```
PS C:\Users\linyonguang\Desktop> istat .\lnx5320.img 4097
inode: 4097
Allocated
Group: 2
Generation Id: 635149009
uid / gid: 0 / 0
mode: drwxr-xr-x
size: 1024
num of links: 2

Inode Times:
Accessed:      2021-10-20 10:13:53 (China Standard Time)
File Modified: 2021-10-20 10:13:57 (China Standard Time)
Inode Modified: 2021-10-20 10:13:57 (China Standard Time)

Direct Blocks:
16899
PS C:\Users\linyonguang\Desktop> |
```

Then the third one is 0146. It is the 4601 in hex and 17921 in decimal. Then, we check its block. It's the dummy.txt. It is not the inode I want to find.

```
C:\Users\linyonguang\Desktop>ffind -a lnx5320.img 4098
* /ierg5320/flag.txt

C:\Users\linyonguang\Desktop>fls -r lnx5320.img
d/d 11: lost+found
d/d 4097:      ierg5320
+ r/r * 4098:  flag.txt
+ r/r 4099:    dummy.txt
V/V 16385:     $OrphanFiles

C:\Users\linyonguang\Desktop>
```

```

PS C:\Users\linyonguang\Desktop> istat .\lnx5320.img 4099
inode: 4099
Allocated
Group: 2
Generation Id: 3056517178
uid / gid: 0 / 0
mode: rrw-r--r--
size: 60
num of links: 1

Inode Times:
Accessed:      2021-10-20 10:12:04 (China Standard Time)
File Modified: 2021-10-20 10:12:04 (China Standard Time)
Inode Modified: 2021-10-20 10:12:04 (China Standard Time)

Direct Blocks:
17921
PS C:\Users\linyonguang\Desktop> |

```

Then is the second one. It is 4401 in hex and 17409 in decimal. Then we use blkcat to see the content in this block. And I found it.

```

000003C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000003D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000003E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000003F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....

PS C:\Users\linyonguang\Desktop> blkcat .\lnx5320.img 17409
find me if you can, dear IERG5320 students. Token is 20211020 10:11:43
PS C:\Users\linyonguang\Desktop> |

```

It is funny!

We can also get all the unallocate into a file and find the possible answer. I use blkls to get the unallocate data. And found the result in the blkls.

æviâûè) r! ?nûb 000w f| ðò7hç@ø%επ- (L»µ| F#-TR° 1γ Tv1L Éx«ⁿU(A|8†|yü| βà-ç||q≥:ø!■ mù=ëÖ\$| RûMoáÇ||"♠2■|ûàV| 3α||L|÷|I||ø-||°â-|√{?-?|ûe-| E!■|

3: p p p: + à 3 ■ r r f r | n c r b 9 j G ° | g ! j L ñ ■ | p | ò | Æ | o | Ò | - y | 5 ò N Æ ε j | 7 . | n | ■ | L % < ■ . | ? i â l » i â | y
 ^ R H G l è j ! | ò N Æ Z â | Ø ò + | 7 | Σ Ø r = 7 ö 8 Ü Ü Ü Ü Æ y j | Ø L \ ? Σ π Y | ■

= [ñz]6 [-y] EU{δ [3+ +böd]

`PyëuIAÄÄ i) r-rZöb100 'oð|çTR,Ü≤!-L=r-@çað»≈öæ*Ü≥Ñþiö @~\hf%7í! ê-eLoΓ^J70tä-find me if you can, dear IERG5320 students. Token is 20211020 10:11:43`