Lab 05 – File Deletion

Raymond Ng: JQG999
IS 3033-CY1 – Summer 2022
July 13, 2022

INTRODUCTION

The purpose of this lab is to allow users to familiarize with issues related to file deletion. Moreover, the lab the allows students to experiment with virtual hard disk, or disk image. Further, this lab will allow users to develop a better understanding of how data and metadata can be organized and managed on disk to file system.

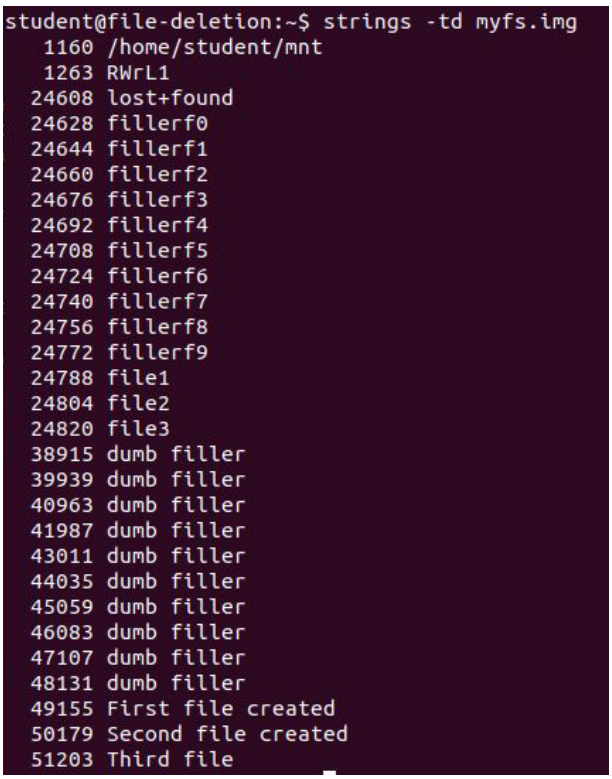
PROCEDURE

Question 1: Record the size of each file below, along with a screenshot of your command and output.

```
student@file-deletion:~$ sudo mount -o loop myfs.img mnt
student@file-deletion:~$ ll mnt/
total 34
drwxr-xr-x 3 student root     1024 Jul 13 13:07 ./
drwxr-xr-x 1 student student  4096 Jul 13 13:09 ../
-rw-rw-r-- 1 student student    25 Jul 13 13:07 file1
-rw-rw-r-- 1 student student    26 Jul 13 13:07 file2
-rw-rw-r-- 1 student student    17 Jul 13 13:07 file3
-rw-rw-r-- 1 student student    18 Jul 13 13:07 fillerf0
-rw-rw-r-- 1 student student    18 Jul 13 13:07 fillerf1
-rw-rw-r-- 1 student student    18 Jul 13 13:07 fillerf2
-rw-rw-r-- 1 student student    18 Jul 13 13:07 fillerf3
-rw-rw-r-- 1 student student    18 Jul 13 13:07 fillerf4
-rw-rw-r-- 1 student student    18 Jul 13 13:07 fillerf5
-rw-rw-r-- 1 student student    18 Jul 13 13:07 fillerf6
-rw-rw-r-- 1 student student    18 Jul 13 13:07 fillerf7
-rw-rw-r-- 1 student student    18 Jul 13 13:07 fillerf8
-rw-rw-r-- 1 student student    18 Jul 13 13:07 fillerf9
drwx------ 2 root    root    12288 Jul 13 13:07 lost+found/
```

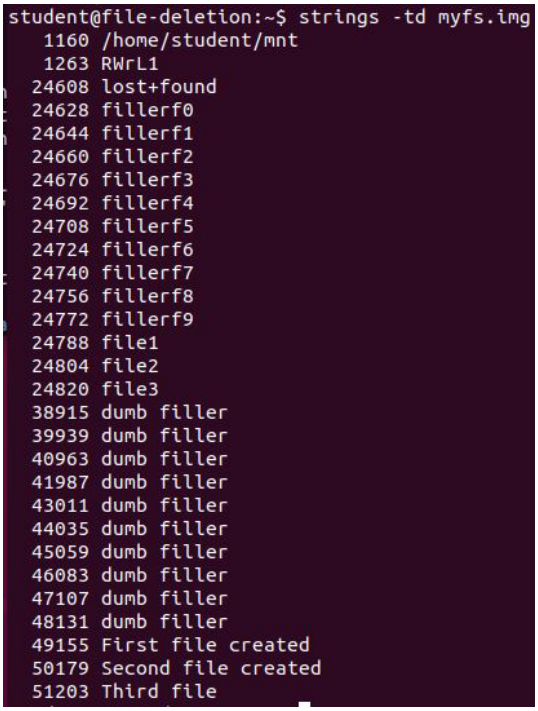
File Name	File Content	File size (bytes)
`file1`	First file created	25
`file2`	Second file created	26
`file3`	Third file	17

Question 2: Record the data and filename offsets reported by the strings command (as seen before a file was deleted) along with a screenshot of your command and output.



Offset	String
1160	/home/student/mnt
1263	RWrL1
24608	lost+found
24628	fillerf0
24644	fillerf1
24660	fillerf2
24676	fillerf3
24692	fillerf4
24708	fillerf5
24724	fillerf6
24740	fillerf7
24756	fillerf8
24772	fillerf9
24788	file1
24804	file2
24820	file3
38915	dumb filler
39939	dumb filler
40963	dumb filler
41987	dumb filler
43011	dumb filler
44035	dumb filler
45059	dumb filler
46083	dumb filler
47107	dumb filler
48131	dumb filler
49155	First file created
50179	Second file created
51203	Third file

Question 3: Record the data and filename offsets reported by the strings command (as seen after file2 was deleted), along with a screenshot of your command and output.



Offset	String
1160	`/home/student/mnt`
1263	`RWrL1`
24608	`lost+found`
24628	`fillerf0`
24644	`fillerf1`
24660	`fillerf2`
24676	`fillerf3`
24692	`fillerf4`
24708	`fillerf5`
24724	`fillerf6`
24740	`fillerf7`
24756	`fillerf8`
24772	`fillerf9`
24788	`file1`
24804	`file2`
24820	`file3`
38915	`dumb filler`
39939	`dumb filler`
40963	`dumb filler`
41987	`dumb filler`
43011	`dumb filler`
44035	`dumb filler`
45059	`dumb filler`
46083	`dumb filler`
47107	`dumb filler`
48131	`dumb filler`
49155	`First file created`
50179	`Second file created`
51203	`Third file`

QUESTION 4: Item #2 of the worksheet shows the output of the strings command before file2 was deleted, while item #3 shows the output after file2 was deleted. Compare these two outputs, and then describe your observations. What security problems, if any, are implied by your observations? I noticed that even after `file2` was removed using `rm` command in question #3, the outputs after executing `strings -td myfs.img` were the same. From what I read about the `rm` command is that is does not actually remove the files contents from the file system, but merely marks the occupied space as available (LinuxBSDox, 2021). Therefore, a deleted file can still be recovered. So, the contents of `file2` that were mounted to `myfs.img` still exist and the `string` command was able to extract the contents of the file as depicted in the terminal window.

I assess that this could potentially be a security problem because if the user was debugging and intentionally wanted to delete the file that was producing unfavorable output, the issue was not fully resolved. Moreover, it creates a potentially vulnerability in the file system that allows the contents of the file that may have been causing issues to be recovered and reintroduced back into the system. In this scenario, if `file2` was intentionally deleted during debugging, and its content still exists, then a vulnerability exists in the file system.

Task 4: Securely Deleting a File on Unix. What camp are you typically in? I think I would be worrisome knowing that something that was deleted may still be there.

QUESTION #5: Record the data and filename offsets reported by the strings command, (as seen after file3 was "shredded"), along with a screenshot of your command and output.

```
student@file-deletion:~$ strings -td myfs.img
   1160 /home/student/mnt
   1263 RWrL1
  24608 lost+found
  24628 fillerf0
  24644 fillerf1
  24660 fillerf2
  24676 fillerf3
  24692 fillerf4
  24708 fillerf5
  24724 fillerf6
  24740 fillerf7
  24756 fillerf8
  24772 fillerf9
  24788 file1
  24804 0000
  24820 00003
  38915 dumb filler
  39939 dumb filler
  40963 dumb filler
  41987 dumb filler
  43011 dumb filler
  44035 dumb filler
  45059 dumb filler
  46083 dumb filler
  47107 dumb filler
  48131 dumb filler
  49155 First file created
  50179 Second file created
```

Offset	String
1160	/home/student/mnt
1263	RWrL1
24608	lost+found
24628	fillerf0
24644	fillerf1
24660	fillerf2
24676	fillerf3
24692	fillerf4
24708	fillerf5
24724	fillerf6
24740	fillerf7
24756	fillerf8
24772	fillerf9
24788	file1
24804	0000
24820	00003
38915	dumb filler
39939	dumb filler
40963	dumb filler
41987	dumb filler
43011	dumb filler
44035	dumb filler
45059	dumb filler
46083	dumb filler
47107	dumb filler
48131	dumb filler
49155	First file created
50179	Second file created

QUESTION 6: Record the file sizes reported by the `ll` command, along with a screenshot of your command and output.

```
student@file-deletion:~$ sudo mount -o loop ntfs.img mnt
student@file-deletion:~$ ll mnt/
total 14
drwxrwxrwx 1 root    root    4096 Jul 13 13:07 ./
drwxr-xr-x 1 student student 4096 Jul 13 14:46 ../
-rwxrwxrwx 1 root    root      25 Jul 13 13:07 file1*
-rwxrwxrwx 1 root    root      26 Jul 13 13:07 file2*
-rwxrwxrwx 1 root    root      17 Jul 13 13:07 file3*
```

File Name	File Content	File size (bytes)
file1	First file created	25
file2	Second file created	26
file3	Third file	17

QUESTION 7: Record the data and filename offsets reported by the strings command (in the NTFS), along with a screenshot of your command and output.

```
student@file-deletion:~$ strings -td ntfs.img | grep file
  82283 First file created
  83307 Second file created
```

Offset	String
82283	First file created
83307	Second file created

QUESTION 8: Report the inode numbers associated with `file1`.

```
student@file-deletion:~$ ntfsundelete -p 100 ntfs.img
Inode     Flags  %age      Date     Time        Size  Filename
-------------------------------------------------------------------------------
64        FR..   100%  2022-07-13 13:07         25  file1
66        FR..   100%  2022-07-13 15:09          0  0

Files with potentially recoverable content: 2
```

Inode number associated with `file1`: 64

QUESTION 9: Include a screenshot of #8 and #9.

```
student@file-deletion:~$ ntfsundelete --undelete --inodes 64 --output rfile1 ntfs.img
Inode     Flags  %age  Date            Size  Filename
-----------------------------------------------------------
64        FR..     0%  2022-07-13 13:07        25  file1
Undeleted 'file1' successfully.
student@file-deletion:~$ ll
total 2216
drwxr-xr-x 1 student student    4096 Jul 13 15:21 ./
drwxr-xr-x 1 root    root       4096 Jan 24  2020 ../
-rw------- 1 student student     742 Jul 13 15:21 .bash_history
-rw-r--r-- 1 student student     220 Aug 31  2015 .bash_logout
-rw-r--r-- 1 student student    3921 Jul 13 13:07 .bashrc
drwxr-xr-x 1 student student    4096 Jul 13 13:07 .local/
-rw-r--r-- 1 student student     980 Jul 13 13:07 .profile
-rw-r--r-- 1 student student       0 Jul 13 13:07 .sudo_as_admin_successful
drwxrwxr-x 2 student student    4096 Jul 13 13:07 mnt/
-rw-rw-r-- 1 student student 1048576 Jul 13 14:57 myfs.img
-rw-rw-r-- 1 student student 2097152 Jul 13 15:09 ntfs.img
-rw------- 1 student student      25 Jul 13 13:07 rfile1
-rw-rw-r-- 1 student student      26 Jul 13 14:46 rfile2
student@file-deletion:~$ cat rfile1
“First file created”
```

QUESTION #10: Explain the meaning of each command below, along with its switches/options and arguments/filenames.

`dd if=myfs.img bs=1 skip=SKIPNUMBER count=FILESIZE of=rfile2`
In task 3, I learned that the `dd` command is used to copy the data from the location on the disk to a new file. I learned that `if` is command in Linux used to execute commands based on conditions, so `if=myfs.img`, it will be taking the input file, `myfys.img`, as file name to be converted/copied (GeeksforGeeks, 2019). `bs` mean the read and write up to bytes at a time, so `bs=1` means 1 byte will be read at a time from the file `myfs.img` (StackExchange, 2020). `skip`, under `dd`, is used to skip some initial bytes while reading input text, so `skip=SKIPNUMBER`, will skip `SKIPNUMER` number of one byte sized—as designated by `bs=1`—blocks while reading `myfs.img` (HowtoForge, n.d.). Executing `of=rfile2`, each copied block is saved to the file name `rfile2`.

`sudo mount -o loop myfs.img mnt`
`mount` command, executed under root privileges (`sudo`), allows me to mount our virtual disks to mount points, or directories. In this case `myfs.img` will be mounted to `mnt` directory as my mount point. I couldn't figure out `-o loop` command, but I assess, from the way it's written, it likely a device that mounts `myfs.img` to `mnt` (GeeksforGeeks, 2019).

`shred -uxz mnt/file3`
`shred` command is used to overwrite a file to hide its contents, and optionally delete it (Computer Hope, 2021). `-uxz`, coupled with `shred`, will truncate or remove file after overwriting it (`-u`), don't round the files up to the next full block (`x`), and add a final overwrite with zeros to hide shredding (`z`) (Hira, 2022). Basically, this command is securely deleting `file3` in `mnt`.

`ntfsundelete --undelete --inodes INODE --output rfile1 ntfs.img`
the `ntfsundelete` device is used recover deleted files from NTFS file system. It also has three modes of operation; scan, undelete, and copy (die.net, n.d.). Interpreting the command logically, its recovering a file (ntfsundelete) via undelete mode (--undelete), specified by a specific inode number (`--indoes INODE`), setting the name of the output file (`--output`) as `rfile1` in `ntfs.img`. So going back to Task 5, Step 7 of this lab I undeleted `file1` and recovered its contents.

CONCLUSION

The goals of this lab were to allow to users to experiment with virtual hard disk (or disk image) and to familiarize users with some issues related to file deletion in a controlled environment via virtual machine. I'm sure these problem sets are encountered routinely by professionals working in the cyber domain from users either inadvertently deleting and/or purposefully deleting files to debug/troubleshoot.

I thought the techniques presented, in my opinion, were very effective. Especially learning about the difference between `rm` and `shred` commands as they pertain to file deletion. I learned that using rm simply removes the pointer to the file system; however, the data may stay exist. On the other hand, using shred, the file is overwritten a specified number of times in a way that the actual content may not be recoverable (Hira, How to Securely Erase a Disk and File using the Linux shred Command, 2022).

In my opinion, I thought the difficulty of this lab was easy to moderately difficult. The instructions were a lot clear than the previous lab. The questions I felt could have been worded better. Maybe provide some additional examples and/or content to promote the user/student to data mine in the right direction to answer the questions correctly. Not really sure what other aspects could have been explored more, maybe provide real life scenarios of file deletion that were favorable or not favorable (i.e. videos, etc.).

REFERENCES

Computer Hope. (2021, November 6). *Linux shred command*. Retrieved from computerhope.com: https://www.computerhope.com/unix/shred.htm

die.net. (n.d.). *ntfsundelete(8) - Linux man page*. Retrieved from die.net: https://linux.die.net/man/8/ntfsundelete#:~:text=ntfsundelete%20has%20three%20modes%20of%20operation%3A%20scan%2C%20undelete%20and%20copy.&text=The%20default%20mode%2C%20scan%20simply,inode%20number%2C%20name%20and%20size.

GeeksforGeeks. (2019, May 19). *if command in linux with examples*. Retrieved from GeeksforGeeks.org: https://www.geeksforgeeks.org/if-command-in-linux-with-examples/#:~:text=if%20is%20a%20command%20in,then%20COMMANDS'%20list%20is%20executed.

GeeksforGeeks. (2019, May 23). *mount command in Linux with Examples*. Retrieved from geeksforgeeks.org: https://www.geeksforgeeks.org/mount-command-in-linux-with-examples/

Hira, Z. (2022, March 14). *How to Securely Erase a Disk and File using the Linux shred Command*. Retrieved from freecodecamp.org: https://www.freecodecamp.org/news/securely-erasing-a-disk-and-file-using-linux-command-shred/

Hira, Z. (2022, March 14). *How to Securely Erase a Disk and File using the Linux shred Command*. Retrieved from freecodecamp.org: https://www.freecodecamp.org/news/securely-erasing-a-disk-and-file-using-linux-command-shred/#:~:text=Simply%20using%20rm%20removes%20the,the%20actual%20content%20is%20unrecoverable.

HowtoForge. (n.d.). *Linux dd Command Explained for Beginners (8 Examples)v*. Retrieved from howtoforge.com: https://www.howtoforge.com/linux-dd-command/

StackExchange. (2020, June 12). *What is the 'bs' option in dd?* Retrieved from askubuntu.com: https://askubuntu.com/questions/1021607/what-is-the-bs-option-in-dd

Collaboration

The entirety of this lab was performed independently coupled with user data-mining via the internet. No additional collaboration.